

UNIVERSIDAD NACIONAL DE INGENIERÍA
FACULTAD DE CIENCIAS



TESIS

**“Análisis de muestras de suelos de minas a partir de imágenes
hiperespectrales”**

PARA OBTENER EL GRADO ACADÉMICO DE DOCTOR EN
CIENCIAS CON MENCIÓN EN FÍSICA

ELABORADA POR:

Leonardo José Chávez Abanto

ASESOR:

Dr. Héctor Raúl Loro Ramírez

LIMA – PERÚ

2021

AGRADECIMIENTOS

En estos tiempos tan agitados donde la economía personal juega un rol importante en los quehaceres, tendemos a dejar la educación de lado para dedicarnos a actividades que nos produzcan bienestar económico.

El seguir estudios de posgrado y dedicarme seriamente al trabajo de tesis con la respectiva compensación económica me fue posible mediante una beca integral otorgada por el Consejo Nacional de Ciencia y Tecnología (CONCYTEC).

Agradezco a mi asesor Héctor Loro, a la Facultad de Ciencias de la Universidad Nacional de Ingeniería y a CONCYTEC por hacer posible mi grado de Doctor en Ciencias con Mención en Física.

ÍNDICE DE CONTENIDOS

INTRODUCCIÓN	1
CAPÍTULO I	
Sistemas de Registro de imágenes Hiperespectrales	3
1.1 Imagen Hiperespectral (HSI)	3
1.1.1 Espectros de reflectancia e imágenes HSI	4
1.1.2 Espectros de reflectancia de minerales	5
1.2 Registro de imágenes hiperespectrales	7
1.2.1 Imágenes satelitales	7
1.2.2 Cámaras Hiperespectrales	10
CAPÍTULO II	
Componentes de imágenes RGB. Descomposición de imágenes Multiespectrales e Hiperespectrales en Series de Fourier	12
2.1 Imágenes RGB	12
2.2 Imágenes Multiespectrales	14
2.3 Espectros a partir de las intensidades de radiación de los píxeles	15
2.4 Descomposición espectral de una imagen usando la base de datos ASTER	17
2.5 Algoritmos matriciales e imágenes multiespectrales	22
2.5.1 Reducción de la dimensión de un espectro base	22
2.6 Descomposición espectral usando Series de Fourier	27
2.7 Serie de Fourier de un espectro	28
2.8 Descomposición de un espectro representado por una Serie de Fourier en componentes base	32
CAPÍTULO III	
Tratamiento estadístico de imágenes hiperespectrales	39
3.1 Datos obtenidos en una imagen hiperespectral	39
3.2 Análisis de componentes principales (PCA)	40
3.3 Fracción Mínima de ruido (MNF)	42
3.3.1 Estimación de la matriz de covarianza del ruido	44
3.4 Desmezclado de un espectro HSI	45
3.4.1 Pre-Procesamiento de datos	46
3.4.2 Simplex Growing Algorithm (SGA), endmembers y HIS	47

3.4.3 Descomposición lineal usando simulated annealing y análisis subpixel	52
CAPÍTULO IV	
Resultados	54
4.1 Imágenes RGB	54
4.2 Imágenes multiespectrales promedio y descomposición espectral	56
4.3 Registro de la HSI con la cámara PIKA NIR	59
4.3.1 Montaje experimental y registro de imágenes	60
4.4 Imágenes Multiespectrales Promedio y Series de Fourier	61
4.4.1 Representación de un espectro por una Serie de Fourier	61
4.4.2 Descomposición de un espectro en una Serie de Fourier	62
4.4.3 Descomposición de un espectro representado por una serie de Fourier en componentes base	63
4.5 Análisis Estadístico de imágenes Hiperespectrales	69
4.5.1 Software desarrollado para el análisis de las imágenes	70
4.5.2 Pre-procesamiento de la imagen	70
4.5.3 Determinación de los endmembers y cálculo de las abundancias	71
Conclusiones	75
Referencias	76
ANEXOS	79

ÍNDICE DE TABLAS

Tabla 1. Bandas de Sentinel-2 y sus correspondientes longitudes de onda centrales	8
Tabla 2. Especificaciones técnicas de la Cámara RESONON Pika NIR 320	59
Tabla 3. Resultados de los ensayos de identificación de elementos	67
Tabla 4. Comparación de las abundancias de los materiales considerados para la muestra D con sus resultados por DRX	68
Tabla 5. Abundancias de los principales elementos identificados por DRX de la muestra en estudio	69
Tabla 6. Abundancias de calcita, dolomita y cuarzo obtenidos por DRX y por un análisis HSI (considerando VD = 5, 10, 15, 20, 25, 30)	73

ÍNDICE DE FIGURAS

Figura 1. Distintas formas de obtener una imagen hiperespectral: puntual, lineal y por snapshot	4
Figura 2. Patrón característico del material	5
Figura 3. Materiales comunes en la superficie terrestre	5
Figura 4. Patrón de algunos minerales comunes	6
Figura 5. Satélite de la misión Sentinel-2	7
Figura 6. Imagen de la página de registro en ESA para el acceso a su base de datos	8
Figura 7. Fotografía “completa” alrededor de 41,449° 41,755°	9
Figura 8. Fotografía con la banda 4	9
Figura 9. Fotografía con la banda 7	10
Figura 10. Fotografía de la Facultad de Ciencias en formato jpg	12
Figura 11. Descomposición de fc.jpg en sus contribuciones RGB	14
Figura 12. Algunas imágenes de la misión Sentinel-2. Banda B1, B4, B6 y B8 respectivamente	15
Figura 13. Espectros correspondientes a los píxeles (3500,750) y (4500,1750)	17
Figura 14. Espectros correspondientes a los píxeles (5580,350) y (6111,1350)	17
Figura 15. Espectro de reflectancia del Rutilo de la base de datos ASTER	18
Figura 16. Recorde de pantalla de los datos de reflectancia en forma de tabla	18
Figura 17. Azul: Espectro completo de la alpita. Rojo: Espectro reducido de la alpita	23
Figura 18. Dirección de máxima variabilidad (P_1) y la dirección ortogonal (P_2) de mínima variabilidad en el espacio X_1 - X_2	40
Figura 19. Comparación del número de componentes significativas en PCA y MNF	42
Figura 20. Diagrama que muestra la secuencia seguida en este trabajo para determinar las abundancias de minerales en una muestra	45
Figura 21. a) Espectros de reflectancia sin procesar y b) Espectros de reflectancia procesados	46
Figura 22. a) Diagrama esquemático de un simplex en dos dimensiones. b) Diagrama esquemático de un simplex en tres dimensiones	51
Figura 23. Diagrama esquemático que muestra un píxel con tres componentes diferentes	52
Figura 24. Recorte de pantalla de la matriz L conteniendo los valores de los elementos de las matrices R, G y B de cada píxel	55

Figura 25. Variación de la intensidad del pixel 1 y del pixel 646	55
Figura 26. Espectro total de un pixel al azar	56
Figura 27. Espectros reducidos de algunos minerales elegidos	57
Figura 28. Cámara hiperespectral Pika NIR 320	59
Figura 29. Montaje experimental donde se muestra la muestra y el sistema de adquisición de Laboratorio que usa una cámara PIKA NIR	60
Figura 30. Espectro de una muestra obtenido con la cámara hiperespectral	61
Figura 31. Serie de Fourier con $k=1$, $r^2=0.85$ y $k=10$, $r^2=0.94$ respectivamente	62
Figura 32. Serie de Fourier con $k=30$, $r^2=0.98$	62
Figura 33. Software SpectronPro en funcionamiento	63
Figura 34. Espectros obtenidos de las muestras A y B en unidades arbitrarias	64
Figura 35. Espectros obtenidos de las muestras C y D en unidades arbitrarias	65
Figura 36. Espectros obtenidos de las muestras E y F en unidades arbitrarias	66
Figura 37. Espectro DRX de la muestra D con carbonatos analizada en este trabajo. En rojo se observa el ruido después de un ajuste de Rietveld del espectro	69
Figura 38. Autovalores normalizados de los datos transformados usando MNF	71
Figura 39. Puntos del espacio de datos transformado, a) usando los autovectores e_1 y e_2 con los endmembers en color azul y el resto de píxeles en color rojo. b) Usado los autovectores e_1 , e_2 y e_3 con los endmembers en color azul y el resto de píxeles en color rojo	72
Figura 40. Espectros de reflectancia normalizados de píxeles clasificados como endmembers	73
Figura 41. Espectro de reflectancia del endmember 6 y el espectro ajustado usando las abundancias finales encontradas para $VD = 15$	74

Lista de términos y abreviaciones

HSI Imagen Hiperespectral

NIR Near infrared (Infrarrojo cercano)

UAV Unmanned Aerial Vehicle (Vehículo aéreo no tripulado)

SA Simulated Annealing

Snapshot Capturar muchas imágenes bidimensionales (una para cada longitud de onda λ) y luego construir una imagen tridimensional (dos dimensiones espaciales y una más de longitud de onda) que se le suele llamar datacube.

Datacube imagen tridimensional (dos dimensiones espaciales y una más de longitud de onda)

Scanning Captura una línea hiperespectral y a continuación la siguiente línea hasta completar toda la imagen.

ESA Agencia Espacial Europea

RGB Red, Green and Blue

FC Facultad de Ciencias

JPG

ASTER Spectral Library del Jet Propulsion Laboratory

NASA National Aeronautics and Space Administration (Administración Espacial Aeronáutica de Estados Unidos)

SNV Standard Normal Variance

MNF Minimum Noise Fraction

SALD Simulated Annealing Linear Decomposition.

PCA Principal Components Analysis

SNR Signal Noise Ratio (Relación señal Ruido)

SGA Simplex Growing Algorithm

DRX Difracción de Rayos X

VD Virtual Dimension

USGS United States Geological Survey

Resumen_: artículo

Abstract

RESUMEN

Determinar la composición y abundancias de minerales en una muestra es de particular interés para diversas áreas como la minería, geología, topografía, etc. Con la finalidad de encontrar estas abundancias, se realizaron ensayos preliminares con imágenes multiespectrales para luego proponer una descomposición por Series de Fourier aplicadas a imágenes hiperespectrales, finalmente se aplicó un método estadístico para encontrar las abundancias de los minerales presentes en la muestra.

El procedimiento estadístico aplicado fue el de dimensión virtual, usado para analizar los datos de la imagen hiperespectral provenientes de espectros de reflectancia (región NIR). Los datos se tratan previamente usando el método de variable normal estándar (SNV) y fracción de ruido mínimo (MNF) para finalmente encontrar las abundancias con el método de recocido simulado (SA). Las abundancias obtenidas con este método resultaron ser muy cercanas a las obtenidas por difracción de rayos X, con un error relativo total del 2%.

Palabras clave: imágenes hiperespectrales, Series de Fourier, recocido simulado.

ABSTRACT

Determining the composition and abundances of minerals in a sample is of particular interest for various areas such as mining, geology, topography, etc. In order to find these abundances, preliminary tests were carried out with multispectral images to later propose a decomposition by Fourier Series applied to hyperspectral images, finally a statistical method was applied to find the abundances of the minerals present in the sample.

The statistical procedure applied was virtual dimension, used to analyze the hyperspectral image data from reflectance spectra (NIR region). The data is pre-treated using the standard normal variate (SNV) and minimal noise fraction (MNF) method to finally find the abundances with the simulated annealing (SA) method. The abundances obtained with this method turned out to be very close to those obtained by X-ray diffraction, with a total relative error of 2%.

Keywords: hyperspectral images, Fourier Series, simulated annealing.

INTRODUCCIÓN

El estudio de la corteza terrestre es un tema de interés para una serie de aplicaciones relacionadas a la geología, topografía, minería, etc. El conocimiento de la estructura de la corteza terrestre permite, entre otras cosas, tomar decisiones respecto al aprovechamiento sostenible de recursos terrestres. Actualmente, estudiar la corteza demanda todo un equipo (tanto instrumental como personal) que, debe ir al campo y tomar mediciones y muestras para el respectivo análisis de las mismas. Este método implica tiempo, demanda de personal, sofisticados y pesados equipos y dinero. Adicionalmente, la toma de muestras da idea de la composición de la corteza, sin embargo, no permite conocer de manera global toda la zona de interés.

Una solución viable es el uso de métodos de prospección geofísica aplicados a la exploración de materiales dentro de la corteza terrestre. En particular, el uso de imágenes hiperespectrales aplicado a la prospección de recursos terrestres es un método con mucho potencial.

La técnica de imágenes hiperespectrales (HSI) se ha aplicado con éxito en diferentes campos como la agricultura [1], problemas del medio ambiente [2] así como en el estudio de la superficie terrestre permitiendo además encontrar los minerales que lo componen [3]. Una HSI consta de cientos de espectros de reflectancia contiguos y adyacentes de una escena que cubren bandas en el espectro electromagnético que van desde la región visible hasta la región NIR. Estas imágenes pueden ser recolectadas por sensores espaciales, aéreos basados en UAV (del inglés *UAV unmanned aerial vehicle*) o simplemente en el laboratorio usando una plataforma móvil donde se coloca la muestra bajo estudio. Por lo general, las bandas cubren el rango de 0,3 a 2,5 μm con un ancho de banda que varía entre 2 y 10 nm. Los espectros de reflectancia en estas bandas ofrecen una técnica rápida y no destructiva que proporciona información sobre la composición química de una muestra, en particular de muestras minerales de carbonatos como calcita, aragonito y dolomita [4-6]. Las imágenes HSI de las muestras no requieren una forma particular de las mismas, pudiendo ser polvos, arenas, superficies de rocas rotas, aserradas o pulidas entre otras. En este trabajo se analiza una muestra en polvo de carbonatos. La posibilidad de utilizar cámaras hiperespectrales para registrar imágenes HSI de una superficie, nos permite obtener un conjunto de espectros de reflectancia para cada pixel de la imagen los cuales al ser correlacionados usando herramientas estadísticas proporcionan información sobre los componentes minerales puros presentes en los granos de polvo en nuestro caso. Los

espectros de reflectancia han sido ampliamente usados para detectar la presencia de minerales [7][8][9], determinando y cuantificando las abundancias de minerales contenidos en las muestras en estudio. Al registrar el espectro de reflectancia de una muestra debe tenerse en cuenta que estos espectros pueden ser originados por un mineral puro o por una mezcla de varios minerales. Adicionalmente las reflectancias observadas depende del tamaño de grano de los minerales [10] y efectos de oscurecimiento tales como la disgregación espacial [11].

En este sentido, este trabajo de investigación propone la implementación de un método que permite el conocimiento “global” de la zona de la corteza terrestre de interés, haciendo uso de diversas “imágenes” (espectrometría de imagen o teledetección hiperespectral) tomadas desde cierta altura (imágenes satelitales o tomadas por un drone) que, en su procesamiento se utilicen modelos físicos y computacionales que permitan realizar la “reconstrucción” virtual de la corteza terrestre. Este trabajo tiene como base las imágenes hiperespectrales y el procesamiento de estas enfocado en la determinación de los materiales presentes en la cierta zona.

Una de las tareas fundamentales que se abordan es la cuantificación de las abundancias de minerales en una muestra de interés. Este problema se abordará de dos formas. Una primera forma de abordar este problema es considerando una descomposición directa del promedio de los píxeles de la imagen registrada construyendo una descomposición del espectro registrado en una serie de Fourier. A continuación, tomando en cuenta consideraciones físicas de los coeficientes del ajuste y minimizando el error relativo, considerando una base de referencia, en nuestro caso la base de espectros ASTER, se encuentran las abundancias buscadas.

Otra forma de abordar el problema es haciendo uso de la estadística y evaluando la correlación entre los distintos píxeles de la imagen para las bandas que se usan. La posibilidad de utilizar cámaras hiperespectrales para registrar imágenes HSI de una superficie, nos permite obtener un conjunto de espectros de reflectancia para cada píxel de la imagen los cuales al ser correlacionados usando herramientas estadísticas proporcionan información sobre los componentes minerales puros presentes en los granos de polvo en nuestro caso. A este proceso se le denomina unmixing process. Este proceso comienza con un preprocesamiento de las imágenes seguido de una identificación de píxeles endmembers, los cuales son un conjunto de píxeles similares presentes en la imagen en estudio. En general, los píxeles traen información de los componentes puros presentes en la muestra los cuales se desea identificar. Existen varios métodos para obtener esta información como el Simulated Annealing (SA) [12] el cual es un

medio eficaz y general de optimización inspirado en la metalurgia, donde la temperatura de un material determina su comportamiento termodinámico y ha sido aplicado con éxito en la teledetección multi-hiperespectral para la clasificación [13] [14] y la estimación de abundancias [15].

En este trabajo usamos SA para estimar las abundancias de los componentes puros presentes en los píxeles endmembers de las imágenes hiperespectrales de la muestra en estudio.

CAPÍTULO I:

Sistemas de Registro de imágenes Hiperespectrales

Se hace una revisión de los conceptos fundamentales a desarrollar en esta tesis, presentando las características de los elementos que se van a estudiar, así como las herramientas que se usan como son las cámaras hiperespectrales.

1.1 Imagen Hiperespectral (HSI)

Una cámara convencional de escala de grises produce una imagen que describe la intensidad como una función de dos variables espaciales $I(x,y)$ donde I representa la intensidad en el pixel ubicado en (x,y) . Por otra parte, las cámaras a color producen imágenes que son combinaciones de intensidades de tres colores: rojo, verde y azul; entonces estas cámaras producen intensidades de la forma $I(x,y,\lambda)$ donde λ es la longitud de onda, en este caso correspondiente al rojo, verde y azul (tres bandas) respectivamente. Mientras que las cámaras multiespectrales e hiperespectrales producen imágenes similares a las de una cámara a color, sin embargo, el número de bandas espectrales es mucho mayor. Generalmente se considera que, si el número de bandas es menor que 10, será una imagen multiespectral y será hiperespectral si el número de bandas es mayor que 10.

En el caso hiperespectral la idea es poder obtener el espectro completo de cada pixel perteneciente a la imagen, para ello hace falta tener infinitas bandas (o un número muy grande de ellas), sin embargo, limitaciones del mismo equipo hacen que el trabajo con imágenes hiperespectrales se restrinja al número de bandas que es capaz de obtener la cámara hiperespectral de trabajo.

Las cámaras hiperespectrales tienen principalmente dos formas de obtener la imagen, como se muestran en la figura 1:

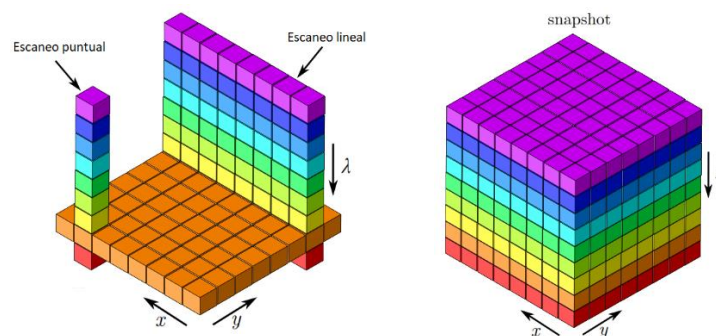


Figura 1. Distintas formas de obtener una imagen hiperespectral: puntual, lineal y por snapshot.

Snapshot. Capturar muchas imágenes bidimensionales (una para cada longitud de onda λ) y luego construir una *imagen tridimensional* (dos dimensiones espaciales y una más de longitud de onda) que se le suele llamar *datacube*.

Scanning. Captura una línea hiperespectral y a continuación la siguiente línea hasta completar toda la imagen.

Haciendo incidir radiación electromagnética simultáneamente en distintas longitudes de onda sobre un material, este reflejará la radiación incidente en diferentes intensidades para cada longitud de onda respectiva, de este modo es posible obtener un patrón característico del material.

1.1.1 Espectros de reflectancia e imágenes HSI

Las imágenes hiperespectrales son un conjunto de espectros de reflectancia pero considerando diferentes longitudes de onda. Estos espectros presentarán diferentes características que dependen de los elementos presentes en la imagen.

La figura 2, presenta el espectro de reflectancia de un pixel, registrado considerando longitudes de onda entre 0,2 nm y 2,7 nm.

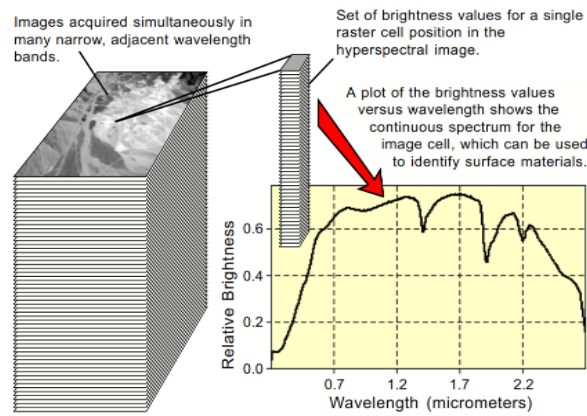


Figura 2. Patrón característico del material.

La figura 3 muestra los espectros de reflectancia de diferentes materiales. Es necesario indicar que pequeñas variaciones en la composición de un elemento da lugar a cambios sutiles que requieren de herramientas de análisis muy elaboradas.

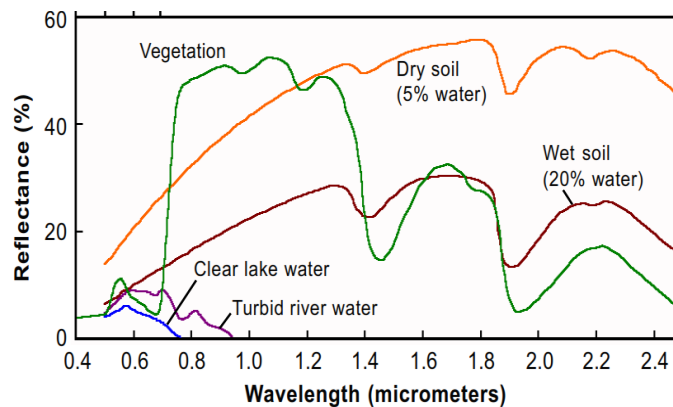


Figura 3. Materiales comunes en la superficie terrestre.

1.1.2 Espectros de reflectancia de minerales

En esta tesis se analizarán imágenes de suelos y de minerales. La figura 4 muestra los espectros de reflectancia de varios minerales. El rango de trabajo ideal como muestran estos espectros es hasta 2,5 nm pero sin embargo también se dispone información para rangos menores. El registro de una base de datos con estos espectros es muy importante cuando se presentan mezclados,

pues permitirán determinar la cantidad de cada elemento presente, por ejemplo, en una muestra en polvo.

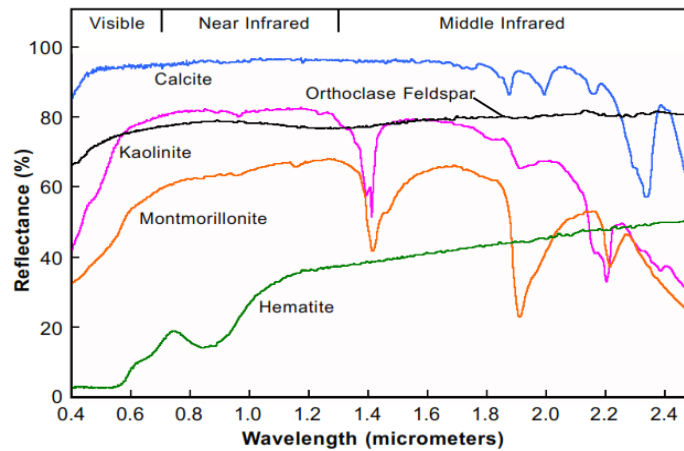


Figura 4. Patrón de algunos minerales comunes.

Obteniendo estos patrones y comparándolo con los patrones conocidos, es posible identificar de qué material se trata. Este procedimiento es conocido como espectrometría de imagen.

La espectrometría de imagen o teledetección hiperespectral explota el hecho de que todos los materiales reflejan, absorben y emiten energía electromagnética, con longitudes de onda específicas, en patrones distintivos relacionados con su composición molecular. Esta técnica proporciona información relacionada con las características superficiales de los materiales que pueden ser explotadas para permitir una detección automatizada de objetivos de interés, a partir de la construcción de un espectro de radiación. Por lo tanto, la explotación de datos de espectrómetros de imagen hace posible la identificación a distancia de materiales del suelo de interés sobre la base de sus firmas espectrales.

La teledetección hiperespectral es una técnica robusta que ha demostrado ser eficiente para aplicaciones en el estudio de la corteza terrestre. Teniendo presente que en nuestro país contamos con grandes recursos minerales y zonas que pueden ser explotadas de manera sostenible, usar la teledetección hiperespectral promete ser una de las mejores alternativas en la exploración de terrenos cuyos resultados serían de gran interés para la geología, topografía, minería, etc.

En el caso particular de la minería (creciente en nuestro país) implementar esta herramienta permitiría la exploración de terrenos de manera aérea, sin necesidad de intervenir directamente sobre el suelo preservando así la integridad de nuestra tierra, analizándola de manera sostenible. Puntos importantes tanto para el medio ambiente, flora y fauna del lugar, como para beneficios económicos en la minería, reduciendo así todo el equipo (instrumental y humano) necesario en la exploración de terrenos; adicionalmente se crearía una nueva rama de investigación y desarrollo en nuestro país.

Finalmente, el resultado de este trabajo de investigación será un código computacional que, teniendo como entradas imágenes de teledetección hiperespectral, realice el procesamiento de estas y, usando modelos físicos y librerías de materiales, sea capaz de determinar la composición del terreno respectivo.

1.2 Registro de imágenes hiperespectrales

1.2.1 Imágenes satelitales

La Agencia Espacial Europea (*European Space Agency - ESA*) tiene a su cargo la misión Sentinel-2 (Figura 5) como parte del programa Copérnico con la finalidad de observar y monitorear la tierra, bosques, desastres naturales.



Figura 5. Satélite de la misión Sentinel-2.

Sentinel-2 tiene 13 bandas multiespectrales repartidas desde el visible hasta el infrarrojo. Cubre los ángulos desde -56° hasta 84° de latitud y todos los ángulos de longitud. Tarda 5 días en volver a enfocar la misma zona terrestre. Además, tiene una resolución espacial de 10, 20 y 60 m.

Tabla 1. Bandas de Sentinel-2 y sus correspondientes longitudes de onda centrales.

Sentinel-2 Bands	Central Wavelength (μm)	Resolution (m)
Band 1 - Coastal aerosol	0.443	60
Band 2 - Blue	0.490	10
Band 3 - Green	0.560	10
Band 4 - Red	0.665	10
Band 5 - Vegetation Red Edge	0.705	20
Band 6 - Vegetation Red Edge	0.740	20
Band 7 - Vegetation Red Edge	0.783	20
Band 8 - NIR	0.842	10
Band 8A - Vegetation Red Edge	0.865	20
Band 9 - Water vapour	0.945	60
Band 10 - SWIR - Cirrus	1.375	60
Band 11 - SWIR	1.610	20
Band 12 - SWIR	2.190	20

La Agencia Espacial Europea (ESA) permite el acceso a su base de datos de imágenes multiespectrales, para lo cual los interesados deben inscribirse en su página web. La ESA proporciona luego un código de usuario y contraseña para acceder a esta base de datos, como se muestra en la Figura 6.

Figura 6. Imagen de la página de registro en la ESA para el acceso a su base de datos.

Dentro de este portal se puede descargar las imágenes multiespectrales, tales como las mostradas en las figuras 7, 8 y 9.



Figura 7. Fotografía “completa” alrededor de 41,449° 41,755°.

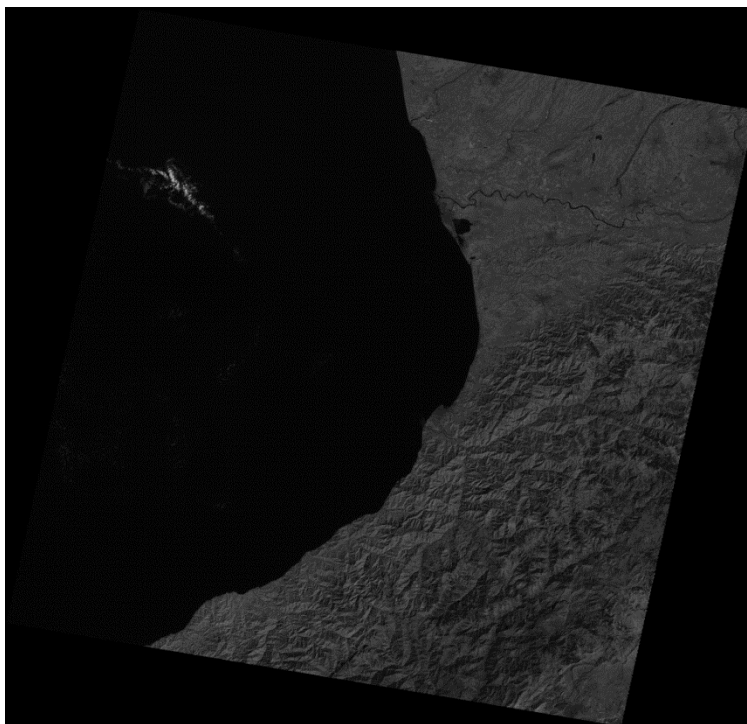


Figura 8. Fotografía con la banda 4.

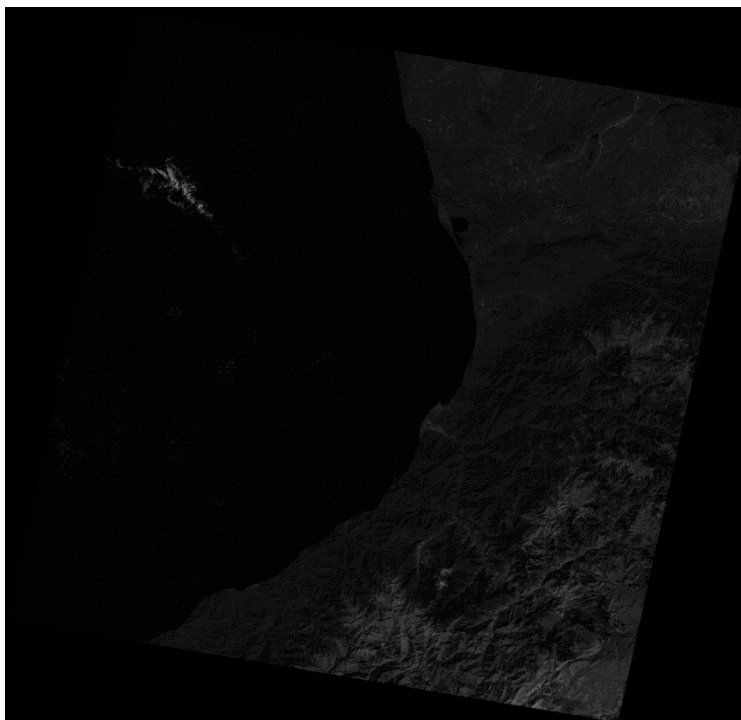


Figura 9. Fotografía con la banda 7.

Así se tienen las imágenes capturadas por las distintas bandas.

1.2.2 Cámaras Hiperespectrales.

Actualmente, existen en el mercado una gran variedad de cámaras hiperespectrales que son usadas de acuerdo a su rango de aplicación. Existen cámaras que registran imágenes en el espectro visible, otras en el espectro infrarrojo cercano, así como aquellas que registran imágenes en el infrarrojo.

Se proyecta que el mercado global de sistemas de imágenes hiperespectrales alcance los USD 30.000 millones en 2025 desde USD 12.400 millones en 2020, a una tasa compuesta anual del 19,3%. Factores como las crecientes aplicaciones industriales de imágenes hiperespectrales y el aumento de la financiación y las inversiones están impulsando el crecimiento del mercado global de sistemas de imágenes hiperespectrales. Sin embargo, se espera que los problemas de almacenamiento de datos y los altos costos asociados con el uso de imágenes hiperespectrales frenen el crecimiento del mercado.

Los actores destacados en este mercado incluyen Headwall Photonics, Inc. (EE.UU.), Specim, Spectral Imaging Ltd. (Finlandia), Corning Incorporated (EE.UU.), Norsk Elektro Optikk AS

(Noruega), Resonon Inc. (EE.UU.), Telops (Canadá), Surface Optics Corporation (EE.UU.), Applied Spectral Imaging Ltd. (EE.UU.), Raytheon Company (EE.UU.), ChemImage Corporation (EE.UU.), Cubert GmbH (Alemania), BaySpec, Inc. (EE.UU.). Los jugadores en este mercado se enfocaron en lanzamientos de productos, asociaciones, acuerdos, colaboraciones y adquisiciones para desarrollar y expandir su presencia en el mercado. Entre 2017 y 2020, los lanzamientos de productos formaron la estrategia de crecimiento clave adoptada por jugadores como Headwall Photonics (EE. UU.), Specim (Finlandia), Cubert GmbH (Alemania) y BaySpec, Inc. (EE. UU.).

Dentro de las principales compañías que trabajan con imágenes Hiperespectrales están:

Headwall Photonics (EE. UU.) Esta compañía es una de los principales actores del mercado de sistemas de imágenes hiperespectrales. Esta empresa fabrica y proporciona módulos ópticos e instrumentos de imagen totalmente integrados con rejillas de difracción holográficas originales. Así mismo, la compañía ofrece productos para inspección industrial, defensa y seguridad, teledetección y aplicaciones de investigación. GHGSat Satellite implementó el sensor de imágenes Micro-Hyperspec de Headwall en Claire, un satélite LEO para el monitoreo de emisiones de gases de efecto invernadero, en 2017.

Specim, Spectral Imaging (Finlandia) es un fabricante líder de instrumentos y sistemas de imágenes hiperespectrales. La compañía proporciona productos de imágenes hiperespectrales a varios clientes industriales OEM y organizaciones de investigación. Persigue la estrategia de lanzamiento de productos para fortalecer su portafolio de productos y aumentar su participación en el mercado. En los últimos tres años, SPECIM ha lanzado varios productos para ampliar su base de consumidores. SPECIM busca constantemente mejorar su red de distribución global.

Norsk Elektro Optikk AS (Noruega) es una empresa orientada a la investigación en el segmento de instrumentación electroóptica. La empresa fabrica productos electroópticos avanzados utilizando tecnología propia para el mercado internacional. También participa en una serie de proyectos internacionales de I+D en el campo electroóptico. La compañía se enfoca en I+D para fortalecer su cartera de productos y promover la adopción de sus tecnologías.

CAPÍTULO II

Componentes de imágenes RGB. Descomposición de imágenes Multiespectrales e Hiperespectrales en Series de Fourier

2.1 Imágenes RGB

Con la finalidad de mostrar cómo se pueden construir espectros, es decir, la intensidad de la radiación en términos de una magnitud característica, a partir de una imagen (la reproducción de la figura de un objeto por la combinación de los rayos de luz que proceden de él) y así presentar la forma en que se forman imágenes multiespectrales e hiperespectrales, se hizo una primera prueba con imágenes RGB, que consistió en:

- ✓ Tomar una fotografía digital y usando MATLAB descomponerla en 3 imágenes es decir en sus respectivas componentes RGB. Así simularemos tener imágenes “multiespectrales” de 3 bandas ($\lambda_1, \lambda_2, \lambda_3$).
- ✓ Construir el espectro (con 3 puntos) de cada uno de los pixeles de la imagen.
- ✓ Graficar algunos “espectros”.

Tomamos la imagen de la figura 10:



Figura 10. Fotografía de la Facultad de Ciencias en formato jpg.

MATLAB dispone de una serie de herramientas que permiten separar las componentes rojo (R), verde (G) y azul (B) presentes en la imagen. A continuación, se muestra el procedimiento que debe seguirse para lograr esta separación.

Usando el MATLAB cargamos la imagen en una variable llamada *FC* con el comando *imread*. Naturalmente el archivo JPG de la imagen debe estar en el directorio de trabajo.

```
>> FC=imread('fc.jpg');
```

Ahora tenemos en el espacio de trabajo de MATLAB una variable *FC* que contiene a la imagen. Con el comando *size* averiguamos las dimensiones de la variable *FC*.

```
>> size(FC)
228    301     3
```

Donde las dos primeras dimensiones corresponden a la resolución en pixeles de la imagen y la tercera dimensión corresponde a la descomposición RGB, esto es: La matriz cuya tercera componente es 1 corresponde a los rojos, la matriz con tercera componente igual a 2 corresponde a los verdes y cuando la tercera componente es 3 corresponde a los azules. La imagen a colores que vemos en la figura 10 corresponde a la contribución de las 3 imágenes RGB.

A continuación, se almacena el contenido de la variable *FC* en otra variable *R*:

```
>> R=FC;
```

En seguida “apagamos” las contribuciones verdes y azules así:

```
>> R(:, :, 2)=0;
>> R(:, :, 3)=0;
```

Ahora todas las terceras componentes iguales a 2 y a 3 son ceros, esto implica que solo tenemos valores diferentes de cero cuando la tercera componente es 1, eso es el color rojo.

Realizamos un procedimiento análogo para quedarnos con las contribuciones verde G y azul B.

Con el comando *imshow* mostramos la imagen deseada, de este modo mostramos las imágenes que hemos generado: R, G y B, figura 11.



Figura 11. Descomposición de fc.jpg en sus contribuciones RGB.

De esta manera se ha logrado disponer de una intensidad de radiación en tres componentes o “bandas” R, G y B, para cada pixel de la imagen.

Las imágenes multiespectrales están formadas por pocas bandas (usualmente menos de 10), mientras que las imágenes hiperespectrales están formadas por un mayor número de bandas.

2.2 Imágenes Multiespectrales

La imagen multiespectral está compuesta por “n” bandas. Luego tendremos “n” imágenes de un mismo objeto (en este caso de una misma zona) donde cada una de estas imágenes debió ser capturada en la longitud de onda propia de su banda respectiva. Si construimos un espectro a partir de la imagen multiespectral, el espectro resultante tendrá también “n” puntos, cada uno correspondiente a la intensidad en cada banda.

La figura 12 muestra imágenes multiespectrales de la misión Sentinel-2.

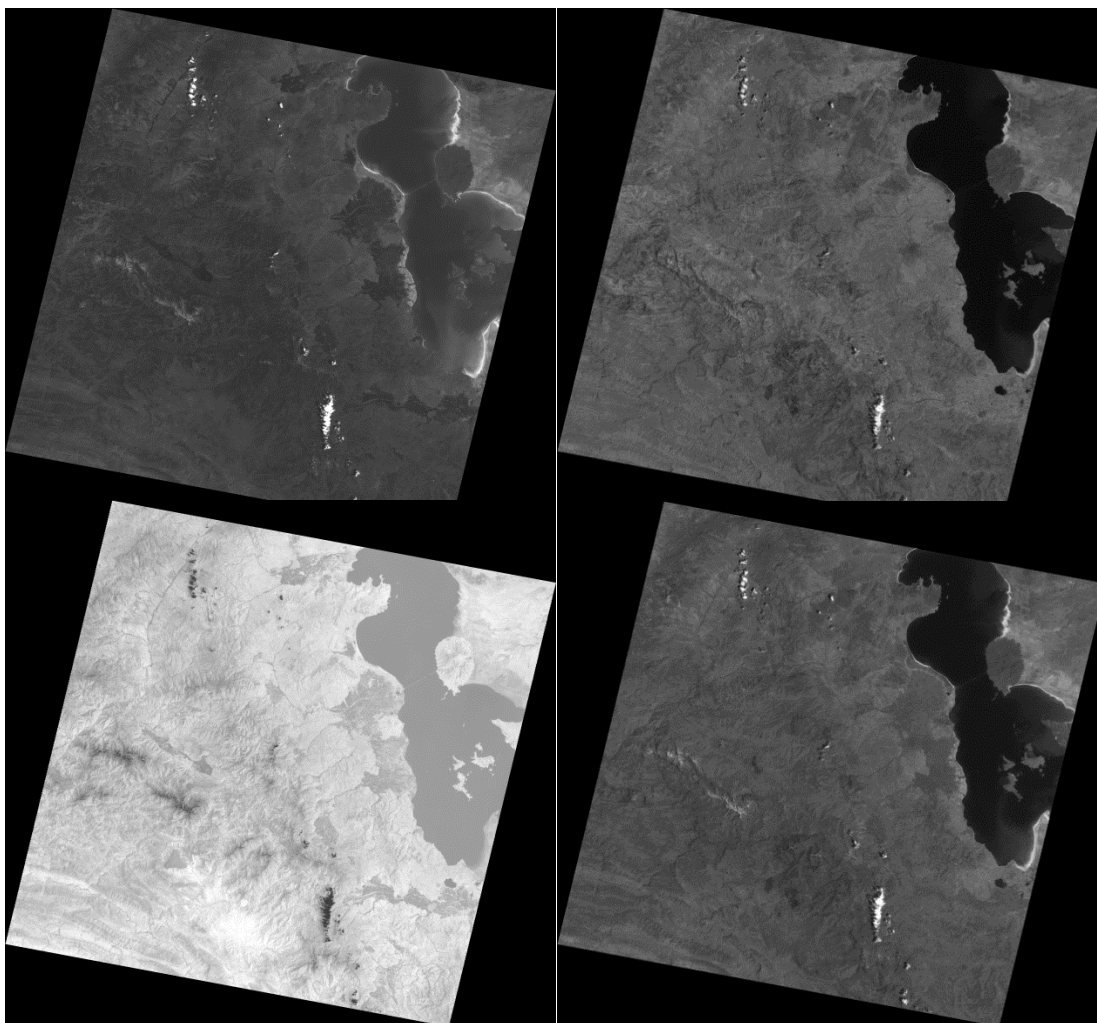


Figura 12. Algunas imágenes de la misión Sentinel-2. Banda B1, B4, B6 y B8 respectivamente.

Cada una de estas imágenes está compuesta por 52 131 040 píxeles, lo que podemos representar como una función bidimensional discreta $f_i(x, y)$ donde (x, y) indica la posición del píxel, el valor que toma la función es la intensidad en dicho píxel y el subíndice i etiqueta la banda ($1 \leq i \leq n$). Si nos concentramos en un píxel, es decir fijamos (x', y') , y analizamos cómo varía su intensidad en cada una de las bandas, tendremos un conjunto de “n” valores numéricos $f_1(x', y')$, $f_2(x', y')$, ..., $f_N(x', y')$ que representan las intensidades. Luego, podemos graficar la intensidad vs la banda para cada píxel de la imagen multiespectral.

2.3 Espectros a partir de las intensidades de radiación de los píxeles

Para conseguir la gráfica *Banda vs Intensidad*, después de importar las imágenes multiespectrales a MATLAB usando la función *imread* y acumular todas las imágenes

multiespectrales en una matriz tridimensional (la tercera dimensión corresponde al número de bandas), se escribió el siguiente código que, indicando las coordenadas del pixel construye la gráfica deseada:

```
function Espectros(x,y,B)
for k=1:length(B(1,1,:))
I(k) = B(x,y,k);
end
% L: Longitudes de onda en nm
L=[443 490 560 665 705 730 750 783 842];
plot(L,I)
grid on
xlabel('Longitud de onda nm')
ylabel('Intensidad 0-255')
end
```

Nota: En el eje horizontal, donde se indican las bandas, se escribió la longitud de onda correspondiente a cada banda en nm.

En MATLAB escribimos:

```
>> Espectros(3500,750,B)
>> Espectros(4500,1750,B)
>> Espectros(5580,350,B)
>> Espectros(6111,1350,B)
```

Obtenemos 4 espectros correspondientes al pixel (3500,750) , (4500,1750) , (5580,350) y (6111,1350) respectivamente.

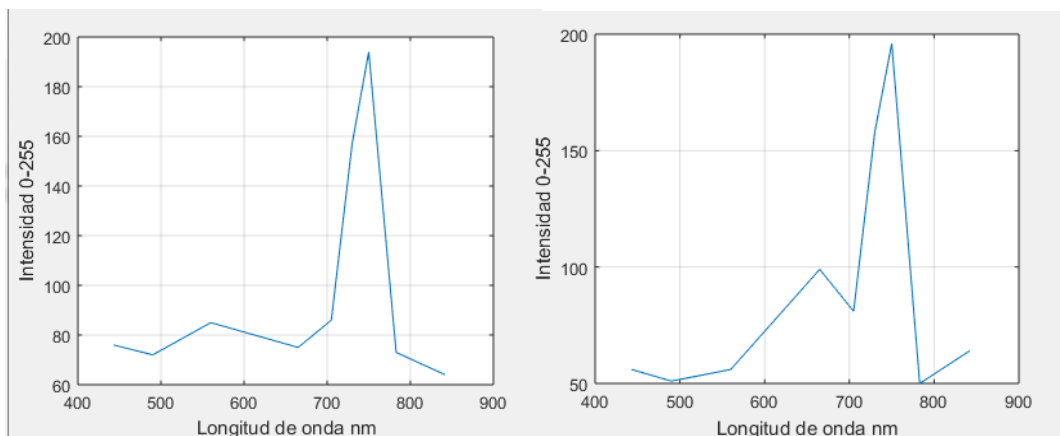


Figura 13. Espectros correspondientes a los pixeles (3500,750) y (4500,1750).

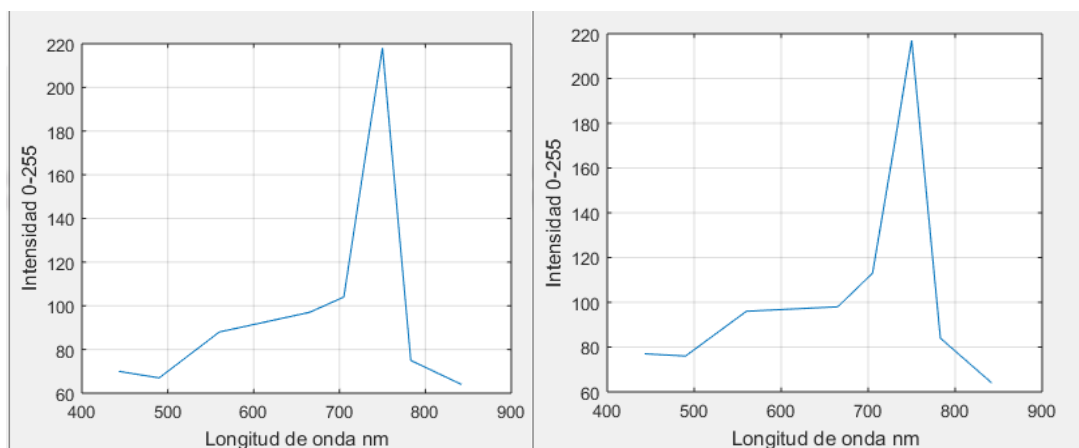


Figura 14. Espectros correspondientes a los pixeles (5580,350) y (6111,1350).

Con esto hemos conseguido obtener los espectros de cada pixel usando las bandas disponibles propias de la misión Sentinel-2. Esto se muestra en las figuras 13 y 14.

2.4 Descomposición espectral de una imagen usando la base de datos ASTER

Los espectros que hemos obtenido son producidos por la presencia de diversos materiales, ahora el desafío es determinar la manera de hallar qué materiales produjeron los espectros como los de las figuras 5 y 6 y en qué cantidades. Dicho de otra manera, necesitamos conocer qué espectros (de materiales conocidos) producen el espectro obtenido de cada pixel.

Como cada material se caracteriza por tener una firma espectral bien determinada, es necesario contar con una base de datos que almacene el espectro de cada material de tal forma que nos permita identificar el mismo en basados en su espectro.

Se encontró la base de datos en línea *ASTER (Spectral Library del Jet Propulsion Laboratory)* de la *NASA* (Administración Espacial Aeronáutica de Estados Unidos). Esta base de datos en línea contiene más de 2400 elementos en donde es posible visualizar el espectro de un material como un gráfico o como una tabla.

Por ejemplo, se muestra el espectro de reflectancia del Rutilo TiO_2 encontrado en la base de datos *ASTER Spectral Library*, figura 15:

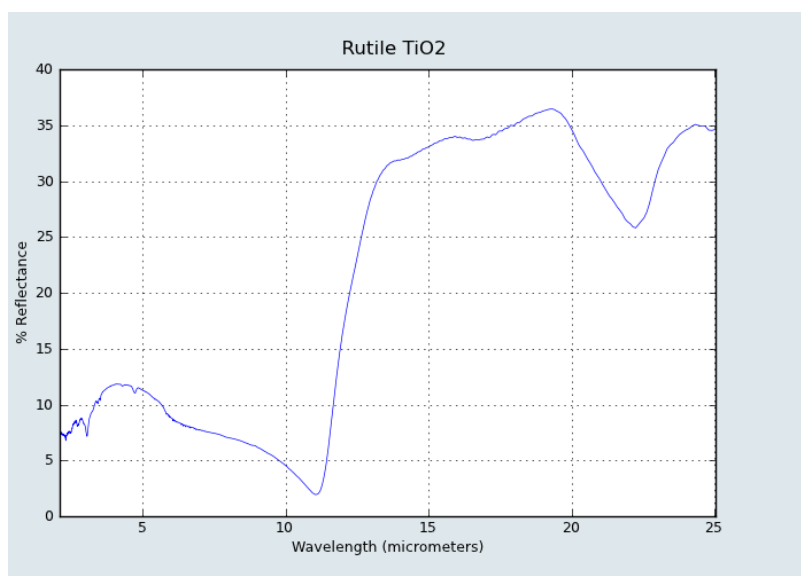


Figura 15. Espectro de reflectancia del Rutilo de la base de datos *ASTER*.

La información contenida en este espectro también puede ser visualizada en la figura 16.

```

X Units: Wavelength (micrometers)
Y Units: Reflectance (percent)
First X Value: 25.0442
Last X Value: 2.079
Number of X Values: 2287
Additional Information: rutile1a.txt

25.04420302    34.6205
24.9237955    34.537
24.80454022   34.557
24.68642074   34.9071
24.5694209    34.9152
24.45358465   34.9648
24.3387763    35.0666
24.22504094   34.9145
24.11236361   34.6923
24.00072962   34.5558
23.89012454   34.371
23.78053421   34.1525
    
```

Figura 16. Recorte de pantalla de los datos de reflectancia en forma de tabla.

Esta última forma tabulada nos es de mayor utilidad puesto que tenemos los valores numéricos de la reflectancia para cada longitud de onda considerada.

Recordemos que estos datos a pesar de mostrarse en forma de tabla, se encuentran en línea y esto los hace poco cómodos para trabajarlos rápidamente dentro de un algoritmo. Por tal motivo el contenido de estas tablas debe añadirse poco a poco a una matriz tridimensional (longitud de onda, reflectancia y material/elemento) de forma que el acceso a los datos sea más ágil.

El espectro de una muestra está formado por la contribución de varios espectros bien definidos, esto sugiere la idea de que un espectro sea escrito como una combinación lineal de otros espectros “básicos” (que no pueden ser escritos como suma de otros), esto sugiere adicionalmente que los espectros básicos podrían comportarse como una base vectorial.

Supongamos tenemos un espectro compuesto de “n” puntos (obtenido por ejemplo a partir de una cámara hiperespectral de “n” bandas), podemos representar este espectro como un vector n-dimensional donde cada componente del vector corresponde a la intensidad en su respectiva banda. Ahora nos encontramos dentro de un espacio vectorial n-dimensional y como tal, debe tener “n” vectores base, por tanto, un vector n-dimensional (espectro) puede escribirse como combinación lineal de “n” vectores base (espectros base).

Ahora, queremos escribir nuestros espectros n-dimensionales como combinación lineal de espectros base (de la base de datos ASTER) donde esta base también, desde luego, debe ser n-dimensional. Los más de 2400 espectros base proporcionados por ASTER tienen 2287 puntos que conforman cada uno de estos, dichos valores definen dos límites. Por una parte, los 2287 puntos de cada espectro implican que los vectores formados deben ser a lo más 2287-dimensionales, lo cual limita también el número de elementos de la base vectorial a 2287. Por otra parte, los 2400 elementos de la base de datos ASTER limita el número de materiales a encontrar en una zona, lo cual no es significativo puesto que es sumamente improbable que en una zona encontremos más de 2400 materiales/elementos.

En general, si N es el número de bandas de una Imagen multiespectral o hiperespectral, se cumplirá que $1 \leq n \leq 200$ que es un valor mucho menor a los 2400 elementos de la base de datos ASTER o a los 2287 puntos que componen estos espectros base. Si vamos a trabajar con imágenes multiespectrales o hiperespectrales cuyo número de puntos que componen un

espectro es “n”, debemos “traducir” toda la información de modo que podamos formar vectores n-dimensionales tanto con los espectros que obtengamos de las imágenes como los obtenidos de la base de datos ASTER. Los valores de las longitudes de onda (bandas) de las imágenes hiperespectrales no necesariamente están contenidos en el conjunto de longitudes de onda pertenecientes a los espectros de la base ASTER, por tanto, debemos interpolar datos de reflectancia de ASTER de modo que construyamos un conjunto de valores de reflectancia (para cada material de la base ASTER) que correspondan a las longitudes de onda de las imágenes que estemos considerando.

Para tal fin se escribió el siguiente código en MATLAB:

```
function DataBaseRed=ReducirBaseDatos(EspTotal, DataBase)
n = length(EspTotal(:,1)); p = length(DataBase(:,1,1)); r =
length(DataBase(1,1,:));
for k = 1:r
    l=1; % l->n
    for j = 1:p
        while ((EspTotal(l,1)-DataBase(j,1,k))<0) && (l<=n)
            a = DataBase(j-1,1,k);
            b = DataBase(j,1,k);
            DataBaseRed(l,1,k) = EspTotal(l,1);
            DataBaseRed(l,2,k) = DataBase(j-
            1,2,k)+(DataBase(j,2,k)-DataBase(j-1,2,k))/(b-
            a)*(EspTotal(l,1)-a);
            l=l+1;
        end
    end
end
end
```

Este código tiene como entradas el espectro total (obtenido de las imágenes hiperespectrales) y la base de datos ASTER y tiene como salida una base de datos “reducida” que contiene la misma cantidad de materiales contemplados en la base de datos original solo que esta nueva *base reducida* tiene “n” puntos por espectro.

Sea N el número de materiales/elementos de la base de datos ASTER y n el número de puntos que compone un espectro obtenido de imágenes hiperespectrales, entonces la dimensión de los vectores que representan estos espectros será n . Consideremos el vector \vec{E} que representa un espectro (de un pixel) obtenido de las imágenes y el conjunto de N vectores que representan cada uno de los materiales/elementos de la base de datos ASTER reducida $A = \{\vec{A}_1, \vec{A}_2, \dots, \vec{A}_N\} = \{\vec{A}_k\}$, entonces proponemos que:

$$\vec{E} = \sum_{j=1}^n \gamma_j \vec{A}_j \quad \dots (1)$$

Donde \vec{A}_j es un subconjunto de vectores linealmente independientes de n elementos de A y γ_j son las constantes de la combinación lineal.

Tengamos presente que los \vec{A}_j son de dimensión n y queremos que sean una base vectorial de un espacio n -dimensional pero tenemos un conjunto de N vectores \vec{A}_k ($n < N$) por tanto si elegimos los N vectores no serán linealmente independientes y no serán base vectorial. Es necesario elegir solo n vectores de un conjunto de N .

Por otra parte, es más cómodo trabajar con una base vectorial ortogonal que no necesariamente será la base \vec{A}_j . Usando el proceso de ortogonalización de Gram-Schmidt se construye una nueva base vectorial \vec{B}_j que si es ortogonal.

$\{\vec{A}_j\} \rightarrow \{\vec{B}_j\}$ usando Gram-Schmidt

$$\vec{E} = \sum_{j=1}^n \gamma_j \vec{A}_j = \sum_{j=1}^n \lambda_j \vec{B}_j \quad \dots (2)$$

Donde λ_j son las nuevas constantes de la combinación lineal con la nueva base ortogonal \vec{B}_j .

La base \vec{B}_j por ser ortogonal permite calcular fácilmente los coeficientes λ_j :

$$\vec{E} = \sum_{j=1}^n \lambda_j \vec{B}_j \quad \dots (3)$$

$$\vec{E} \cdot \vec{B}_i = \left(\sum_{j=1}^n \lambda_j \vec{B}_j \right) \cdot \vec{B}_i \quad \dots (4)$$

$$\vec{E} \cdot \vec{B}_l = \lambda_l \vec{B}_l \cdot \vec{B}_l \quad \dots (5)$$

$$\lambda_l = \frac{\vec{E} \cdot \vec{B}_l}{\vec{B}_l \cdot \vec{B}_l} \quad l = 1, 2, \dots, n \quad \dots (6)$$

De este modo calculamos todos los coeficientes λ_l de la combinación lineal con la base ortogonal \vec{B}_j . Sin embargo la base \vec{B}_j fue construida por facilidad de cálculo, lo que realmente nos interesa son los coeficientes γ_j de la combinación lineal de la base \vec{A}_j , la relación entre los coeficientes λ_l y γ_j será obtenida del proceso de Gram-Schmidt. De los coeficientes γ_j se espera obtener información no solo de los materiales/elementos que están presentes en la zona de análisis (pixel) sino también de las cantidades.

2.5 Algoritmos matriciales e imágenes multispectrales

Hasta ahora tenemos espectros contruidos a partir de imágenes multispectrales (espectros multispectrales), una base de datos llamada ASTER que nos brinda espectros de muchos elementos a manera de biblioteca (espectros base). Además se había propuesto tratar los espectros como vectores de \mathbb{R}^n , vectores 10-dimensionales y 2287-dimensionales, para los espectros multispectrales y espectros base respectivamente.

Sabemos que una muestra está compuesta por varios elementos, es decir, si conociéramos el espectro de la muestra, este debería ser resultado de la contribución de los espectros de los elementos que contiene. Siguiendo esta idea, queremos descomponer un espectro multispectral (vectores 10-dimensional) como una combinación de espectros base (vectores 2287-dimensionales). Para hacer esto, los vectores que representan ambos tipos de espectros deben tener la misma dimensión, entonces debemos reducir la dimensión de los espectros base.

2.5.1 Reducción de la dimensión de un espectro base

Con el fin de reducir la dimensión de un espectro base se elaboró el código `ReducirBaseDatos`.

Este código construye una base de datos reducida a partir de la base de datos ASTER, esta base reducida (espectros base 10-dimensionales) tiene la misma dimensión que los espectros multispectrales. Por ejemplo, en la base de datos se encuentra la alpita, una roca magmática,

de la cual tenemos su espectro (uno de los espectros base) y con el código ReducirBaseDatos construiremos su espectro reducido (uno de los espectros de la base reducida).

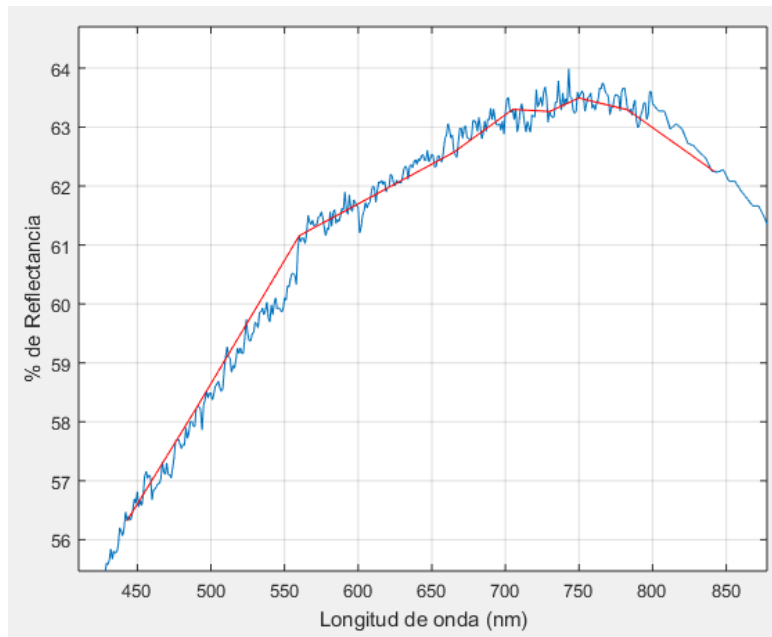


Figura 17. Azul: Espectro completo de la albita. Rojo: Espectro reducido de la albita.

Los espectros pertenecientes a la base reducida son como el espectro en color rojo mostrado en la figura 17. Se remarca que los vectores que representan estos espectros son de dimensión 10, es decir igual al número de canales de la cámara multiespectral de la misión Sentinel-2 (imágenes multiespectrales).

Tenemos la base reducida cuyos elementos (espectros base) son de la misma dimensión que los espectros multiespectrales, ahora estamos listos para descomponer un espectro multiespectral (vector de \mathbb{R}^{10}) en sus espectros base (10 vectores de \mathbb{R}^{10} pertenecientes a la base reducida).

Notación:

\bar{X} : Vector que representa a un espectro multiespectral

$\{\bar{A}_i\}$: Conjunto de n vectores base que representan a n elementos de la base reducida

$\{\bar{b}_j\}$: Base ortonormal construida a partir de la base $\{\bar{A}_i\}$

λ_i : Coeficientes de la combinación lineal usando la base $\{\bar{A}_i\}$

β_j : Coeficientes de la combinación lineal usando la base $\{\bar{b}_j\}$

Tenemos que $\bar{X} \in \mathbb{R}^n$ y que $\{\bar{A}_i\}$ es una base del mismo espacio, entonces podemos escribir:

$$\bar{X} = \sum_{i=1}^n \lambda_i \bar{A}_i \quad \dots (7)$$

Queremos calcular los coeficientes λ_i de la combinación lineal mostrada en (7).

Luego, usaremos el proceso de Gram-Schmidt para construir una base ortonormal a partir de la base $\{\bar{A}_i\}$. A la nueva base ortonormal la denotaremos como $\{\bar{b}_j\}$. Así tendremos que:

$$\bar{X} = \sum_{i=1}^n \lambda_i \bar{A}_i = \sum_{j=1}^n \beta_j \bar{b}_j \quad \dots (8)$$

Como $\{\bar{b}_j\}$ es una base ortonormal, resulta sencillo calcular los coeficientes β_j de la combinación lineal:

$$\bar{X} = \sum_{j=1}^n \beta_j \bar{b}_j \quad \dots (9)$$

$$\bar{X} \cdot \bar{b}_k = \left(\sum_{j=1}^n \beta_j \bar{b}_j \right) \cdot \bar{b}_k = \sum_{j=1}^n \beta_j (\bar{b}_j \cdot \bar{b}_k) \quad \dots (10)$$

Pero el producto escalar $\bar{b}_j \cdot \bar{b}_k$ resulta ser 1 cuando $j = k$, caso contrario $\bar{b}_j \cdot \bar{b}_k$ siempre es cero. Entonces:

$$\bar{b}_{j=k} \cdot \bar{b}_k = 1 \quad \rightarrow \quad \bar{X} \cdot \bar{b}_k = \beta_k \quad \dots (11)$$

Por otra parte, los vectores \bar{A}_i al ser también vectores de \mathbb{R}^n , se pueden escribir como combinación lineal de la base $\{\bar{b}_l\}$:

$$\bar{A}_i = \sum_{l=1}^n \gamma_{l,i} \bar{b}_l \quad \dots (12)$$

Del resultado (11) podemos calcular los coeficientes $\gamma_{i,l}$ como:

$$\bar{A}_i \cdot \bar{b}_k = \gamma_{k,i} \quad \dots (13)$$

Reemplazando (12) en (7)

$$\bar{X} = \sum_{i=1}^n \lambda_i \bar{A}_i = \sum_{i=1}^n \lambda_i \left(\sum_{l=1}^n \gamma_{l,i} \bar{b}_l \right) = \sum_{i=1}^n \sum_{l=1}^n \lambda_i \gamma_{l,i} \bar{b}_l \quad \dots (14)$$

Como los coeficientes i y l de la expresión (6) son independientes, podemos cambiar el orden de los sumandos:

$$\bar{X} = \sum_{i=1}^n \sum_{l=1}^n \lambda_i \gamma_{l,i} \bar{b}_l = \sum_{l=1}^n \sum_{i=1}^n \lambda_i \gamma_{l,i} \bar{b}_l \quad \dots (15)$$

$$\bar{X} = \sum_{l=1}^n \left(\sum_{i=1}^n \lambda_i \gamma_{l,i} \right) \bar{b}_l \quad \dots (16)$$

De (9) y (16) tenemos:

$$\bar{X} = \sum_{j=1}^n \beta_j \bar{b}_j = \sum_{j=1}^n \left(\sum_{i=1}^n \lambda_i \gamma_{j,i} \right) \bar{b}_j \quad \dots (17)$$

Entonces:

$$\beta_j = \sum_{i=1}^n \lambda_i \gamma_{j,i} \quad \dots (18)$$

Que, escrito de manera explícita seria:

$$\begin{aligned} \beta_1 &= \lambda_1 \gamma_{1,1} + \lambda_2 \gamma_{1,2} + \dots + \lambda_n \gamma_{1,n} \\ \beta_2 &= \lambda_1 \gamma_{2,1} + \lambda_2 \gamma_{2,2} + \dots + \lambda_n \gamma_{2,n} \\ &\vdots \\ \beta_n &= \lambda_1 \gamma_{n,1} + \lambda_2 \gamma_{n,2} + \dots + \lambda_n \gamma_{n,n} \end{aligned}$$

Con esto tenemos n ecuaciones con n incógnitas $\{\lambda_i\}$. Tengamos presente que la finalidad es calcular los coeficientes λ_i ya que todo lo demás ya fue calculado. Escribiéndolo de forma matricial:

$$\begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix} = \begin{pmatrix} \gamma_{1,1} & \gamma_{1,2} & \dots & \gamma_{1,n} \\ \gamma_{2,1} & \gamma_{2,2} & \dots & \gamma_{2,n} \\ \vdots & \vdots & \vdots & \vdots \\ \gamma_{n,1} & \gamma_{n,2} & \dots & \gamma_{n,n} \end{pmatrix} \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_n \end{pmatrix} = M \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_n \end{pmatrix} \quad \dots (19)$$

De (13) tenemos los coeficientes $\gamma_{k,i}$, por tanto la matriz M , y de (11) tenemos los coeficientes β_k . Así, calculamos los coeficientes λ_i :

$$\begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_n \end{pmatrix} = M^{-1} \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{pmatrix} \quad \dots (20)$$

Como ya conocemos los coeficientes λ_i , podemos reemplazarlos en (7) y habremos escrito el vector \bar{X} como combinación lineal de la base $\{\bar{A}_i\}$. Es decir, habremos descompuesto un espectro multiespectral en sus correspondientes espectros base (de la base reducida).

Para realizar este proceso de descomposición de un espectro multiespectral en espectros base, se escribió el siguiente código:

```
function L=CombLineal(X,BaseRed)
% BaseRed: Matriz cuadrada cuyas columnas son los vectores Ai
% (base reducida)
% X : Vector que representa al espectro multiespectral
A = BaseRed;
n = length(X);
% n : Dimension
[B,R] = qr(BaseRed,0);
% B : Matriz cuadrada cuyas columnas son los vectores bi (base
ortonormal)
% b(k) : Coeficientes beta de la combinacion lineal
% g(k,i) : Coeficientes gamma de cambio de base
for j=1:n
    b(j)=dot(B(:,j),X);
    for i=1:n
        g(j,i)=dot(B(:,j),A(:,i));
    end
end
b = b(:);
% M : Matriz de coeficientes gamma
M = g;
L = linsolve(M,b);
end
```

Este código calcula el vector columna L, que contiene a todos los coeficientes λ_i de la combinación lineal mostrada en (1). Nótese que para calcular la base ortonormal $\{\bar{b}_j\}$ a partir de la base multiespectral $\{\bar{A}_i\}$, usamos la descomposición QR de una matriz a través de la

función $qr(A)$ implementada en la biblioteca de MATLAB y usamos la modificación $qr(A,0)$ que brinda mayor estabilidad numérica a la descomposición.

Además para usar el código CombLineal debimos construir una matriz cuyas columnas son los vectores base $\{\bar{A}_i\}$ a partir de la base de datos reducida, para tal fin se escribió:

```
function BaseRed = BuildBaseRed(DataBaseRed)
for j=1:length(DataBaseRed(:,2,1))
    BaseRed(:,j) = DataBaseRed(:,2,j);
end
```

De este modo BaseRed es la matriz cuyas columnas contienen los vectores base $\{\bar{A}_i\}$. En adelante el trabajo se centrará en un espacio 10-dimensional.

2.6 Descomposición espectral usando Series de Fourier

La descomposición de un espectro en una base presenta el siguiente inconveniente:

Las imágenes multiespectrales con las que se cuentan tienen 9 canales, es decir $n = 9$, esto implica trabajar en el espacio \mathbb{R}^9 donde la dimensión nos restringe a tener solo 9 vectores base (9 espectros base). Esto limita la cantidad de elementos a identificar a 9, cuando se cuenta con una biblioteca de más de 2000 elementos base, es decir que se tiene que elegir varios grupos de 9 elementos de un conjunto de 2000 elementos para correr el código y verificar la presencia de esos 9 elementos o sino otros.

Esta limitación está asociada a la dimensión del espacio vectorial \mathbb{R}^n elegido para modelar los espectros, siguiendo con esta idea necesitaríamos un espacio vectorial de dimensión 2000 o más para que todo los elementos de la base de datos ASTER puedan usarse como vectores base del espacio vectorial elegido para trabajar.

Para resolver este problema se propone usar un espacio vectorial de dimensión infinita: el espacio de funciones. Proponemos modelar los espectros como funciones reales de una variable real. Usando esta idea se pretende elaborar un modelo que desconponga un espectro en sus espectros de elementos base, en otras palabras, escribir una función como combinación de otras

que correspondan a los espectros de elementos base. Siguiendo esta idea planteamos los siguientes pasos:

Escribir un espectro como una función.

Encontrar la manera de escribir una función como combinación de otras asociadas a la representación de los espectros de elementos base.

Escribir el procedimiento anterior como un código y hacer los ensayos de identificación correspondientes.

Validación.

Recordemos que los espectros de elementos base obtenidos de la biblioteca ASTER están dados por una colección de 826 pares ordenados (λ_i, r_i) donde $i = 1, 2, \dots, 826$. Los espectros obtenidos por la cámara hiperespectral están dados por 168 pares ordenados del mismo tipo. Naturalmente para que sea compatible trabajar con ambos tipos de espectros (los de 826 pares ordenados y los de 168), se homogenizarán con el código `ReducirBaseDatos` escrito anteriormente. En adelante decimos que un espectro está dado por 168 pares ordenados (λ_i, r_i) .

2.7 Serie de Fourier de un espectro

Dado un espectro o conjunto de puntos con coordenadas (Reflectancia vs Banda) se construye una función en forma de Serie de Fourier que mejor describa el comportamiento de estos puntos.

Se busca convertir esta colección de puntos numéricos en una función analítica continua de tal manera que la función represente a esta colección de puntos.

Para realizar esta tarea usaremos una expansión en Serie de Fourier:

$$f(t) = \sum_{k=-\infty}^{\infty} c_k e^{\frac{2\pi i k t}{T}} \quad \dots (21)$$

Para realizar esta representación se elaboró el siguiente código:

```
function [F_Y,F_Yi,C] = FourierExp1D_mod(Y,X,M,graf)
n = length(X); T = X(n)-X(1); w = 2*pi/T;
```

ANÁLISIS DE MUESTRAS DE SUELOS DE MINAS A PARTIR DE IMÁGENES HIPERESPECTRALES

```
% CALCULO DE LA RECTA A RESTAR
m = (Y(n)-Y(1))/(X(n)-X(1)); b = Y(1)-m*X(1);
% y = m*x + b
for j = 1:n
    Y(j) = Y(j) - (m*X(j)+b);
end
% CALCULO DE LOS COEFICIENTES DE FOURIER
for p = 1:(2*M+1)
    C_Sum_inf = 0; C_Sum_sup = 0;
    for j = 1:(n-1)
        dX = X(j+1)-X(j);
        C_Sum_inf = C_Sum_inf + Y(j)*exp(-1i*(p-M-1)*w*X(j))*dX;
        C_Sum_sup = C_Sum_sup + Y(j+1)*exp(-1i*(p-M-1)*w*X(j+1))*dX;
    end
    C(p) = (1/T)*(C_Sum_inf+C_Sum_sup)/2; % Coeficiente de
    Fourier
end
% CALCULO DE Y DE FOURIER: F_Y
for j = 1:n
    Y_Fourier_Sum = 0;
    for p = 1:(2*M+1)
        Y_Fourier_Sum = Y_Fourier_Sum + C(p)*exp(1i*(p-M-1)*w*X(j));
    end
    F_Y(j) = real(Y_Fourier_Sum);
    F_Yi(j) = imag(Y_Fourier_Sum);
end
% CALCULO DE Y DE FOURIER (para grafica): F_Y
n_graf = 3000;
t = linspace(X(1),X(n),n_graf);
for j = 1:n_graf
    Y_Fourier_Sum = 0;
```

```

    for p = 1:(2*M+1)
        Y_Fourier_Sum = Y_Fourier_Sum + C(p)*exp(1i*(p-M-
            1)*w*t(j));
    end
    F_Y_graf(j) = real(Y_Fourier_Sum);
    F_Yi_graf(j) = imag(Y_Fourier_Sum);
end
% CALCULO DEL COEFICIENTE DE CORRELACION NO LINEAL
% Ymean: Promedio de los valores de Y
Ymean = 0;
for j = 1:n
    Ymean = Ymean + Y(j);
end
Ymean = Ymean/n;
% SSE: Suma de los errores al cuadrado
% SST: Suma de las desviaciones de la media al cuadrado
SSE = 0; SST = 0;
for j = 1:n
    SSE = SSE + (Y(j)-F_Y(j))^2;
    SST = SST + (Y(j)-Ymean)^2;
end
% r: Coeficiente de correlacion no lineal
% r = sqrt(1-SSE/SST)
r2 = 1-SSE/SST
% SUMANDO LA RECTA
for j = 1:n
    Y(j) = Y(j) + (m*X(j)+b);
end
for j = 1:n_graf
    F_Y_graf(j) = F_Y_graf(j) + m*t(j)+b;
    F_Yi_graf(j) = F_Yi_graf(j) + m*t(j)+b;
end
% GRAFICA DE Y y DE F_Y
switch graf

```

```

case 1 % Grafica Y y F_Y
    hold on
    title('Azul: Numérico.   Rojo: Fourier.')
    xlabel('Componente "X"')
    ylabel('Componente "Y"')
    grid on
    plot(X,Y)           % Azul: Numerico
    plot(t,F_Y_graf,'r') % Rojo: Fourier
case 2 % Grafica F_Y
    hold on
    title('Rojo: Fourier')
    xlabel('Componente "X"')
    ylabel('Componente "Y"')
    plot(t,F_Y_graf,'r') % Rojo: Fourier
case 3 % Grafica F_Y - Y
    hold on
    title('F_Y - Y')
    xlabel('Componente "X"')
    ylabel('Componente "Y"')
    plot(X,F_Y - Y)
case 4 % Grafica F_Yi
    hold on
    title('F_Yi (complejo)')
    xlabel('Componente "X"')
    ylabel('Componente "Y"')
    plot(X,F_Yi_graf)
end
end

```

Este código encuentra la Serie de Fourier de la forma exponencial compleja de la colección de puntos, la cantidad de términos de la serie la puede definir el usuario. Además, el código calcula el coeficiente de correlación que evalúa que tan buena es la aproximación de la serie.

2.8 Descomposición de un espectro representado por una Serie de Fourier en componentes base

Ahora estamos en el espacio de las funciones. Las funciones que tenemos están escritas con Series de Fourier, entonces conviene aprovechar este hecho usando las propiedades de la base

$$\left\{ e^{\frac{2\pi i k t}{T}} = e_k \right\}.$$

La base $\{e_k\}$ es ortonormal, es decir:

$$\langle e_k, e_l \rangle = 0 \quad , \quad \langle e_k, e_k \rangle = 1$$

El producto interno de funciones resulta ser 1 si es que se multiplica la función e_k por sí misma, en cualquier otro caso el resultado será cero.

Notación:

f : Función que representa un espectro obtenido con la cámara hiperespectral

$\{^m g\} = \{^1 g, ^2 g, \dots, ^N g\}$: Conjunto de N funciones que representan a cada uno de los espectros de elementos base de la biblioteca ASTER.

Tenemos que:

$$f = \sum_{l=-\infty}^{\infty} c_l e_l \quad , \quad ^m g = \sum_{l=-\infty}^{\infty} {}^m c_l e_l \quad \dots (22)$$

Donde c_l son los coeficientes de Fourier de la función f y ${}^m c_l$ son los coeficientes de Fourier de la función ${}^m g$, donde $m = 1, 2, \dots, N$.

Estamos buscando expresar f como combinación de las funciones $\{^m g\}$. Entonces:

$$f = \sum_{m=1}^N \alpha_m {}^m g \quad \dots (23)$$

La tarea es calcular los coeficientes α_m , con esto habremos logrado escribir f como combinación de $\{^m g\}$.

De (22), podemos escribir:

$$\sum_{l=-\infty}^{\infty} c_l e_l = \sum_{m=1}^N \alpha_m \sum_{l=-\infty}^{\infty} {}^m c_l e_l \quad \dots (24)$$

$$\sum_{l=-\infty}^{\infty} c_l e_l = \sum_{m=1}^N \sum_{l=-\infty}^{\infty} \alpha_m {}^m c_l e_l \quad \dots (25)$$

Usando la propiedad de la ortonormalidad de las funciones $\{e_k\}$, multiplicamos escalarmente (25) por e_k :

$$\left\langle \sum_{l=-\infty}^{\infty} c_l e_l, e_k \right\rangle = \left\langle \sum_{m=1}^N \sum_{l=-\infty}^{\infty} \alpha_m {}^m c_l e_l, e_k \right\rangle \quad \dots (26)$$

$$\sum_{l=-\infty}^{\infty} c_l \langle e_l, e_k \rangle = \sum_{m=1}^N \sum_{l=-\infty}^{\infty} \alpha_m {}^m c_l \langle e_l, e_k \rangle \quad \dots (27)$$

$$\sum_{l=-\infty}^{\infty} c_l \delta_{l,k} = \sum_{m=1}^N \sum_{l=-\infty}^{\infty} \alpha_m {}^m c_l \delta_{l,k} \quad \dots (28)$$

En (28) tenemos en ambos lados, sumatorias infinitas pero dentro de las mismas hay un delta de Kronecker, esto se reduce a:

$$c_k = \sum_{m=1}^N \alpha_m {}^m c_k \quad , \quad k = -\infty, \dots, -2, -1, 0, 1, 2, \dots, \infty \quad \dots (29)$$

Como son conocidos los coeficientes c_k y ${}^m c_k$ de las Series de Fourier de f y de la familia de funciones $\{{}^m g\}$ respectivamente, para calcular los coeficientes α_m basta con elegir al menos N valores de k y se forma sistema lineal de N ecuaciones con N incógnitas, por ejemplo elijamos $k = 1, 2, \dots, N$ (puede ser cualquier otro conjunto de N valores para k)

$$\begin{aligned} c_k &= \alpha_1 {}^1 c_k + \alpha_2 {}^2 c_k + \dots + \alpha_N {}^N c_k & k = 1, 2, \dots, N \\ c_1 &= \alpha_1 {}^1 c_1 + \alpha_2 {}^2 c_1 + \dots + \alpha_N {}^N c_1 \\ c_2 &= \alpha_1 {}^1 c_2 + \alpha_2 {}^2 c_2 + \dots + \alpha_N {}^N c_2 \\ &\vdots \\ c_N &= \alpha_1 {}^1 c_N + \alpha_2 {}^2 c_N + \dots + \alpha_N {}^N c_N \end{aligned}$$

Este sistema de ecuaciones lo podemos escribir de forma matricial:

$$\begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_N \end{pmatrix} = \begin{pmatrix} {}^1c_1 & {}^2c_1 & \dots & {}^Nc_1 \\ {}^1c_2 & {}^2c_2 & \dots & {}^Nc_2 \\ \vdots & \vdots & \ddots & \vdots \\ {}^1c_N & {}^2c_N & \dots & {}^Nc_N \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_N \end{pmatrix} \quad \dots (30)$$

De (30), llamemos M a la matriz que contiene a los coeficientes de las Series de Fourier de la familia de funciones $\{^m g\}$:

$$M = \begin{pmatrix} {}^1c_1 & {}^2c_1 & \dots & {}^Nc_1 \\ {}^1c_2 & {}^2c_2 & \dots & {}^Nc_2 \\ \vdots & \vdots & \ddots & \vdots \\ {}^1c_N & {}^2c_N & \dots & {}^Nc_N \end{pmatrix} \quad \dots (31)$$

Entonces para resolver (30):

$$\begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_N \end{pmatrix} = M^{-1} \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_N \end{pmatrix} \quad \dots (32)$$

La solución para los coeficientes α_m dada en (32) fue construida a partir de la elección de $k = 1, 2, \dots, N$ pero puede ser cualquier otro conjunto de N valores para k , por ejemplo $k = 2, 3, \dots, N + 1$, $k = -5, -4, -3, \dots, N - 6$, etc.

NO tiene sentido físico coeficientes α_m negativos pues de existir, estaríamos hablando de contribuciones “negativas” de espectros (sería como poner un espectro *de cabeza*) lo cual no es correcto.

Otra observación sobre el modelo anterior es la dificultad que la igualdad $f = \sum_{m=1}^N \alpha_m {}^m g$ se cumpla, en general no tiene porqué cumplirse.

Se buscará escribir el espectro modelado por la función f y se tratará de descomponerlo en base a espectros puros representados por $\{^m g\}$ de la siguiente manera:

$$f = \sum_{m=1}^N \alpha_m {}^m g + \varepsilon \quad \text{con } \alpha_m \geq 0 \quad \dots (33)$$

Donde

N : Número de elementos puros

ε : Residuo que queremos que sea mínimo

α_m : Coeficientes de la combinación lineal que se le impondrá la condición que sean mayores o iguales que cero.

Ahora, queremos que ε tienda a cero

$$\varepsilon = f - \sum_{m=1}^N \alpha_m^m g \quad \text{con } \alpha_m \geq 0 \quad \dots (34)$$

haciendo álgebra

$$\varepsilon = f - \sum_{m=1}^N \alpha_m^m g \quad \dots (35)$$

$$\varepsilon = \sum_{l=-\infty}^{\infty} c_l e_l - \sum_{m=1}^N \alpha_m \sum_{l=-\infty}^{\infty} c_l e_l \quad \dots (36)$$

$$\varepsilon = \sum_{l=-\infty}^{\infty} c_l e_l - \sum_{m=1}^N \sum_{l=-\infty}^{\infty} \alpha_m^m c_l e_l \quad \dots (37)$$

Como los índices m y l son independientes, podemos cambiar el orden de las sumatorias

$$\varepsilon = \sum_{l=-\infty}^{\infty} c_l e_l - \sum_{l=-\infty}^{\infty} \sum_{m=1}^N \alpha_m^m c_l e_l \quad \dots (38)$$

$$\varepsilon = \sum_{l=-\infty}^{\infty} \left(c_l e_l - \sum_{m=1}^N \alpha_m^m c_l e_l \right) \quad \dots (39)$$

Aprovechando nuevamente la ortogonalidad de las funciones $\{e_k\}$, multiplicamos (39) por e_k y tenemos:

$$\langle \varepsilon, e_k \rangle = \left\langle \sum_{l=-\infty}^{\infty} \left(c_l e_l - \sum_{m=1}^N \alpha_m^m c_l e_l \right), e_k \right\rangle \quad \dots (40)$$

$$\langle \varepsilon, e_k \rangle = \sum_{l=-\infty}^{\infty} \left(c_l \langle e_l, e_k \rangle - \sum_{m=1}^N \alpha_m^m c_l \langle e_l, e_k \rangle \right) \quad \dots (41)$$

Como $\langle e_l, e_k \rangle = \delta_{l,k}$

$$\langle \varepsilon, e_k \rangle = \sum_{l=-\infty}^{\infty} \left(c_l \delta_{l,k} - \sum_{m=1}^N \alpha_m {}^m c_l \delta_{l,k} \right) \quad \dots (42)$$

En las sumatorias infinitas, el índice l toma todos los valores enteros, por tanto en algún momento será $l = k$, y las deltas de Kronecker $\delta_{l,k}$ serán $\delta_{k,k} = 1$ reduciendo la expresión anterior a la siguiente:

$$\langle \varepsilon, e_k \rangle = c_k - \sum_{m=1}^N \alpha_m {}^m c_k \quad \text{con} \quad \alpha_m \geq 0 \quad \text{y} \quad k \in \mathbb{Z} \quad \dots (43)$$

Recordemos que los c_k y ${}^m c_k$ son conocidos (ya fueron calculados) pues son los coeficientes de las series de Fourier de f y los ${}^m g$ (espectro obtenido por la cámara y espectros base). Como ya se mencionó, estas series de Fourier tienen un número finito de términos pues son calculados numéricamente y cada coeficiente está etiquetado con el índice k de (43).

Retomando, queremos hacer cero la expresión (34), entonces

Si $\varepsilon \rightarrow 0 \Leftrightarrow \langle \varepsilon, e_k \rangle \rightarrow 0$, luego es equivalente a *minimizar* la expresión (43).

Usualmente cuando se pretende minimizar alguna función con ciertas condiciones (de igualdad) se suele pensar en el Método de los Multiplicadores de Lagrange, sin embargo, la expresión que queremos minimizar (16) tiene condiciones de desigualdad, lo cual no hace posible utilizar los Multiplicadores de Lagrange en este caso.

Para resolver el problema de minimización de una función con condiciones de desigualdad utilizaremos las Condiciones de Karush-Kuhn-Tucker (condiciones KKT).

Siendo $\bar{\alpha} = (\alpha_1, \alpha_2, \dots, \alpha_N)$, definimos la función

$$h(\bar{\alpha}) = \sum_k \langle \varepsilon, e_k \rangle^2 = \sum_k \left(c_k - \sum_{m=1}^N \alpha_m {}^m c_k \right)^2 \quad \dots (44)$$

NOTA: Tomamos la suma de los cuadrados pues los $\langle \varepsilon, e_k \rangle$ pueden ser positivos o negativos, además de englobar la información de todos los términos de las series de Fourier de f y ${}^m g$.

Recordemos que los coeficientes c_k y ${}^m c_k$ son los correspondientes a las series de Fourier de f y ${}^m g$ respectivamente, donde el índice k tiene que ver con la cantidad de términos de la serie,

es decir, la cantidad de términos el usuario la introduce manualmente dependiendo de qué tantos términos se desee en las series y de qué tanto poder computacional se tenga.

Pretendemos minimizar la expresión (44) con las condiciones $\alpha_m \geq 0$, usando las condiciones KKT, tenemos:

$$\alpha_m \geq 0 \quad \rightarrow \quad -\alpha_m \leq 0 \quad \dots (45)$$

Como la cantidad $-\alpha_m$ es menor o igual que cero, podemos sumarle una determinada cantidad positiva s_m^2 tal que la suma sea cero.

$$-\alpha_m + s_m^2 = 0 \quad \dots (46)$$

Ahora construimos la función $H(\bar{\alpha}, \bar{\lambda}, \bar{s})$

$$H(\bar{\alpha}, \bar{\lambda}, \bar{s}) = h(\bar{\alpha}) + \sum_{m=1}^N \lambda_m (-\alpha_m + s_m^2) \quad \dots (47)$$

Entonces, los puntos que minimizan $h(\bar{\alpha})$ con las condiciones $\alpha_m \geq 0$ son los puntos críticos de $H(\bar{\alpha}, \bar{\lambda}, \bar{s})$, luego:

Derivando H respecto de α_j e igualando a cero

$$\frac{\partial H}{\partial \alpha_j} = \frac{\partial h}{\partial \alpha_j} + \sum_{m=1}^N \lambda_m \frac{\partial}{\partial \alpha_j} (-\alpha_m + s_m^2) \quad \dots (48)$$

$$\frac{\partial H}{\partial \alpha_j} = \frac{\partial}{\partial \alpha_j} \left(\sum_k \left(c_k - \sum_{m=1}^N \alpha_m^m c_k \right)^2 \right) + \sum_{m=1}^N \lambda_m (-\delta_{j,m}) \quad \dots (49)$$

$$\frac{\partial H}{\partial \alpha_j} = \sum_k 2(-^j c_k) \left(c_k - \sum_{m=1}^N \alpha_m^m c_k \right) + (-\lambda_j) \quad \dots (50)$$

$$\frac{\partial H}{\partial \alpha_j} = 2 \sum_k \left(\sum_{m=1}^N \alpha_m^j c_k^m c_k - ^j c_k c_k \right) - \lambda_j \quad \dots (51)$$

$$\frac{\partial H}{\partial \alpha_j} = 2 \sum_k \sum_{m=1}^N \alpha_m^j c_k^m c_k - 2 \sum_k ^j c_k c_k - \lambda_j \quad \dots (52)$$

Como los índices k y m son independientes, podemos cambiar el orden de las sumas.

$$\frac{\partial H}{\partial \alpha_j} = 2 \sum_{m=1}^N \sum_k \alpha_m^j c_k^m c_k - 2 \sum_k^j c_k c_k - \lambda_j \quad \dots (53)$$

$$\frac{\partial H}{\partial \alpha_j} = 2 \sum_{m=1}^N \alpha_m \sum_k^j c_k^m c_k - 2 \sum_k^j c_k c_k - \lambda_j = 0 \quad \dots (54)$$

Derivando H respecto de λ_j e igualando a cero

$$\frac{\partial H}{\partial \lambda_j} = \frac{\partial h}{\partial \lambda_j} + \sum_{m=1}^N \frac{\partial}{\partial \lambda_j} \lambda_m (-\alpha_m + s_m^2) \quad \dots (55)$$

$$\frac{\partial H}{\partial \lambda_j} = \frac{\partial}{\partial \lambda_j} \left(\sum_k \left(c_k - \sum_{m=1}^N \alpha_m^m c_k \right)^2 \right) + \sum_{m=1}^N \delta_{j,m} (-\alpha_m + s_m^2) \quad \dots (56)$$

$$\frac{\partial H}{\partial \lambda_j} = -\alpha_j + s_j^2 \quad \dots (57)$$

$$-\alpha_j + s_j^2 = 0 \quad \dots (58)$$

Derivando H respecto de s_j

$$\frac{\partial H}{\partial s_j} = \frac{\partial h}{\partial s_j} + \sum_{m=1}^N \lambda_m \frac{\partial}{\partial s_j} (-\alpha_m + s_m^2) \quad \dots (59)$$

$$\frac{\partial H}{\partial s_j} = \frac{\partial}{\partial s_j} \left(\sum_k \left(c_k - \sum_{m=1}^N \alpha_m^m c_k \right)^2 \right) + \sum_{m=1}^N \lambda_m (2s_m \delta_{j,m}) \quad \dots (60)$$

$$\frac{\partial H}{\partial s_j} = 2\lambda_j s_j \quad \dots (61)$$

$$2\lambda_j s_j = 0 \rightarrow \lambda_j s_j = 0 \quad \dots (62)$$

De (62), se deduce que:

$$\lambda_j = 0 \vee s_j = 0$$

pero de (58) si $s_j = 0$, entonces $\alpha_j = 0$, luego la condición anterior se puede escribir como:

$$\lambda_j = 0 \vee \alpha_j = 0 \leftrightarrow \alpha_j \lambda_j = 0 \quad \dots (63)$$

Finalmente, de (54) y (63) tenemos las ecuaciones:

$$\begin{cases} 2 \sum_{m=1}^N \alpha_m \sum_k^j c_k^m c_k - 2 \sum_k^j c_k c_k - \lambda_j = 0 \\ \alpha_j \lambda_j = 0 \end{cases} \quad \text{donde } j = 1, 2, \dots, N \quad \dots (64)$$

Es decir:

$$\alpha_j \left(\sum_{m=1}^N \alpha_m \sum_k^j c_k^m c_k - \sum_k^j c_k c_k \right) = 0 \quad \text{donde } j = 1, 2, \dots, N \quad \dots (65)$$

Un sistema no lineal de N ecuaciones con N incógnitas, que al resolverlo, encontraremos los coeficientes α_j que minimizan

$$\sum_k \langle \varepsilon, e_k \rangle^2 = \sum_k \left(c_k - \sum_{m=1}^N \alpha_m^m c_k \right)^2 \quad \text{con } \alpha_m \geq 0 \quad \dots (66)$$

es decir, hacen que $\varepsilon = f - \sum_{m=1}^N \alpha_m^m g$ sea mínimo con $\alpha_m \geq 0$

CAPÍTULO III

Tratamiento estadístico de imágenes hiperespectrales

Se presentan los conceptos básicos que se usan en el proceso de análisis de una imagen hiperespectral como el análisis de componentes principales (PCA), el método de la mínima fracción de ruido (MNF), el Simplex growing algorithm (SGA), así como el procedimiento seguido para hacer un desmezclado o obtención de las abundancias de minerales presentes en una imagen.

3.1 Datos obtenidos en una imagen hiperespectral

En una imagen hiperespectral se tienen reflectancias para diferentes longitudes de onda. A estas reflectancias se les denomina variables aleatorias por lo que si se tienen p longitudes de onda se tendrán p variables aleatorias o componentes (X_1, X_2, \dots, X_p). Estas componentes se pueden organizar en un vector aleatorio $X = [X_1, X_2, \dots, X_p]^T$. La forma en que se comportan estas variables se puede describir en términos del valor esperado $\mu = E(X)$ y la matriz de varianza-covarianza $\Sigma = \text{Var}(X)$ [16].

Sea $E(X_i)$ es el valor esperado o valor medio de la variable X_i , definimos:

El vector de las poblaciones medias μ como

$$\mu = E(X) = [E(X_1), E(X_2), \dots, E(X_p)]^T \quad \dots (67)$$

La varianza Σ o $\text{Var}(X)$ como

$$\Sigma = \text{Var}(X) = E[(X - \mu)(X - \mu)^T] \quad \dots (68)$$

La covarianza $\text{Cov}(X_1, X_2)$ como

$$\text{Cov}(X_1, X_2) = E[(X_1 - E(X_1))(X_2 - E(X_2))^T] \quad \dots (69)$$

3.2 Análisis de componentes principales (PCA)

Análisis de componentes principales o Principal Component Analysis (PCA por sus siglas en inglés), es un método estadístico que permite simplificar la complejidad de espacios muestrales con muchas dimensiones a la vez que conserva su información. En nuestro caso las distintas longitudes de onda forman el espacio multidimensional, así 168 longitudes de onda o variables X_i formarán un espacio de dimensión $p=168$. Luego si tenemos n pixeles, cada uno de ellos tendrá p variables (X_1, X_2, \dots, X_p), y decimos que el espacio muestral tiene p dimensiones. Asimismo, como cada imagen de n pixeles corresponde a una longitud de onda entonces cada una de las p variables tendrá una población de n datos.

PCA permite encontrar una combinación lineal de las variables originales que nos permite hallar una dimensión z ($z < p$) que da aproximadamente la misma información en el sentido de variabilidad que las p variables originales.

En la figura 18 [16] se puede observar intuitivamente la dirección de máxima varianza P_1 . Para dos variables X_1, X_2 .

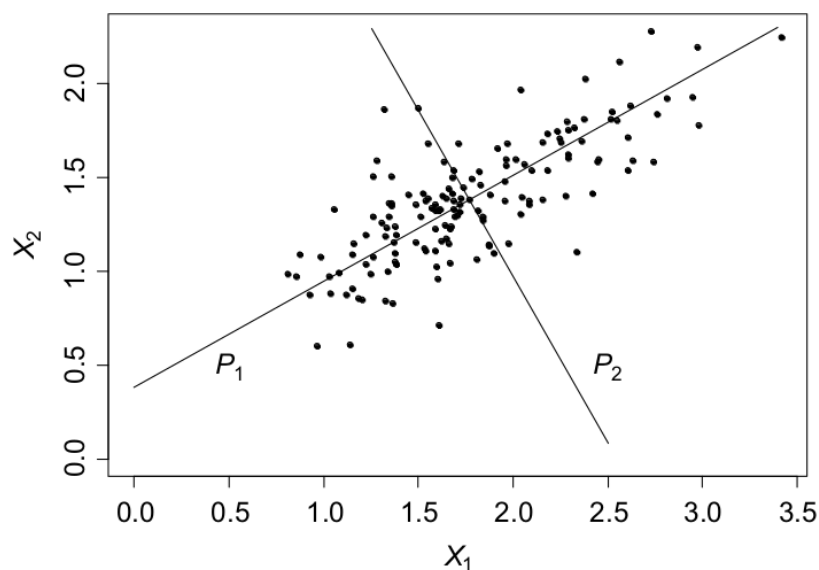


Figura 18. Dirección de máxima variabilidad (P_1) y la dirección ortogonal (P_2) de mínima variabilidad en el espacio X_1 - X_2 [16].

Consideremos un vector p -dimensional $X=[X_1, X_2, \dots, X_p]^T$ y una combinación lineal de sus componentes,

$$Y_1 = a_1^T X = a_{11}X_1 + a_{21}X_2 + \dots + a_{p1}X_p \quad \dots (70)$$

La primera componente principal (PC) se define como la combinación lineal $Y_1 = a_1^T X$ que maximiza la Varianza $\text{Var}(Y_1)$ sujeta a la restricción $a_1^T a_1 = 1$, es decir que el vector a_1 este normalizado.

La segunda componente principal es el vector $Y_2 = a_2^T X = a_{12}X_1 + a_{22}X_2 + \dots + a_{p2}X_p$ con la condición que el vector unitario a_2 sea ortogonal al vector a_1 , (con el fin de descartar cualquier contribución de a_1) tal que maximiza la varianza $\text{Var}(Y_2)$.

De la misma forma se definen el resto de componentes principales con $j \leq p$ tal que $\text{Var}(Y_j)$ es máxima sujeta a la restricción que a_j es un vector de módulo 1, ortogonal a todos los a_k con $k < j$. Como $\text{Var}(Y_j) = a_j^T \Sigma a_j$, donde Σ es la matriz de varianza-covarianza de X , la maximización de esta varianza y de aquí la forma de hallar las componentes principales está relacionada con Σ . La solución de este problema viene con el siguiente teorema:

Teorema.- Sea Σ la matriz de varianza-covarianza de un vector aleatorio $X=[X_1, X_2, \dots, X_p]^T$ que tiene pares de autovalores-autovectores normalizados $(\lambda_1, e_1), (\lambda_2, e_2), \dots, (\lambda_p, e_p)$, donde los autovalores están ordenados de tal manera que $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_p \geq 0$. Entonces la j -ésima componente principal viene dada por:

$Y_j = e_j^T X$ y la varianza de esta componente principal Y_j es $\text{Var}(Y_j) = e_j^T \Sigma e_j = \lambda_j, j=1, 2, \dots, p$.

Como la covarianza $\text{Cov}(Y_i, Y_j) = \text{Cov}(e_i^T X, e_j^T X)$ entonces,

$$\text{Cov}(Y_i, Y_j) = e_i^T \text{Cov}(X, X) e_j = e_i^T \text{Var}(X) e_j = e_i^T \Sigma e_j = \lambda_j e_i^T e_j = 0 \text{ si } i \neq j. \quad \dots (71)$$

Este resultado indica que los componentes principales no están correlacionados y el teorema nos permite encontrar entonces las componentes principales $Y_j = e_j^T X$. Tomando los λ_j más significativos podemos reducir la dimensión del espacio de trabajo.

3.3 Fracción mínima de ruido (MNF)

Aunque PCA es una transformación que produce componentes que muestran una calidad de imagen decreciente al incrementar el número de componentes, agrupa la información relevante en muy pocas imágenes dificultando su análisis. Esto es debido a que el ruido intrínseco o ruido blanco está incluida dentro de la información recolectada. Un procedimiento adecuado sería identificar el ruido, separarlo y proceder con la reducción de la dimensión como en una transformación PCA. Este procedimiento se conoce como fracción de mínimo ruido (MNF por sus siglas en inglés) [17] el cual también produce imágenes que van disminuyendo en calidad. La transformación MNF requiere el conocimiento de las matrices de varianza-covarianza tanto de la señal como del ruido.

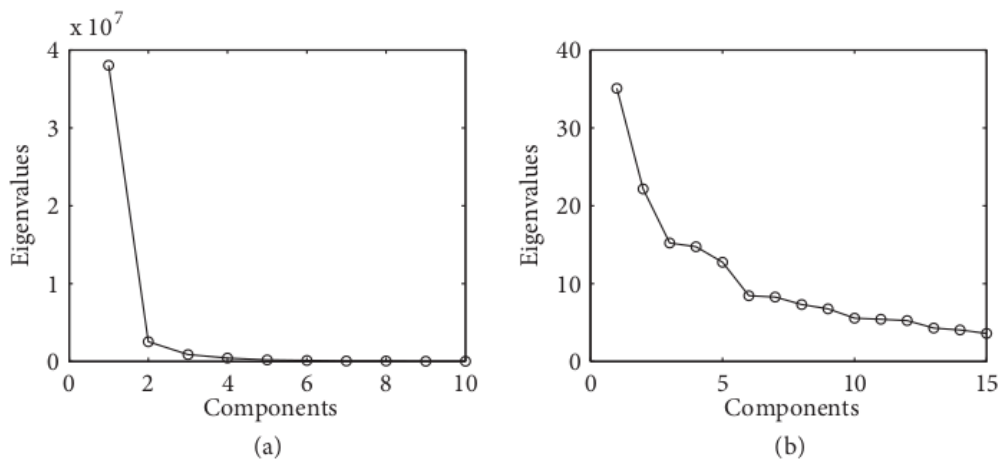


Figura 19. Comparación del número de componentes significativas en PCA y MNF [18].

La figura 19 muestra como en PCA el número de componentes más correlacionados (proporcional a los autovalores de la transformación PCA) decae rápidamente, mientras que esto no sucede en la transformación MNF. Esto permite separar las características de la imagen y no agruparlas en unas pocas componentes.

MNF es un procedimiento de vectores propios basado en la estructura de covarianza del ruido en el conjunto de datos de la imagen que consta esencialmente de dos pasos:

- ✓ Se usa la matriz de covarianza de ruido para descorrelacionar y reescalar el ruido en los datos (blanqueamiento de ruido). De esta forma, el ruido tiene variación unitaria y no tiene correlaciones de banda a banda.
- ✓ Se realiza una transformación PCA estándar a los datos blanqueados por ruido.

A continuación, se detalla este procedimiento:

Consideremos un conjunto de datos multivariados de p -bandas con reflectancias

$$Z_i(x), i=1, \dots, p \quad \dots (72)$$

donde x especifica las coordenadas de la muestra.

Asumimos que:

$$Z(x)=S(x)+N(x) \quad \dots (73)$$

donde $Z^T(x)=(Z_1(x), \dots, Z_p(x))$ y $S(x)$ y $N(x)$ son las señales no correlacionadas y las componentes del ruido de $Z(x)$.

Así,

$$\text{Var}(Z(x)) = \Sigma = \Sigma_S + \Sigma_N \quad \dots (74)$$

donde Σ_S y Σ_N son las matrices de covarianza de $S(x)$ y $N(x)$, respectivamente.

El objetivo de la transformación MNF es seleccionar componentes de modo que maximicen la relación señal-ruido en lugar del contenido de información. Para hacer esto, se deben conocer las matrices de covarianza tanto de la señal Σ_S como del ruido Σ_N . Σ_S se calcula de la misma manera que se hace para la transformada PCA. Σ_N se puede estimar haciendo cálculos estadísticos en la imagen. Existen muchos métodos para estimar la matriz de covarianza de ruido y en este trabajo se utiliza el método sugerido por Roger y Arnold basado en residuos [19].

El MNF estándar de Green [17] ordena las bandas de salida a lo largo de los ejes de relación señal ruido (SNR) mínima utilizando la descomposición propia $\Sigma_S \Sigma_S^{-1} = V \Lambda V^{-1}$ donde V es la matriz ortogonal que contiene los vectores propios de $\Sigma_S \Sigma_S^{-1}$, y Λ es la matriz diagonal que contiene los valores propios correspondientes. Además, la matriz proporciona los vectores base para el cubo de datos transformado.

Las componentes de MNF muestran una calidad de datos en constante disminución de ruido. MNF, como la transformada PCA, es un procedimiento de vectores propios, pero incorpora la estructura de covarianza del ruido presente en el conjunto de datos de imagen. MNF es mucho más eficaz que PCA para crear un conjunto de imágenes ordenadas según la calidad de la imagen. Esto da como resultado una identificación y eliminación más confiable de componentes

ruidosos y permite la conservación de componentes que se considera que contienen información útil.

Para aplicar esta transformación debe determinarse primero la matriz de covarianza del ruido Σ_N .

3.3.1 Estimación de la matriz de covarianza del ruido

Existen varios métodos para estimar la matriz de covarianza del ruido tales como el método basado en residuos [19][20] la cual toma en cuenta la correlación interbandas, la diferencia entre primeros vecinos [17] así como la predicción basada en el método de regresión lineal [21], entre otros. Dentro de estos métodos el método basado en residuos es uno de los que ha dado mejores resultados [22], y es el método que usamos para realizar la transformación MNF. En este método el ruido se obtiene siguiendo un procedimiento similar al de descomponer la matriz de varianza-covarianza de los datos como:

$$\Sigma = D_K E_K D_K \quad \dots (75)$$

donde la matriz D_K es una matriz diagonal con elementos $\{\sigma_1, \sigma_1, \dots, \sigma_n\}$, donde σ_i^2 son los elementos de la diagonal de Σ , es decir la varianza de los pixeles, y E_K es una matriz con 1's en la diagonal y elementos ρ_{mn} , con $m \neq n$, fuera de la diagonal iguales a los coeficientes de correlación de Σ .

De la misma forma es posible descomponer Σ^{-1} como,

$$\Sigma^{-1} = D_G E_G D_G \quad \dots (76)$$

donde D_G y E_G son matrices similares a D_K y E_K pero referidos a la matriz Σ^{-1} .

Si los elementos de la matriz diagonal D_G son $\{\zeta_1, \zeta_2, \dots, \zeta_n\}$, con ζ_i^2 , elementos de la diagonal de Σ^{-1} , se demuestra [22] que la matriz de covarianza del ruido Σ_N puede estimarse como la matriz diagonal con elementos $\{1/\zeta_1^2, 1/\zeta_2^2, \dots, 1/\zeta_n^2\}$.

Una vez estimada la matriz de covarianza del ruido se puede proceder a realizar la transformación MNF.

3.4 Desmezclado de un espectro HSI

En el análisis de una imagen es frecuente estar interesado en las abundancias de los elementos presentes en la imagen. Este es un proceso que puede realizarse analizando la correlación entre las reflectancias de los píxeles para diferentes longitudes de onda, es decir haciendo uso de herramientas estadísticas como las descritas anteriormente.

El proceso seguido para el desmezclado, es decir para obtener las abundancias de los elementos presentes en una imagen hiperespectral es el siguiente:

- a) Después del registro de la imagen HSI, se realiza un proceso de pre-procesamiento el cual incluye los métodos Standard Normal Variance (SNV) [23] y Minimum Noise Fraction (MNF).
- b) El siguiente paso es la obtención de los endmembers o píxeles más representativos en la imagen con sus respectivos pesos. El método usado para la obtención de los endmembers en esta tesis es el método Simple growing algorithm (SGA) [24].
- c) A continuación se realiza un análisis de subpixel para identificar las abundancias de los minerales en cada endmember usando un ajuste SALD (Simulated Annealing Linear Decomposition) [25].

El proceso completo que se siguió se muestra en la figura 20.

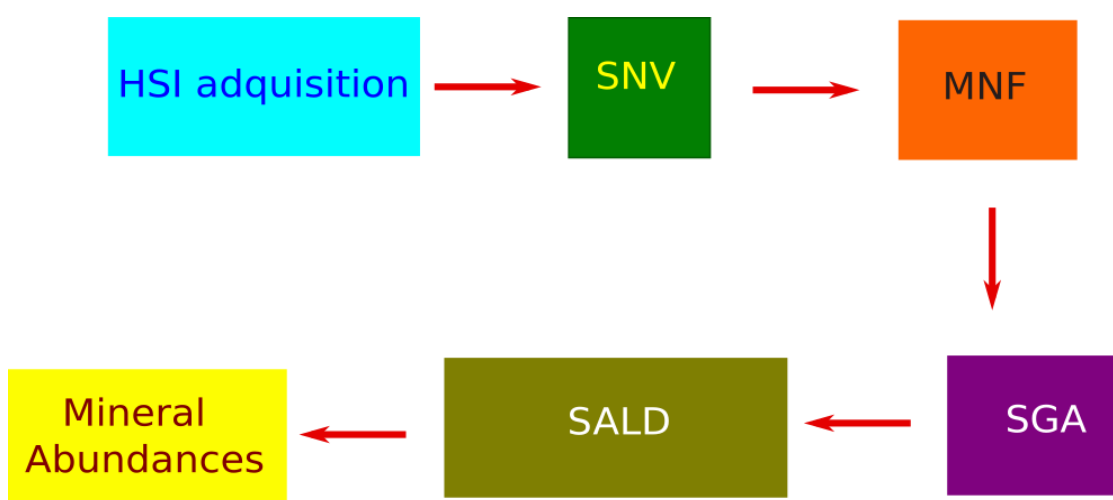


Figura 20. Diagrama que muestra la secuencia seguida en este trabajo para determinar las abundancias de minerales en una muestra.

3.4.1 Pre-Procesamiento de datos

Los datos obtenidos a partir de las HSI, contienen ruido por scattering, básicamente debido a la diferencia en el tamaño de las partículas en la muestra lo que causa que la luz se desvíe a diferentes ángulos. Los efectos de dispersión producen un desplazamiento en la línea base de los espectros a lo largo del eje vertical y una inclinación del espectro.

La idea detrás de las correcciones por dispersión es deshacerse los efectos que no están relacionados a la naturaleza química de la muestra y quedarse con aquellos efectos que dependen de la morfología de la muestra. Para esto, se toma en cuenta el promedio y la desviación estandar de todos los datos para cada espectro calculado. El valor promedio es sustraído de las reflectancias para cada punto y el resultado se divide por la desviación estandar.

Este procedimiento se conoce como la técnica Standard Normal Variance (SNV) y se le denomina una técnica de procesamiento para los datos en el infrarrojo cercano. En la figura 21 se muestra un espectro de reflectancia sin procesar y otro procesado con SNV. La corrección SNV se realiza para cada uno de los espectros y puede definirse en términos de dos pasos:

1. Centrar en su valor medio $\langle X_i \rangle$ cada uno de los espectros restándoles este valor medio.
2. Dividir cada espectro centrado por su desviación estandar $X^{SNV} = (X - \langle X_i \rangle) / \sigma_i$.

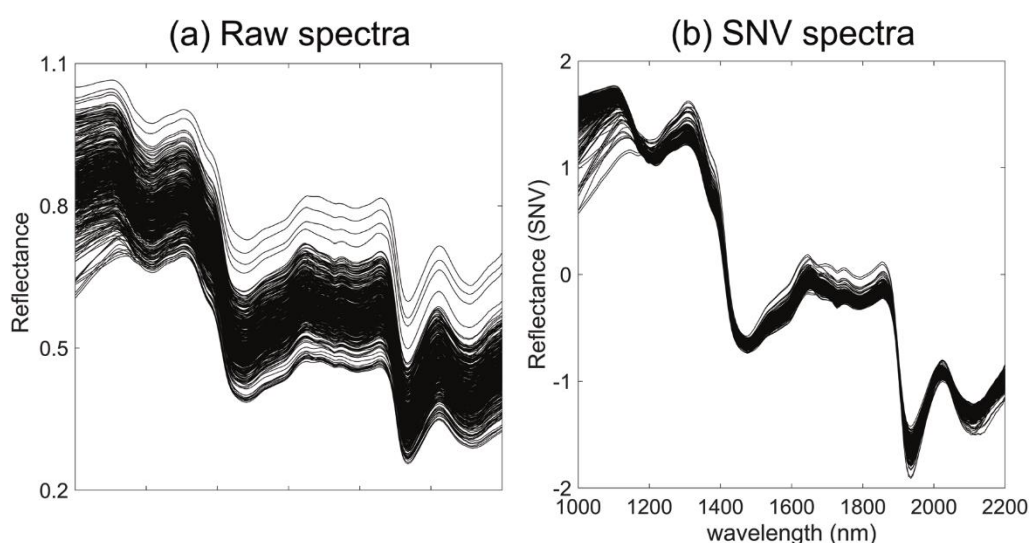


Figura 21. a) Espectros de reflectancia sin procesar y b) Espectros de reflectancia procesados [26].

Hay otras fuentes de ruido en los datos de las imágenes hiperespectrales que pueden oscurecer fácilmente las características hiperespectrales sutiles. Los datos son contiguos en un amplio rango espectral y pueden incluir rangos espectrales muy ruidosos. En general, existen bandas que son esencialmente ruido, así como bandas que tienen una relación señal-ruido relativamente baja. Los grupos de bandas están frecuentemente muy correlacionados. Las grandes fluctuaciones en estas regiones pueden enmascarar características sutiles que ocurren dentro de las bandas en esas regiones. Por lo tanto, se requerirá una eliminación cuidadosa del ruido con una pérdida mínima de información para permitir la extracción de información útil.

3.4.2 Simplex Growing Algorithm (SGA), endmembers y HIS

La reflectancia espectral ha sido ampliamente aplicada para detectar la presencia de minerales [14] [15], determinando y cuantificando abundancias de minerales contenidas en las muestras en estudio. Sin embargo, cuantificar las abundancias no es una tarea trivial porque existe una relación no lineal entre los espectros de reflectancia de los endmembers y el espectro de reflectancia de la superficie [16]. El espectro de reflectancia observado puede provenir de un mineral puro o de una mezcla de varios minerales. Además, la reflectancia observada depende del tamaño de grano de los minerales [17] y de los efectos de oscurecimiento como la meteorización espacial [18].

Para considerar estos efectos, se debe considerar un conjunto de endmembers para cada posible tamaño de grano y la reflectancia espectral observada debe depender de la abundancia de los endmembers [16]. Una identificación correcta de los endmembers es esencial porque el modelo puede utilizar miembros adicionales o incorrectos para cubrir las incertidumbres de medición que dan como resultado valores de abundancia pequeños erróneos o en el peor de los casos, valores de abundancia negativos o excesivamente grandes. En consecuencia, la identificación de los endmembers antes de la estimación de abundancia es una parte crítica del procedimiento de desmezclado.

Un paso previo al desmezclado es identificar los endmembers en la escena y las abundancias fraccionarias en cada píxel. Para identificar los endmembers usamos un enfoque geométrico del problema, usando el concepto de simplex [27].

Un simplex [27] en el espacio euclidiano n-dimensional es un sólido convexo con $n + 1$ vértices. Así, en un espacio unidimensional, un simplex es solo el segmento de línea entre dos puntos especificados. Un simplex en un espacio bidimensional es un triángulo de tres vértices, un simplex en un espacio tridimensional es un tetraedro (cuatro vértices), y así sucesivamente. El contenido de un simplex (es decir, la longitud de un simplex unidimensional, el área de un simplex bidimensional, el volumen de un simplex tridimensional, etc.) se puede expresar de manera muy simple en función de las coordenadas de los $n + 1$ vértices.

Primero, observamos que cualquier vértice de un simplex en el espacio k-dimensional puede considerarse como el vértice de una "pirámide" en una base $(k-1)$ -dimensional definida por los otros vértices. Si V_{k-1} denota el contenido o "volumen" de la base y h la distancia perpendicular del vértice al sub-espacio que contiene la base, el contenido V_k de la pirámide está dado por

$$V_k = \int_{h=0}^{h_k} V_{k-1} \left(\frac{h}{h_k} \right)^{k-1} dh = \frac{h_k}{k} V_{k-1} \quad \dots (77)$$

Por lo tanto, aplicando esta fórmula a los vértices (en cualquier orden) de forma recursiva, comenzando con $k = n$, tenemos

$$V_n = \frac{1}{n!} h_n h_{n-1} h_{n-2} \dots h_1 \quad \dots (78)$$

donde h_1 es simplemente la distancia entre los dos primeros vértices, h_2 es la altura del tercer vértice sobre la línea que contiene esos dos vértices, h_3 es la altura del cuarto vértice sobre el plano que contiene los tres primeros vértices, y así sucesivamente. Por tanto, el contenido de un simplex n-dimensional es $1 / n!$ multiplicado por las "alturas" de los vértices (tomados en cualquier secuencia lineal) sobre el subespacio que contiene los vértices anteriores.

A continuación, demostramos que este volumen se puede expresar como un determinante donde se ubican las coordenadas de los vértices que forman el simplex. Esta expresión es la que aparece en los textos y artículos con frecuencia.

Para determinar las "alturas" requeridas, comenzamos por señalar que, sin afectar el contenido, toda la disposición de los vértices se puede trasladar rígidamente en el espacio de modo que un vértice se ubique en el origen. Entonces podemos construir una matriz $n \times n$ que consta de las

n coordenadas de cada uno de los n vértices restantes. Por ejemplo, con n = 3 construimos la matriz

$$\begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{bmatrix} \quad \dots (79)$$

donde (x_i, y_i, z_i) con $i = 1,2,3$ son las coordenadas de tres vértices, recordando que el cuarto vértice se encuentra en el origen.

Cualquier rotación rígida de estos puntos alrededor del origen dejará el determinante de esta matriz sin cambios (porque el determinante de una matriz de rotación es 1 por definición). En general, podemos aplicar una rotación n-dimensional que coloca n-1 de los vértices en un subespacio ortogonal a uno de los ejes. Por lo tanto, en nuestro ejemplo podemos rotar la configuración para que los dos primeros vértices estén en el plano x-y, lo que da como resultado una matriz transformada con el mismo determinante, pero con ceros en todos menos el último elemento de la última fila, por lo que tenemos

$$\begin{vmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{vmatrix} = \begin{vmatrix} x_1' & y_1' & 0 \\ x_2' & y_2' & 0 \\ x_3' & y_3' & z_3' \end{vmatrix} \quad \dots (80)$$

Aquí, la coordenada z_3' es la altura del tercer vértice sobre el subespacio que contiene los otros tres vértices (incluido el origen). Por descomposición de cofactores de la última columna, y dejando que h_3 denote z_3' , este determinante se puede escribir como

$$\begin{vmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{vmatrix} = \begin{vmatrix} x_1' & y_1' & 0 \\ x_2' & y_2' & 0 \\ x_3' & y_3' & z_3' \end{vmatrix} = h_3 \begin{vmatrix} x_1' & y_1' \\ x_2' & y_2' \end{vmatrix} \quad \dots (81)$$

La matriz 2 x 2 en la expresión de la derecha contiene las coordenadas de dos de los vértices de un simplex bidimensional en el plano x-y (recordando que el tercer vértice está en el origen). Sin afectar el valor de este determinante, podemos aplicar una rotación que coloque el primer vértice en el eje x, por lo que tenemos

$$\begin{vmatrix} x_1' & y_1' \\ x_2' & y_2' \end{vmatrix} = \begin{vmatrix} x_1'' & 0 \\ x_2'' & y_2'' \end{vmatrix} \quad \dots (82)$$

Aquí, la coordenada y_2 es la altura del segundo vértice sobre el subespacio (es decir, la línea) que contiene los otros dos vértices (incluido el origen). Por descomposición del cofactor de la última columna, y haciendo que h_2 denote y_2 , este determinante se puede escribir como $h_2 x_1$. Teniendo en cuenta que x_1 es la altura h_1 del primer vértice sobre el origen, tenemos

$$\begin{vmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{vmatrix} = h_3 h_2 h_1 \quad \dots (83)$$

Luego, encontramos que el contenido del simplex tridimensional original con un vértice en el origen está dado en términos de las coordenadas de los otros tres vértices por

$$V_3 = \frac{1}{3!} \begin{vmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{vmatrix} \quad \dots (84)$$

Esta derivación se generaliza inmediatamente a símplex en cualquier número de dimensiones. Además, podemos escribir inmediatamente el volumen de un símplex de n dimensiones con $(n + 1)$ vértices arbitrarios, sin requerir que uno de ellos esté en el origen. En nuestro ejemplo tridimensional podemos imaginar cuatro vértices arbitrarios, incluido un cuarto en (x_4, y_4, z_4) , y podemos escribir el contenido de este símplex como

$$V_3 = \frac{1}{3!} \begin{vmatrix} x_1 - x_4 & y_1 - y_4 & z_1 - z_4 \\ x_2 - x_4 & y_2 - y_4 & z_2 - z_4 \\ x_3 - x_4 & y_3 - y_4 & z_3 - z_4 \end{vmatrix} \quad \dots (85)$$

Además, sabemos por descomposición de cofactor que podemos escribir esto como el determinante de una matriz de $n + 2$ dimensiones

$$V_3 = \frac{1}{3!} \begin{vmatrix} 1 & x_1 - x_4 & y_1 - y_4 & z_1 - z_4 \\ 1 & x_2 - x_4 & y_2 - y_4 & z_2 - z_4 \\ 1 & x_3 - x_4 & y_3 - y_4 & z_3 - z_4 \\ 1 & 0 & 0 & 0 \end{vmatrix} \quad \dots (86)$$

Ahora podemos hacer uso del hecho de que agregar un múltiplo de cualquier columna (o fila) a otra columna (o fila) no cambia el determinante.

Para ver por qué esto es así, tenga en cuenta que si cualquier vector de columna c_i se reemplaza con $c_i + kc_j$ donde c_j es uno de los otros vectores de columna, entonces cada cofactor se multiplica por un elemento de $c_i + kc_j$, y por lo tanto está claro que el determinante es la suma del determinante original más el determinante de la matriz original con c_i reemplazado por kc_j . Pero el último determinante es idénticamente cero, porque una columna es múltiplo de otra.

Entonces, aplicando esta proposición a la matriz anterior, podemos agregar x_4 veces la primera columna a la segunda columna, y así sucesivamente, para dar

$$V_3 = \frac{1}{3!} \begin{vmatrix} 1 & x_1 & y_1 & z_1 \\ 1 & x_2 & y_2 & z_2 \\ 1 & x_3 & y_3 & z_3 \\ 1 & x_4 & y_4 & z_4 \end{vmatrix} \quad \dots (87)$$

Esta derivación es completamente general y se aplica a un símplex en n dimensiones, lo que nos permite calcular el contenido en términos de los coeficientes de los $(n + 1)$ vértices.

El algoritmo de crecimiento simplex o simplex growing algorithm (SGA) es un algoritmo para encontrar endmembers el cual va haciendo crecer un simplex incrementando un vértice a la vez buscando siempre el volumen máximo para encontrar los endmembers.

Los volúmenes de los simplex se calculan usando los determinantes de una matriz formada por los endmembers.

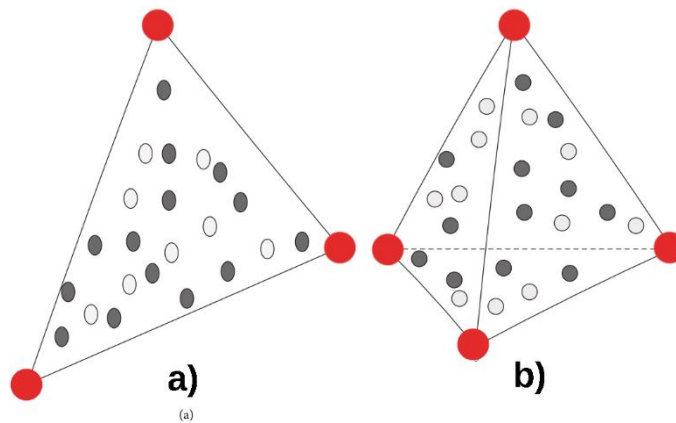


Figura 22. a) Diagrama esquemático de un simplex en dos dimensiones. b) Diagrama esquemático de un simplex en tres dimensiones [28].

Para identificar los endmembers se utiliza el algoritmo de crecimiento simplex (SGA) [29][30]. La implementación detallada del proceso simplex creciente mencionado anteriormente se resume a continuación.

1. Determinar el número de $k + 1$ endmembers a generar. k es la dimensión virtual (VD) del espacio muestral.
2. Buscar el píxel con mayor distancia Mahalanobis [16] al origen. Este es el primer endmember y su vector de muestra en el espacio virtual se denomina e_1 .
3. El segundo endmember, con vector muestral e_2 , es el que tiene la mayor distancia de Mahalanobis al primer endmember.
4. El j -ésimo endmember, con vector muestral e_j se determina evaluando el volumen simplex V dado por,

$$V(e_1, e_2, \dots, e_j) = \frac{1}{(j-1)!} \begin{vmatrix} 1 & 1 & \dots & 1 \\ e_1 & e_2 & \dots & e_j \end{vmatrix}$$

El j -mo endmember con vector muestral e_j será aquel que da el mayor volumen V .

Ya que $\begin{pmatrix} 1 & 1 & \dots & 1 \\ e_1 & e_2 & \dots & e_j \end{pmatrix}$ no es necesariamente una matriz cuadrada, se necesita reducir la dimensión. En esta tesis se usa una transformación MNF para reducir la dimensión.

5. El paso 4 se repite hasta que todos los endmembers se han encontrado.

3.4.3 Descomposición lineal usando simulated annealing y análisis subpíxel

Después de la identificación de los miembros finales, se realizó un análisis subpíxel [31][32] para identificar la cantidad de minerales propuestos presentes en las muestras. Una imagen representativa del análisis subpíxel se muestra en la figura 23.

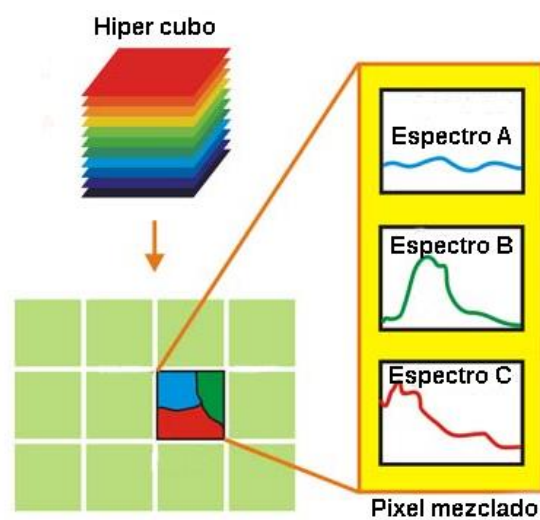


Figura 23. Diagrama esquemático que muestra un píxel con tres componentes diferentes. Adaptado de [31].

Consideramos que cada endmember es una combinación lineal de elementos puros es decir es un píxel mixto y buscamos la contribución de cada uno de estos elementos en cada endmember. Para ello, se debe utilizar una base de datos de minerales. Para manejar el problema de los píxeles mixtos, se considera el concepto de mezcla de subpíxeles.

La idea del mapeo de subpíxeles fue presentada por primera vez por Atkinson [33]. Desde entonces, se han propuesto una serie de técnicas centradas en estimar mejor la determinación de abundancias fraccionarias de subpíxeles y obtener mapas de cobertura terrestre con mayor resolución espacial [18].

En este trabajo se considera una superposición de espectros de reflectancia de los minerales propuestos para cada endmember, esto significa que se realiza una descomposición de subpíxeles. Para esto, se aplica una descomposición lineal de recocido simulado (SALD) [25] a cada endmember aprovechando su simplicidad y fácil implementación. El recocido simulado (SA) ha mostrado buenos resultados en una serie de aplicaciones en optimización y problemas reales y su amplia gama de parámetros le otorga una alta flexibilidad con respecto al problema analizado. En la teledetección multi-hiperespectral, se ha utilizado con éxito para la clasificación [13] [14] y la estimación de abundancias [15].

El recocido simulado es un medio eficaz y general de optimización y está inspirado en la metalurgia, donde la temperatura de un material determina su comportamiento en termodinámica. En el recocido simulado las acciones que realiza el algoritmo dependen completamente del valor de una variable que capta la noción de temperatura. Por lo general, el recocido simulado comienza con una temperatura alta, lo que hace que el algoritmo sea bastante impredecible y enfría gradualmente la temperatura para volverse más estable.

CAPÍTULO IV

Resultados

4.1 Imágenes RGB

Estas nuevas imágenes (R, G y B) son también matrices de 228 x 301 donde cada valor contenido en cada elemento de matriz es un número entero desde 0 hasta 255 indicando la intensidad del color.

Si nos concentramos en el elemento de matriz (i,j) y vemos cómo cambia el valor de este elemento en las 3 matrices R, G y B estaríamos encontrando el espectro de intensidad para ese elemento de matriz.

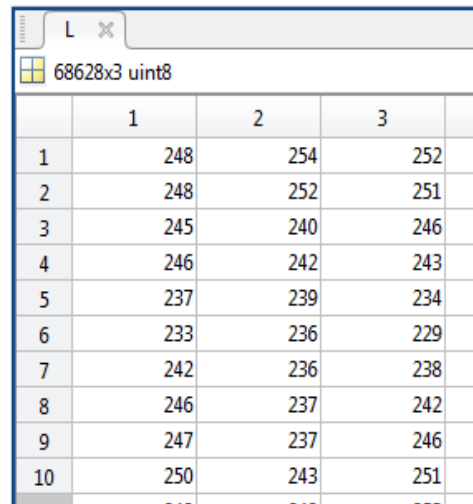
Para hacer esta tarea se escribió un algoritmo en MATLAB que, teniendo como entradas a las matrices R, G y B construya una matriz de Nx3 (N: número de pixeles totales de la imagen) donde cada fila muestra la variación del valor del pixel correspondiente.

```
function L=spectrum(R,G,B)
x=size(R,1); y=size(R,2); l=1;
for i=1:x
    for j=1:y
        L(l,1)=R(i,j,1);
        L(l,2)=G(i,j,2);
        L(l,3)=B(i,j,3);
        l=l+1;
    end
end
end
```

Luego corremos el programa:

```
>> L=spectrum(R,G,B);
```

El resultado obtenido luego de ejecutar esta rutina, se muestra en la figura 24.



	1	2	3
1	248	254	252
2	248	252	251
3	245	240	246
4	246	242	243
5	237	239	234
6	233	236	229
7	242	236	238
8	246	237	242
9	247	237	246
10	250	243	251

Figura 24. Recorte de pantalla de MATLAB de la matriz L conteniendo los valores de los elementos de las matrices R , G y B de cada pixel.

Como cada fila de la matriz L contiene la variación de la intensidad de un pixel, podemos plotear esta variación obteniendo una gráfica de 3 puntos, como se muestra en la figura 25.

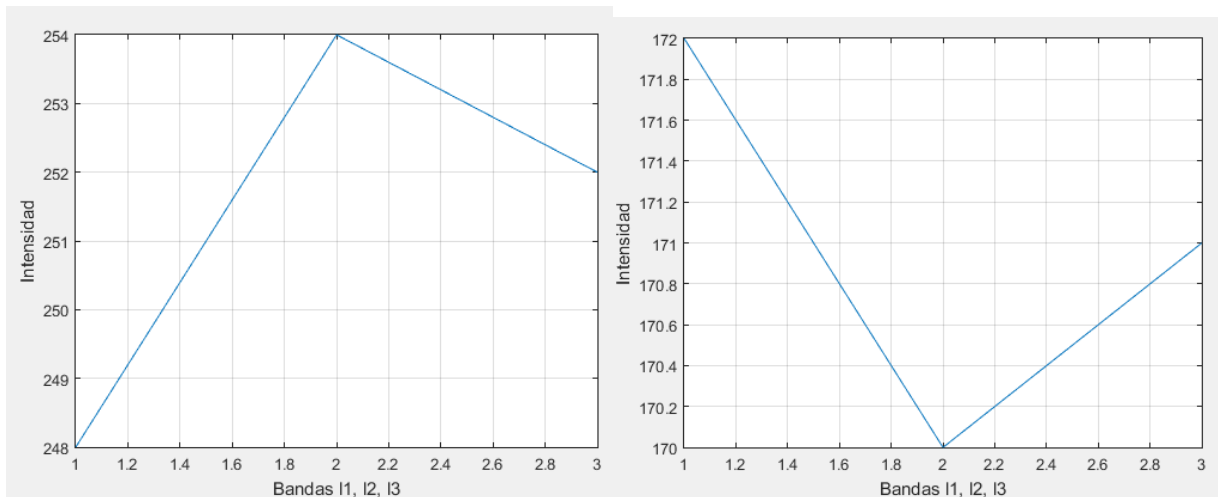


Figura 25. Variación de la intensidad del pixel 1 y del pixel 646.

4.2 Imágenes multiespectrales promedio y descomposición espectral

Se procede ahora a aplicar el proceso de descomposición espectral discutido en la sección 4.3 usando la base espectral en 10 dimensiones construida a partir de la base de datos ASTER. BaseRed es la matriz cuyas columnas contienen los vectores base que definen un espacio 10-dimensional igual al número de canales de las imágenes multiespectrales que se analizan.

Como una primera aplicación de los programas elaborados se construyó un modelo arbitrario al que llamaremos “espectro total”. Se aplicarán los programas de descomposición a este espectro con el fin de encontrar las contribuciones de los elementos base usados.

Consideremos el siguiente espectro total:

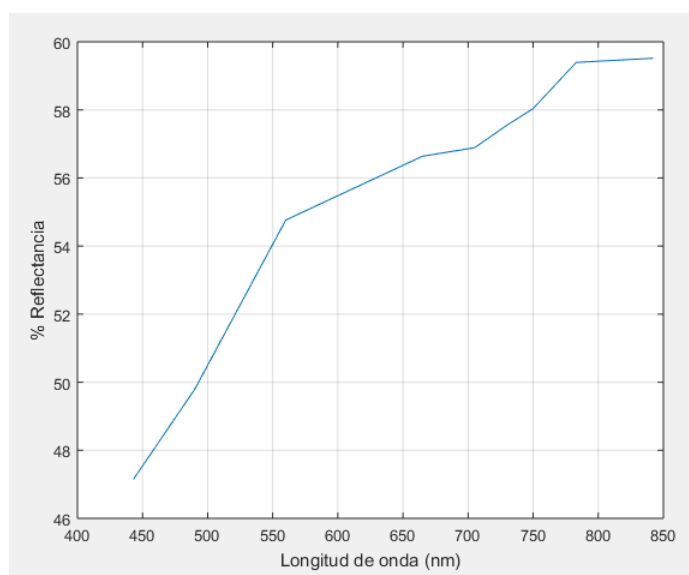


Figura 26. Espectro total de un pixel al azar.

Un espectro como el mostrado en la figura 26 lo representaremos con el vector X , luego será descompuesto usando espectros base $\{\bar{A}_i\}$ que, en este caso, corresponden a nueve materiales (rocas minerales). La elección de estos materiales es en función de su disponibilidad en la biblioteca ASTER.

A continuación, la lista de ellos.

- ✓ Alpita
- ✓ Granito alcalino
- ✓ Granito
- ✓ Riolita
- ✓ Diorita
- ✓ Granodiorita
- ✓ Monzonita
- ✓ Obsidiana riolítica
- ✓ Monzonita de cuarzo

Los espectros de algunos de estos materiales son:

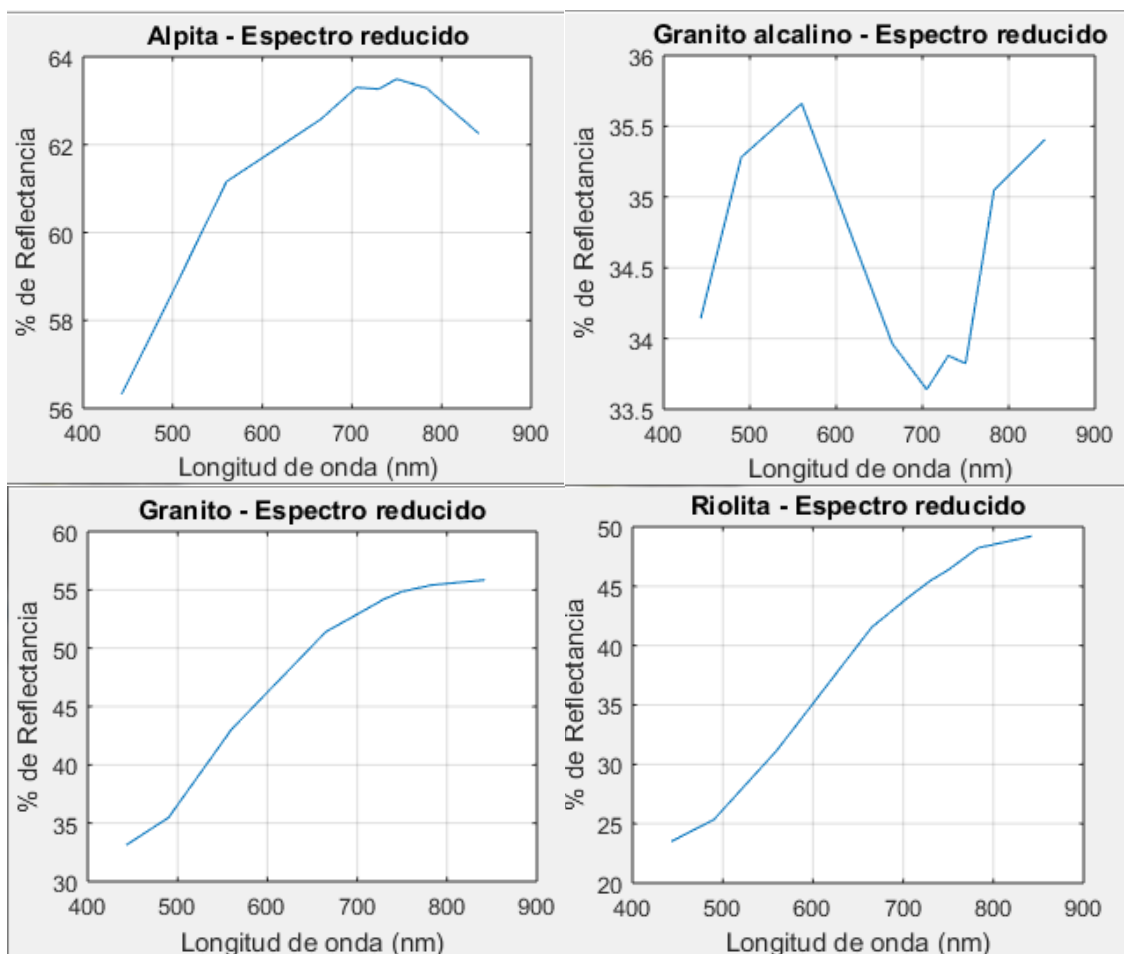


Figura 27. Espectros reducidos de algunos minerales elegidos.

Para descomponer el espectro mostrado en la figura 26, representado por el vector X , en sus espectros base, representados por la base $\{\bar{A}_i\}$, escribimos en MATLAB $L=\text{CombLineal}(X, \text{BaseRed})$ y obtenemos:

```
>> L=CombLineal(X,BaseRed)
L =
0.1000
0.2500
0.0400
0.0700
0.3000
0.0600
0.0570
0.1700
0.3700
```

El vector columna L contiene a los coeficientes λ_i de la combinación lineal, es decir:

$$X = 0.1A_1 + 0.25A_2 + 0.04A_3 + 0.07A_4 + 0.30A_5 + 0.06A_6 + 0.057A_7 + 0.17A_8 + 0.37A_9$$

Donde cada A_i es un espectro base como los mostrados en la Figura 27.

De esta manera hemos encontrado los espectros base presentes en un espectro multispectral, es decir, un espectro obtenido a partir de imágenes multispectrales.

Tengamos presente que una de las primeras tareas de todos los códigos elaborados es leer la dimensión de X , es decir la cantidad de bandas de la imagen o la cantidad de canales de la cámara con la que fue tomada la imagen, de modo que los códigos pueden trabajar con la dimensión que se desee (siempre que se cuente con la cantidad de canales necesarios).

4.3 Registro de la HSI con la cámara PIKA NIR

La cámara hiperespectral RESONON Pika NIR 320 [34], figura 28, permite registrar imágenes en el rango de 900-1700 nm y puede usarse para registrar imágenes en el laboratorio como para registrar imágenes usando un drone o UAV.

Esta cámara toma imágenes escaneando la región de interés, por lo que en el laboratorio usa una plataforma móvil. El software SpectrononPro analiza estas imágenes.



Figura 28. Cámara hiperespectral Pika NIR 320.

La Tabla 2 muestra algunas especificaciones técnicas de la cámara RESONON Pika-NIR.

Tabla 2. Especificaciones técnicas de la Cámara RESONON Pika NIR 320.

Rango Espectral	900-1700 nm	Voltaje, amperaje	12V, 5A
Resolución espectral	4,9 nm	Masa	2,7 kg
Número de canales espectrales	164	Dimensiones (cm)	11.0 x 29.6 x 8.9
Píxeles espaciales	320	Temperatura de operación	41-104 F, 5-40 C
Max. Velocidad de cuadros	520 fps		

El análisis HSI comienza con el registro de la imagen hiperespectral. Las imágenes se tomaron utilizando un sistema de imágenes hiperespectrales de sobremesa con una cámara Pika NIR 320 de RESONON.

4.3.1 Montaje experimental y registro de imágenes

El sistema de adquisición de imágenes consta básicamente de los siguientes elementos:

- ✓ Sistema de luces. Alimentado por una fuente de poder continua de 12 V estable con el fin de garantizar la estabilidad lumínica del sistema.
- ✓ Cámara Hiperespectral Pika NIR [32]. La cámara debe estar previamente calibrada.
- ✓ Plataforma móvil, sincronizada a la velocidad de escaneo.
- ✓ Software de adquisición de datos SpectrononPro de la cámara RESONON.

La figura 29 muestra el sistema de adquisición de datos.

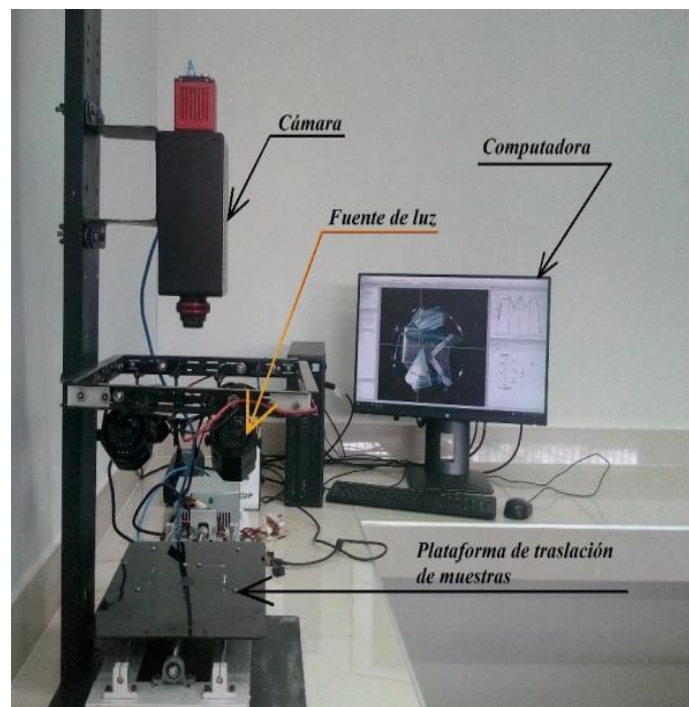


Figura 29. Montaje experimental donde se muestra el sistema de adquisición de Laboratorio que usa una cámara PIKA NIR [33].

Para la calibración de la cámara se siguió el siguiente procedimiento [32]:

- ✓ Enfoque. Ajustar el enfoque de la cámara para lograr una visualización óptima de la muestra usando una llave Allen 5/64, de acuerdo a lo indicado en [32].

- ✓ Ajuste de deformaciones. Verificar que al escanear la cámara no deforme las imágenes de los objetos. Esto se logra ajustando el tiempo de escaneo, controlado por el tiempo de integración de la Imagen (Controlado desde el Software SpectrononPro (integration time) y los cuadros por segundo de la toma que realiza la cámara (Frame Rate).
- ✓ Calibración de los espectros. En primer lugar, se debe cubrir la lente de la cámara para obtener la corriente “oscura”. La remoción de corriente oscura se realiza desde el Software (“Dark Current” en la barra de herramientas). A continuación, se usa la referencia de Teflón blanco como el máximo de reflectancia). Esta es la opción “Response Correction cube” en la barra de tareas del Software SpectrononPro.

Finalizada la calibración se procede a registrar las imágenes de las muestras en estudio. Los espectros obtenidos son guardados en la computadora en formato *.bil*

4.4 Imágenes Multiespectrales Promedio y Series de Fourier

4.4.1 Representación de un espectro por una Serie de Fourier

La figura 30 muestra una colección de 168 puntos en R^2 que representa el espectro de un material.

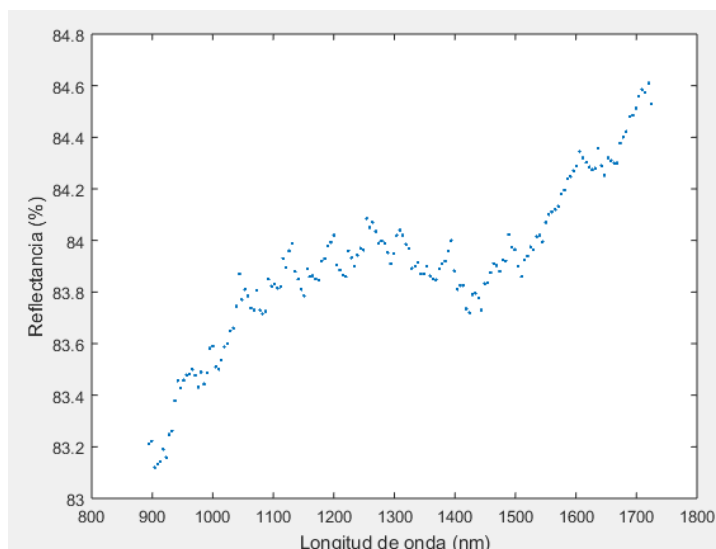


Figura 30. Espectro de una muestra obtenido con la cámara hiperespectral.

4.4.2 Descomposición de un espectro en una Serie de Fourier

La figura 31 muestra una serie de Fourier con $k=1$. Se aprecia una baja correlación entre la serie y el espectro.

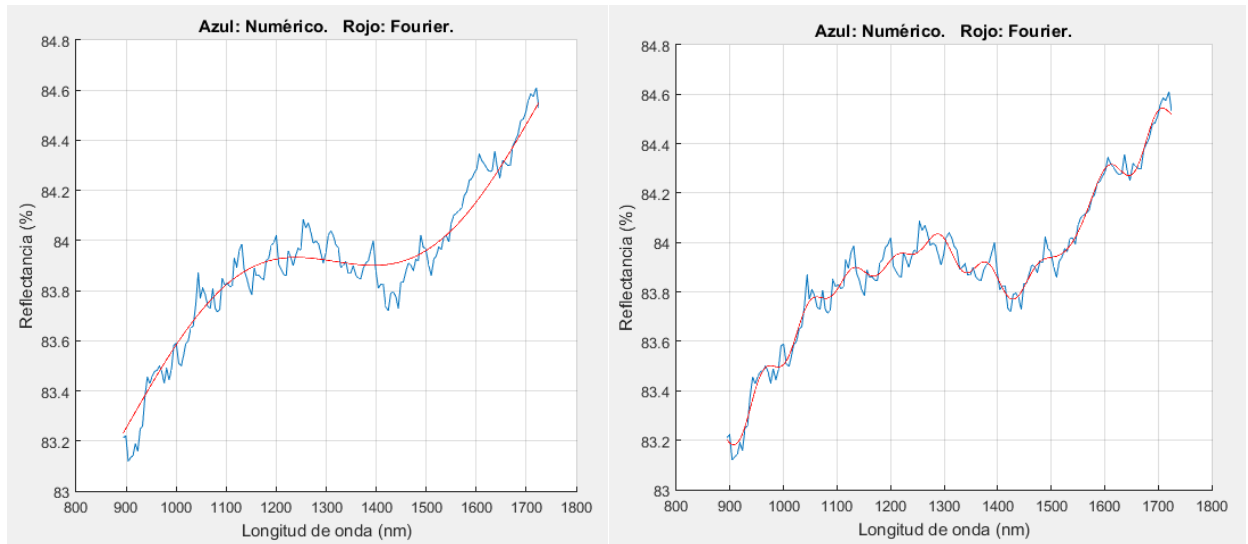


Figura 31. Serie de Fourier con $k=1$, $r^2=0.85$ y $k=10$, $r^2=0.94$ respectivamente.

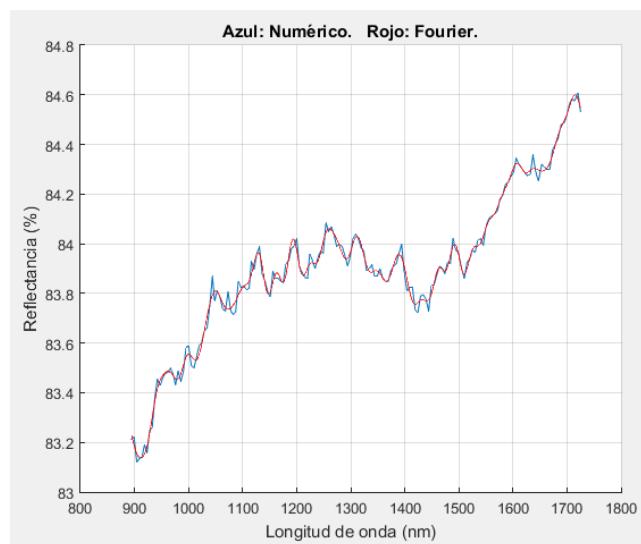


Figura 32. Serie de Fourier con $k=30$, $r^2=0.98$.

Como se muestra en la figura 32, con una Serie de Fourier con $k=30$ se encuentra una alta correlación entre la serie y el espectro.

4.4.3 Descomposición de un espectro representado por una serie de Fourier en componentes base

Para poner a prueba el código Identify y verificar su funcionamiento, contamos con muestras caracterizadas de tierras del yacimiento llamado Las Bambas. Tenemos 6 muestras que llamaremos Muestra A, B, C, D, E y F.

Usando la cámara hiperespectral Pika NIR de RESONON y la estructura de soporte e iluminación para pruebas de laboratorio con una plataforma móvil que facilita el escaneo de la muestra, se tomaron imágenes hiperespectrales de las 6 muestras.

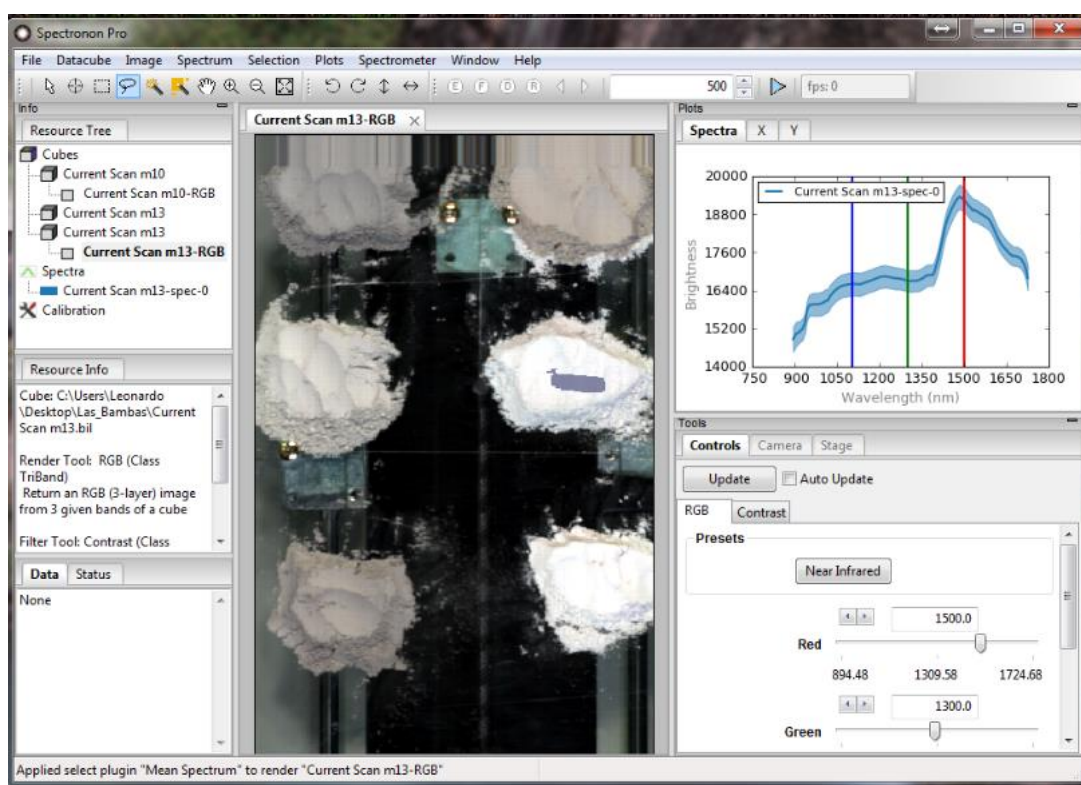


Figura 33. Software SpectrononPro en funcionamiento

Como se aprecia en la figura 33, el software provisto por los fabricantes, SpectrononPro, permite visualizar las imágenes hiperespectrales así como graficar el espectro de un pixel o de un conjunto de pixeles de la imagen. Usando el software SpectrononPro se obtuvieron los espectros (figuras 34, 35, 36) de cada una de las muestras, luego los datos numéricos del espectro fueron exportados en toma de texto para ser añadidos a MATLAB que es el software en el cual venimos trabajando.

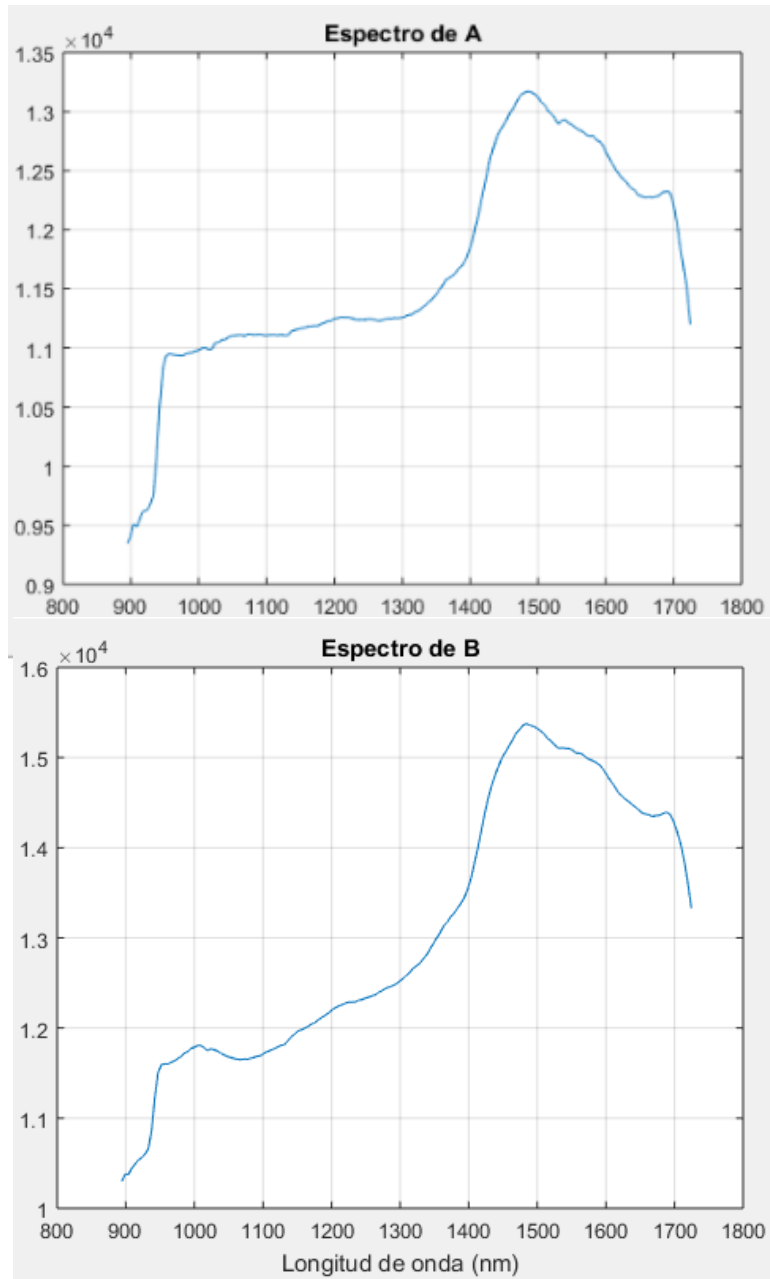


Figura 34. Espectros obtenidos de las muestras A y B en unidades arbitrarias.

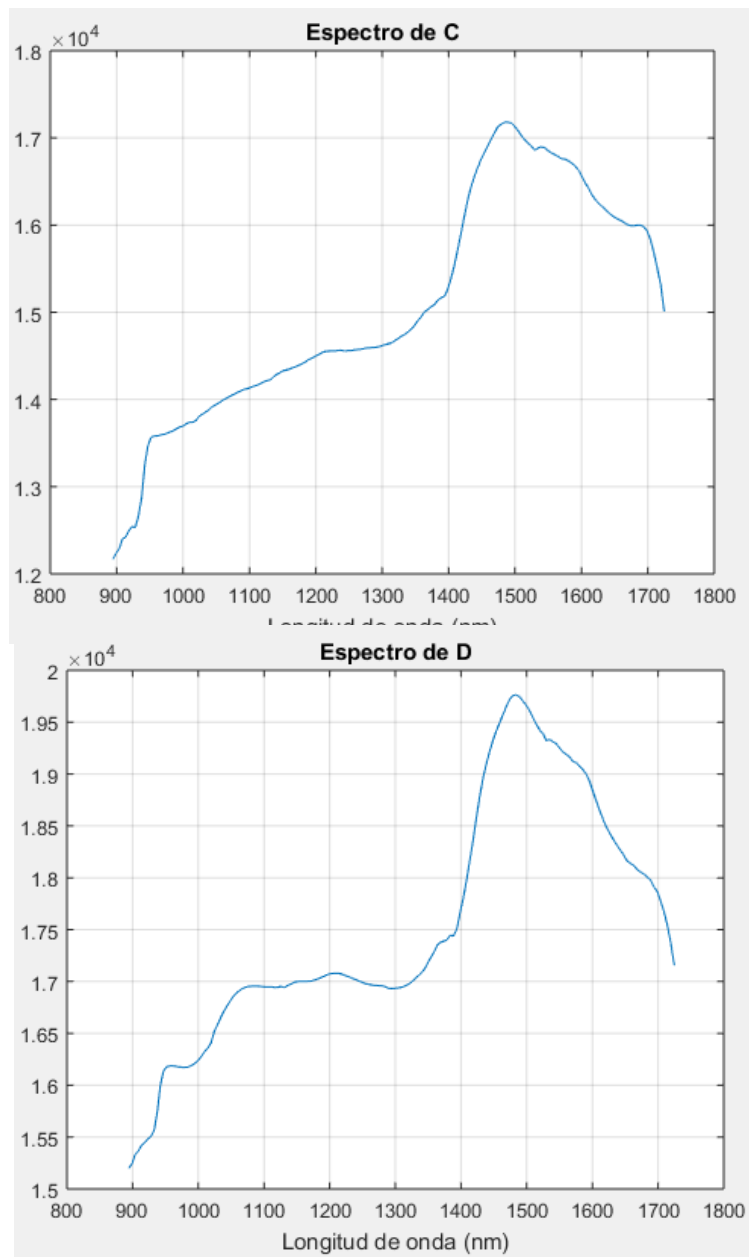


Figura 35. Espectros obtenidos de las muestras C y D en unidades arbitrarias.

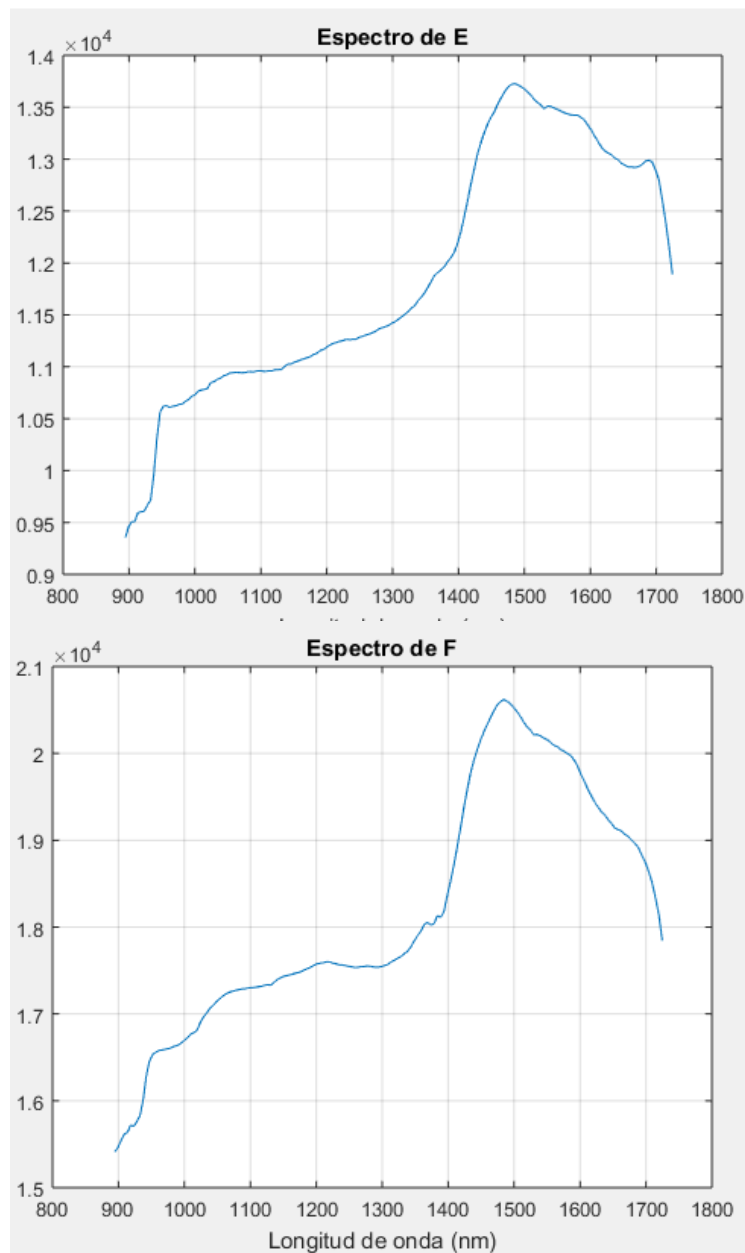


Figura 36. Espectros obtenidos de las muestras E y F en unidades arbitrarias.

A pesar que, a primera vista, los espectros parecen muy similares, un análisis más detallado permite encontrar diferencias que son las que permitirán identificar los elementos que componen este espectro.

Se tienen los resultados del análisis DRX de cada muestra (ANEXO), notándose que la muestra D es la más pura por lo que es la candidata a analizar. En adelante se analizará la muestra D por la razón expuesta.

La zona de Las Bambas, de donde provienen las seis muestras, tiene un color blanquesino y se caracteriza por contener principalmente calcita, dolomita, cuarzo, entre otros.

Considerando esta prioridad y la disponibilidad de espectros en la base de datos, se procedió con el siguiente ensayo de verificación tomando en cuenta los minerales a continuación: albita, calcita, caolinita, chabacita, clinocloro, cuarzo, dolomita, esfalerita, montmorillonita y ortoclasa.

Con los espectros de las muestras y los de la base de datos se procedió a calcular su respectiva Serie de Fourier y posteriormente a correr el código `Identify` y se obtuvo lo siguiente:

Tabla 3. Resultados de los ensayos de identificación de elementos.

	Primera ejecución (UA)	Segunda ejecución (UA)	Tercera ejecución (UA)	Cuarta ejecución (UA)
Cuarzo	0.35	0.07	0.98	0.21 (10.4%)
Calcita	0.11	0.21	1.48	1.56 (77.6%)
Albita	0.37	0.87	0.06	
Montmorillonita	0.29	0.12	0.21	
Dolomita	0.47	0.36	0.64	0.24 (11.9%)
Ortoclasa	0.29	0.41	0.08	
Clinocloro	0.67			
Caolinita	0.06			
Esfalerita	0.26	0.24		
Chabacita	0.04			

Para obtener los resultados de la Tabla 3 se realizaron cuatro ejecuciones del código `Identify` para que, de forma asistida, eliminar candidatos que no representan físicamente el comportamiento esperado.

- ✓ **Primera ejecución.** Descartamos los elementos de menor valor además del clinocloro pues no se espera que sea un elemento de mayor presencia en la muestra.
- ✓ **Segunda ejecución.** Estos resultados los descartamos porque se espera que la calcita sea uno de los principales componentes de la muestra estudiada y no se aprecia este hecho, esto ocurre porque hay elementos que distorsionan los resultados finales. Luego de algunos ensayos, se determinó que el elemento distorsionante es la esfalerita.
- ✓ **Tercera ejecución.** De la lista de elementos reducidos, eliminamos los elementos de menor contribución.
- ✓ **Cuarta ejecución.** A manera de ensayo, se ejecuta con los elementos cuarzo, calcita y dolomita mostrando una presencia prioritaria de la calcita.

Tabla 4. Comparación de las abundancias de los materiales considerados para la muestra D con sus resultados por DRX.

	Cuarta ejecución (%)	DRX (%)
Cuarzo	10.4	2.1
Calcita	77.6	86.6
Dolomita	11.9	9.9
Otros	--	1.4

Si bien los resultados hasta el momento no son exactos, se muestra una clara tendencia predominante de calcita y dolomita, seguido de otros elementos en menor grado. Esto nos indica que el método descrito es aplicable en una primera etapa de eliminación asistida para finalmente trabajar con otro método posteriormente.

En la siguiente etapa se aplicará un método estadístico partiendo de la lista reducida de elementos producida por el método de Series de Fourier.

4.5 Análisis Estadístico de imágenes Hiperespectrales

Con el fin de obtener los valores de referencia de las abundancias de los minerales presentes en la muestra se registró el DRX de la muestra en estudio entre 20° y 90°. La figura 32 muestra el espectro obtenido.

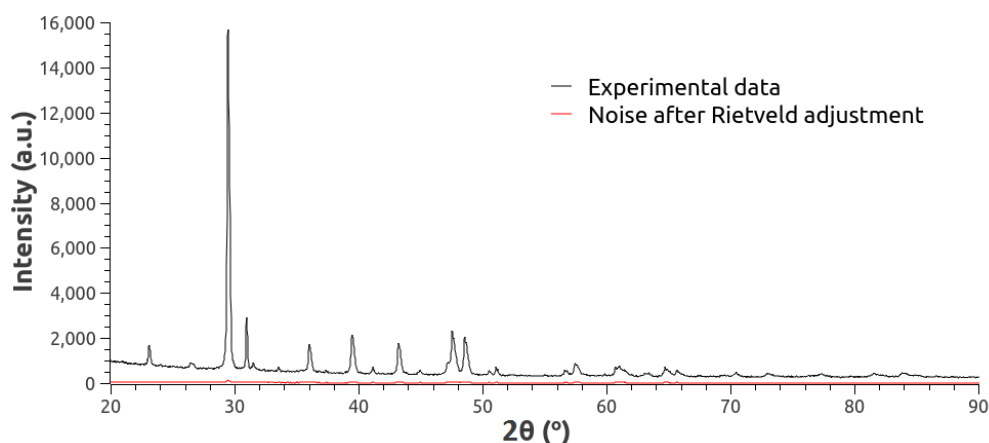


Figura 37. Espectro de DRX de la muestra D con carbonatos analizada en este trabajo. En rojo se observa el ruido después de un ajuste de Rietveld del espectro.

Se utilizó un ajuste Rietveld para identificar y cuantificar los componentes minerales presentes en la muestra estudiada. En rojo se muestra la diferencia entre el espectro de DRX y el espectro ajustado mostrando la calidad del ajuste. Los principales minerales identificados a partir del ajuste Rietveld fueron: Calcita, Dolomita y Cuarzo y sus porcentajes se presentan en la Tabla 5. Estos resultados se usarán como referencia y se compararán con las abundancias de minerales obtenidas mediante un procedimiento HSI.

Tabla 5. Abundancias de los principales elementos identificados por DRX de la muestra en estudio.

	Abundancia (%)
Calcita	86,6
Dolomita	9,9
Cuarzo	2,1
Otros	1.4

4.5.1 Software desarrollado para el análisis de las imágenes

Para analizar las imágenes usando técnicas estadísticas, se elaboró un programa que realiza las siguientes funciones:

1. Lectura de datos en formato *.bil*
2. Pre-procesamiento de imágenes
3. Encontrar los endmembers y sus abundancias
4. Leer de datos de espectros de referencia de la base USGS
5. Realizar un proceso de Simulated annealing para encontrar las abundancias con respecto a los espectros de referencia.
6. Reportar la información e imágenes necesarias.

Todas estas funciones las realiza el programa SALD elaborado en el Lenguaje de programación R. El código escrito se muestra en el Anexo.

4.5.2 Pre-procesamiento de la imagen

A continuación, se realizó el preprocesamiento de los datos registrados, usando un algoritmo implementado en el software R seguido de tratamientos SNV y MNF. Después del preprocesamiento de los datos, la primera tarea fue identificar el VD de los datos transformados. Los autovalores y autovectores de la matriz de correlación formada con la señal y el ruido nos dan esta información.

Los valores propios más pequeños corresponden a autovectores o bandas transformadas que están menos correlacionados y están asociados al ruido blanco de la medida.

La figura 38 muestra los valores relativos de los diferentes valores propios de los datos de la matriz MNF transformada.

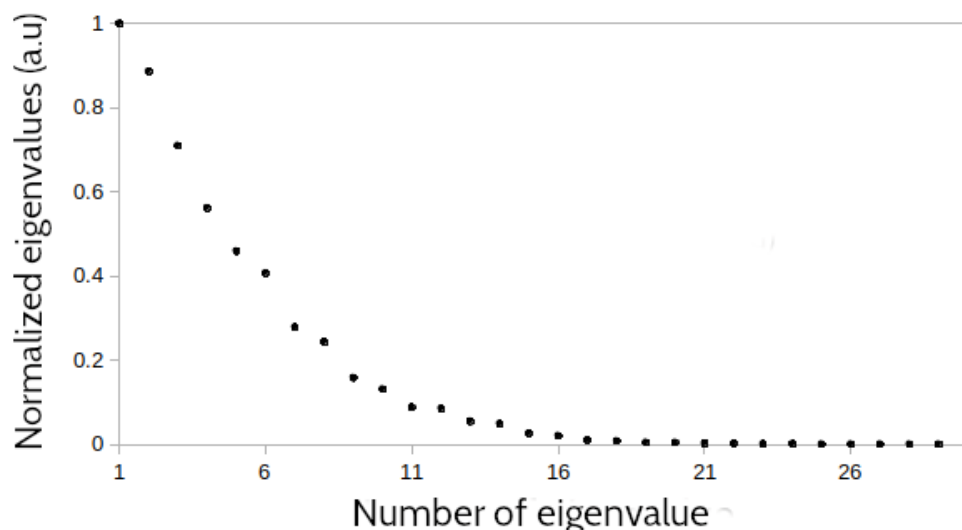


Figura 38. Autovalores normalizados de los datos transformados usando MNF.

Esta figura nos permite identificar los componentes menos correlacionados que básicamente corresponden al ruido. Se puede observar que aproximadamente después de la decimo quinta componente, no existe un valor de correlación significativo, por lo que VD=15 es un valor aceptable que representa a estos datos. Se espera que las abundancias calculadas con VD=15 sean las más cercanas a los valores referenciales comparadas con las calculadas, con otros valores de VD.

4.5.3 Determinación de los endmembers y cálculo de las abundancias

Después de la reducción de la dimensión usando MNF, se abordó la tarea de la búsqueda de los pixeles endmembers. Esto se llevó a cabo mediante el método SGA. SGA no solo nos permite identificar los endmembers en las imágenes, sino que también permite estimar la abundancia de los mismos.

Para identificar los endmembers representamos por un punto cada pixel en el espacio formado por los autovectores de la transformación MNF y con coordenadas dadas por las componentes de las intensidades de cada pixel en esta nueva base. Cada uno de estos puntos corresponde a un píxel en nuestra imagen hiperespectral. Los endmembers en este nuevo espacio definen un volumen que contiene a la mayoría de los pixeles de la imagen. Con el fin de identificar el comportamiento de los endmembers obtenidos usando SGA se localizaron los puntos asociados

a los endmembers en un gráfico 2-D y 3-D (figura 39) considerando los primeros dos (e_1 , e_2) y tres autovectores (e_1 , e_2 , e_3) ordenados con respecto a sus autovalores.

Los resultados obtenidos confirman el buen comportamiento de los endmembers aún considerando solo algunos autovectores.

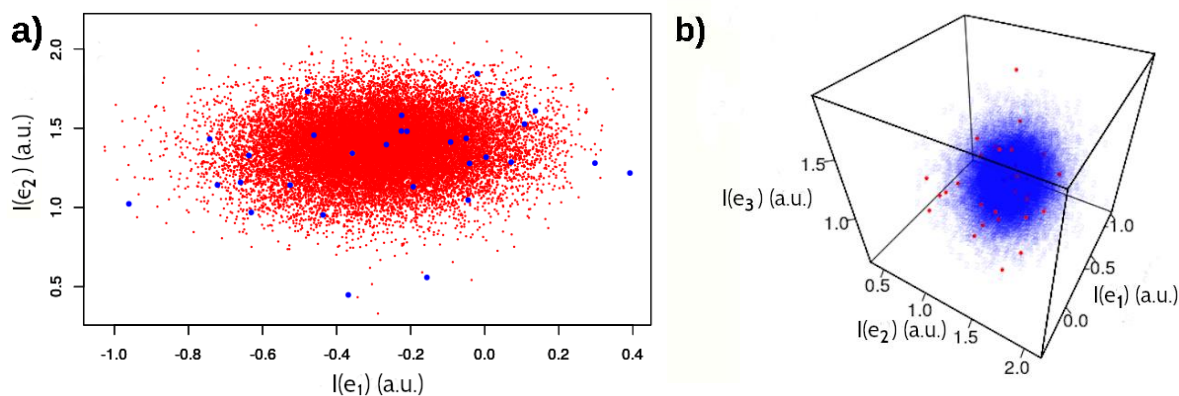


Figura 39. Puntos del espacio de datos transformado, a) usando los autovectores e_1 y e_2 con los endmembers en color azul y el resto de pixeles en color rojo. b) Usado los autovectores e_1 , e_2 y e_3 con los endmembers en color azul y el resto de pixeles en color rojo.

Este espacio bidimensional se formó considerando solos los dos primeros autovectores de la matriz de transformación MNF. En la figura 39 (b) se tienen un espacio 3-D considerando solo los tres primeros autovectores de la matriz de transformación MNF. El comportamiento de los endmembers corresponde a lo que se espera es decir que definen un volumen que cubre a la mayoría de los pixeles de la imagen y forman un buen conjunto representativo de toda la imagen.

Después del proceso de identificación de VD y endmembers, se realizó un análisis subpíxel. Dada una lista de minerales propuestos, los cuales se tomaron del espectro DRX, la tarea consistía en determinar la abundancia de estos minerales presentes en cada endmember de la muestra. Como se mencionó anteriormente, este es un problema con muchas variables y se debe seguir un proceso muy cuidadoso para obtener resultados confiables. Los componentes minerales en la muestra que se usaron fueron Calcita, Dolomita y Cuarzo. En este análisis de subpixeles se usaron los espectros de reflectancia de referencia de la base de datos USGS [34].

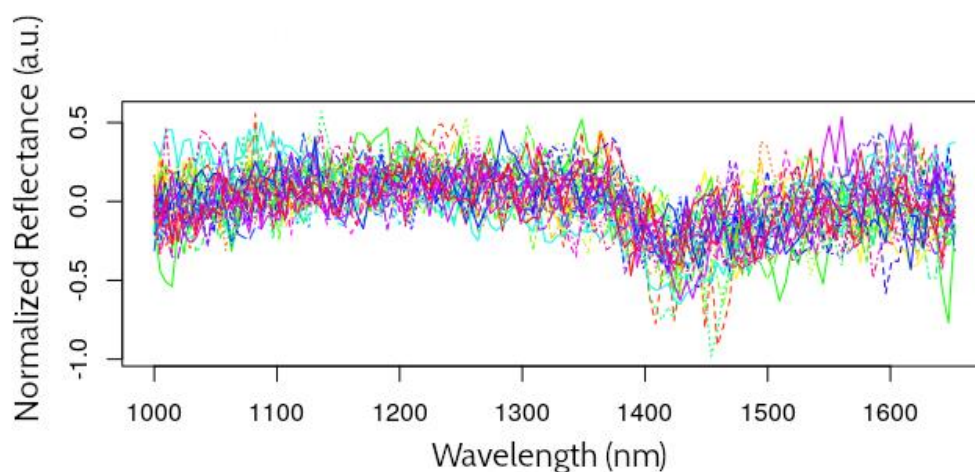


Figura 40. Espectros de reflectancia normalizados de píxeles clasificados como endmembers.

A cada uno de los endmembers identificados le corresponde un píxel en nuestra imagen hiperespectral inicial. Los espectros de reflectancia de los píxeles clasificados como endmembers se muestran en la figura 40. Los espectros de cada uno de estos endmembers se desmezclaron usando SALD, Un algoritmo lineal que usa simulated annealing, y los espectros de reflectancia de referencia. Las abundancias finales se determinaron considerando las abundancias de los endmembers en la imagen de la muestra.

Este proceso se repitió considerando un VD de 5, 10, 15, 20, 25 y 30, con el fin de corroborar la validez de considerar como autovalores significativos solo primeros 15 autovalores de la matriz de transformación MNF.

Las abundancias finales obtenidas para estos diferentes valores de VD se muestran en la Tabla 6.

Tabla 6. Abundancias de calcita, dolomita y cuarzo obtenidos por DRX y por un análisis HSI (considerando VD = 5, 10, 15, 20, 25, 30).

Componente	DRX	VD=5	VD=10	VD=15	VD=20	VD=25	VD=30
Calcita	86.6	82.0	82.0	85.6	84.8	83.1	79.5
Dolomita	9.9	11.0	10.3	8.4	8.4	9.4	11.5
Cuarzo	2.1	7.0	7.7	6.0	6.9	7.5	9.0
Error relativo total	---	2.50	2.76	2.02	2.45	2.66	3.53

En la Tabla 6 se observa que el menor error relativo total en el cálculo de las abundancias obtenidas por un análisis HSI es de 2.02 % y corresponde a $VD = 15$. Este resultado es consistente con los valores propios más significativos de la matriz transformada de MNF.

Es interesante también mencionar que para $VD = 30$ se tiene un aumento significativo en este error relativo, llegando a ser de 3.23, el más alto de los calculados.

La figura 41 muestra el espectro de reflectancia del endmember 6 y el espectro ajustado usando las abundancias finales encontradas para $VD = 15$, mostrando una buena correspondencia entre ellos.

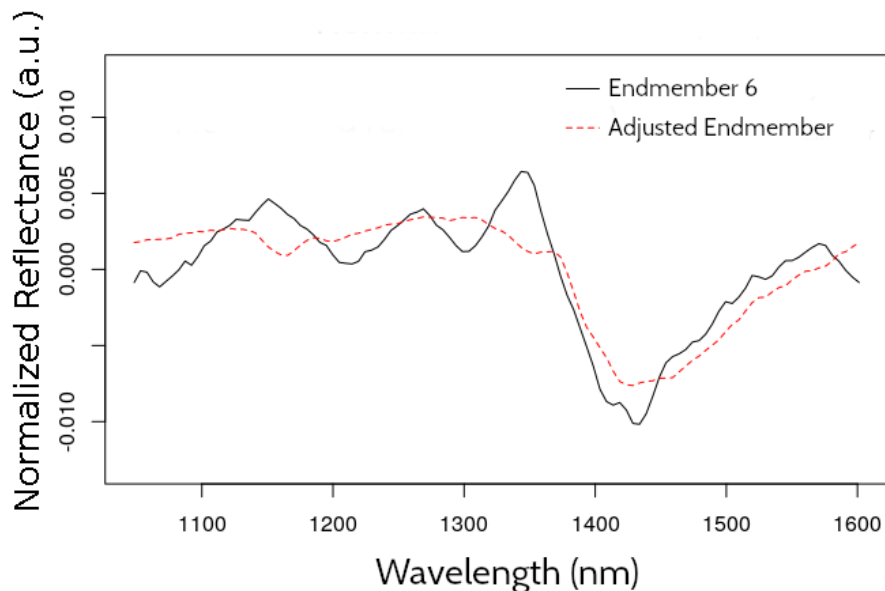


Figura 41. Espectro de reflectancia del endmember 6 y el espectro ajustado usando las abundancias finales encontradas para $VD = 15$.

En la figura 41 se observa una variación significativa en el espectro alrededor de 1430 nm. Esta variación para esta longitud de onda, también se observa en los espectros usados como referencia (USGS Database).

En realidad, la presencia de esta característica es la responsable de los buenos resultados obtenidos a pesar del pequeño rango de datos espectrales usados, como se observa en la figura 41.

Conclusiones

1. Los valores obtenidos para el análisis de muestras consideradas en este trabajo, están asociados con la parte real de los valores de números complejos involucrados en la metodología de Series de Fourier utilizada.
2. El tiempo de cálculo para hallar los coeficientes mostrados en la Tabla 1 fue aproximadamente de dos horas, usando un computador con procesador Core i7 8th Gen y 12 GB de RAM, tiempo relacionado, entre otros factores, a la cantidad de variables consideradas en la ejecución numérica de las ecuaciones involucradas.
3. El ajuste de los espectros usando Series de Fourier nos ha permitido identificar los mejores candidatos para un ajuste posterior mediante un tratamiento estadístico en la identificación de los minerales presentes en la muestra. Los resultados de la cuarta ejecución, Cuarzo 10.4% Calcita 77.6% Dolomita 11.9%, están en muy buena concordancia con los resultados finales a pesar de ser un primer ensayo de las abundancias. Los cuales se encuentran en muy buena correspondencia con los identificados por DRX.
4. Mediante un análisis HSI, se aplica un proceso de desmezclado a una muestra de estudio para obtener las abundancias de los minerales correspondientes. En el proceso de desmezclado se utilizó la técnica MNF y se encontró una dimensión virtual de $VD = 15$ para los datos de la muestra. Los endmembers se encontraron usando SGA y las abundancias finales se obtuvieron usando SALD. Las abundancias finales se compararon con las obtenidas mediante un análisis de Rietveld de los espectros DRX de la muestra y se encontró un error relativo total de 2.02 % en la estimación de las abundancias de Calcita, Dolomita y Cuarzo en la muestra, mostrando una buena correspondencia entre HSI y resultados de DRX. También se calcularon diferentes abundancias finales considerando $VD = 5, 10, 20, 25$ y 30 . En todos los casos se observó un aumento en el error relativo en comparación con los resultados de DRX con el mayor error relativo total para $VD = 30$, mostrando el aumento del ruido blanco cuando se incrementa la dimensión virtual.

Referencias

- [1] L. M. Dale, A. Thewis, C. Boudry, I. Rotar, P. Dardenne, V. Baeten, and J. A. Fernandez, “Hyperspectral Imaging Applications in Agriculture and Agro-Food Product Quality and Safety Control: A Review”, *Applied Spectroscopy Reviews*, vol. 48, 2, 142-159, Jan , 2013.
- [2] M. B. Stuart, A. J. S. McGonigle, and J. R. Willmott, “Hyperspectral Imaging in Environmental Monitoring: A Review of Recent Developments and Technological Advances in Compact Field Deployable Systems”, *Sensors (Basel)*, vol. 19(14):3071, Jul, 2019.
- [3] D. M. Rogge, B. Rivard, J. Zhang, and J. Feng, “Iterative Spectral Unmixing for Optimizing Per-Pixel Endmember Sets”, *IEEE Trans. Geosci. Remote Sens.*, vol. 44, no. 12, 3725-3736, Dec, 2006.
- [4] J. B. Adams, “Visible and near-infrared diffuse reflectance spectra of pyroxenes as applied to remote sensing of solid objects in the solar system”, *J. Geophys. Res.*, vol. 79, 4829-4836, Nov, 1974.
- [5] R. M. Hexter, R. “High-resolution, temperature-dependent spectra of calcite”, *Spectrochim. Acta*, vol. 10, 3, pp. 281-290, Jan, 1958.
- [6] G. R. Hunt, and J. W. Salisbury, “Visible and near-infrared spectra of minerals and rocks: II Carbonates”, *Mod. Geol.*, vol. 2, pp. 23-30, 1971.
- [7] R. G. Burns, “Spectral mineralogy of terrestrial planets: scanning their surfaces remotely”. *Mineral. Mag.* 53, 135–151, April, 1989.
- [8] S. Tompkins, C. M. Pieters., “Spectral characteristics of lunar impact melts and inferred mineralogy”, *Meteorit. Planet. Sci.*, 45 (7), 1152–1169, 2010.
- [9] B. Hapke., “Theory of Reflectance and Emittance Spectroscopy”, 2nd Cambridge Univ. Press, New York, 2012.
- [10] T. Hiroi, C. M. Pieters, “Estimation of grain sizes and mixing ratios of fine powder mixtures of common geologic minerals”, *J. Geophys. Res. Planets* 99 (E5), 10,867–10,879, 1994.
- [11] C. M. Pieters, E. M. Fischer, O. Rode, A. Basu, “Optical effects of space weathering the role of the finest fraction”, *J. Geophys. Res. Planets* 98, 20817–20824 (1993).
- [12] B. S. Penn, ”Using simulated annealing to obtain optimal linear end-member mixtures of hyperspectral data”, *Computers & Geosciences* 28, 809–817, Aug, 2002.

- [13] A. Bardossy and L. Samaniego, "Fuzzy rule-based classification of remotely sensed imagery", *IEEE Trans. Geosci. Remote Sens.*, vol. 2, no. 2, pp. 262–274, Mar, 2002.
- [14] A. Robin, S. L. Hegarat-Masclé, and L. Moisan, "Unsupervised Subpixelic Classification Using Coarse-Resolution Time Series and Structural Information", *IEEE Trans. Geosci. Remote Sens.*, vol. 46, no. 5, pp. 1359–1374, May 2008.
- [15] P. Debba, E. Carranza, F. van der Meer, and A. Stein, "Abundances estimation of spectrally similar minerals by using derivative spectra in simulated annealing," *IEEE Trans. Geosci. Remote Sens.*, vol. 44, no. 12, pp. 3469–3658, Dec. 2006.
- [16] Peter Bajorski, *Statistics for Imaging, Optics and Photonics*, John Wiley & Sons, 2012.
- [17] A. A. Green, M. Berman, P. Switzer, and M. D. Craig, "A transformation for ordering multispectral data in terms of image quality with implications for noise removal," *IEEE Trans. Geosci. Remote Sensing*, vol. 26, pp. 65–74, Jan. 1988.
- [18] Ozan ARSLAN, Özer AKYÜREK, Sinasi KAYA, "A comparative analysis of classification methods for hyperspectral images generated with conventional dimension reduction methods", *Turk J Elec Eng & Comp Sci*(2017) 25: 58 – 72.
- [19] R.E. Roger, "Principal components transform with simple", automatic noise adjustment. *International Journal of Remote Sensing* 17 (1996) 2719-2727
- [20] J. B. Lee, A. S. Woodyatt, and M. Berman, "Enhancement of high spectral resolution remote sensing data by a noise-adjusted principal components transform," *IEEE Trans. Geosci. Remote Sensing*, vol. 28, pp. 295–304, May 1990.
- [21] R. E. Roger and J. F. Arnold, "Reliably estimating the noise in AVIRIS hyperspectral imagers," *Int. J. Remote Sens.*, vol. 17, no. 10, pp. 1951–1962, 1996.
- [22] Chein-I Chang, Qian Du, "Estimation of Number of Spectrally Distinct Signal Sources in Hyperspectral Imagery" *IEEE Transactions on Geoscience and Remote Sensing*, vol. 42, no. 3, march 2004.
- [23] R.J. Barnes, M.S. Dhanoa and S.J. Lister, "Standard Normal Variate Transformation and De-trending of Near-Infrared Diffuse Reflectance Spectra", *Appl. Spectrosc.* 43, 772-777, 1989.
- [24] C. Chang, C. Wu, C. Lo and M. Chang, "Real-Time Simplex Growing Algorithms for Hyperspectral Endmember Extraction", *IEEE Transactions on Geoscience and Remote Sensing*, vol. 48, no. 4, pp. 1834-1850, April 2010, doi: 10.1109/TGRS.2009.2034979.
- [25] Brian S. Penn, "Using simulated annealing to obtain optimal linear end-member mixtures of hyperspectral data ", *Computers & Geosciences* 28 (2002) 809–817.

- [26] Hideaki Kanayama, Te Ma, Satoru Tsuchikawaa and Tetsuya Inagaki, "Cognitive spectroscopy for wood species identification: near infrared hyperspectral imaging combined with convolutional neural networks", *Analyst*, 2019,144, 6438-6446.
- [27]<https://www.mathpages.com/home/kmath664/kmath664.htm>
- [28] Jiaojiao Wei, and Xiaofei Wang, "An Overview on Linear Unmixing of Hyperspectral Data", *Mathematical Problems in Engineering*, Volume 2020, Article ID 3735403, 12 pages. <https://doi.org/10.1155/2020/3735403>
- [29] Chein-I Chang, Chao-Cheng Wu, Wei-min Liu, and Yen-Chieh Ouyang, "A New Growing Method for Simplex-Based Endmember Extraction Algorithm", *IEEE Transactions on Geoscience and Remote Sensing*, VOL. 44, NO. 10, October 2006.
- [30] P. Honeine and C. Richard, "Geometric Unmixing of Large Hyperspectral Images: A Barycentric Coordinate Approach", *IEEE Transactions on Geoscience and Remote Sensing*, vol. 50, no. 6, pp. 2185-2195, June 2012, doi: 10.1109/TGRS.2011.2170999.
- [31] Xiong Xu, Xiaohua Tong, Antonio Plaza, Yanfei Zhong, Huan Xie and Liangpei Zhang, "Using Linear Spectral Unmixing for Subpixel Mapping of Hyperspectral Imagery: A Quantitative Assessment", *IEEE Journal on selected Topics in Applied Earth Observations and Remote Sensing*, VOL. 10, NO. 4, April 2017.
- [32]L. Zhang, L. Zhang, D. Tao, X. Huang and B. Du, "Hyperspectral Remote Sensing Image Subpixel Target Detection Based on SupervisedMetric Learning", *IEEE Transactions on Geoscience and Remote Sensing*, vol. 52, no. 8, pp. 4955-4965, Aug. 2014, doi: 10.1109/TGRS.2013.2286195.
- [33] P. Atkinson, *Innovations in GIS*. New York Taylor and Francis, 1997, vol. 4, ch. Mapping sub-pixel boundaries from remotely sensed images, pp. 166–180.
- [34] Resonon Inc. (Jan 17, 2019). *SpectrononPro Manual Release 5.3*. [Online]. Available: <http://bit.ly/2m6KWFn>
- [35] W. Sullivan Gaspar, "Análisis y procesamiento de imágenes hiperespectrales nir para detección de contaminantes plásticos", *Tesis de Maestría*, 2019.
- [36] USGS Spectral Library Version 7: U.S. Geological Survey Data Series 1035, 61 p., <https://doi.org/10.3133/ds1035>.

ANEXO

Programas en MATLAB

Identify

Código para el proceso de descomposición

```
function [Alpha,X,M_CgN]=Identify(Cf,CgN)
% ( M > N )
% Cf : Coeficientes (fila) de Fourier del espectro total representado por "f"
% CgN : Coeficientes de Fourier de los espectros base representados por
%      g1, g2, ..., gN (cada "g" en una fila)
N = length(CgN(:,1)); M = length(CgN(1,:));
% N : Numero de filas de CgN, numero de espectros base ( M > N )
% M : Numero de columnas de CgN, numero de coeficientes de Fourier
% M-N+1 : Numero de matrices cuadradas a formar dentro de la matriz
%        rectangular NxM
for l=1:(M-N+1)
for j=1:N % indice de filas para moverse a lo largo de la columna
for k=1:N % indice de columnas para moverse a lo largo de la fila
M_CgN(j,k,l) = CgN(k,j+1-1);
end
B_Cf(j,l) = Cf(j+1-1);
end
X(:,l) = linsolve(M_CgN(:,:,l),B_Cf(:,l));
end
% M_CgN : Matriz 3D donde la tercera dimension es (M-N+1)
% B_Cf : Matriz acumulando las columnas B para el linsolve
```

```
% X : Matriz que acumulará en columnas las soluciones para alpha
for i=1:length(X(:,1))
Alpha(i) = mean(X(i,:));
end
Alpha=Alpha(:);
end
```

PROGRAMA SALD escrito en R

```
library(hyperSpec)
library(EBImage)
library(spacetime)#
library(MASS)#ginv
#library(matlib)#inv
library(ppls)#orden
library(prospectr)#sg
library(wordspace)#normalizar
library(qlcMatrix)#rowMin
library("plot3D")
library("rgl")
library(stringr)
library(spectacles)
rm(list=ls())
#####
#LECTURA DE DATOS: res1
#####
#path<-"/11_11/"
path<-"/03_02/"
#File_name<-"8397_04_02"
#File_name<-"175149_04_02"
File_name<-"1691_05_02_3"
#File_name<-"161_03_02"
```



```
#File_name<-"013326_04_02"
#####
ntbase<-4
name<-c(1:ntbase)
#####
#name[1]<-"Calcite"
#name[2]<-"Dolomite"
#name[3]<-"Vermiculite"
#####
name[1]<-"Albite"
name[2]<-"Quartz"
name[3]<-"Orthoclase"
name[4]<-"Anorthoclase"
#####
#name[1]<-"Albite"
#name[2]<-"Quartz"
#name[3]<-"Orthoclase"
#name[4]<-"Pargasite"
#####
#Parámetros
#####
miggrupo<-5#espectros agrupados en endmembers
vd<-30#5,10,15,20,25,30
ciclos<-100#numero de ciclos
pruebas<-800#numero de pruebas
p1<-0.7#probabilidad de aceptar la peor solucion al inicio
p50<-0.05#probabilidad de aceptar la peor solucion al final
mm<-1#aproximaciones
npol<-5#grado del polinomio suavizado
Linicio<-950#inclusive
Lfinal<-1650#inclusive
ww<-21
#####
file<-paste(path,File_name, ".bil", sep="")
```

```

filehdr<-paste(File_name, ".hdr", sep="")
my_txt<- readLines(paste(path,filehdr, sep = ""),n = 5)
nr<-strtoi(gsub("[^[:digit:]]", "", my_txt[4]))
nc<-strtoi(gsub("[^[:digit:]]", "", my_txt[5]))
nw<-168#imagen inicial
#nr=lines;nc=samples
resonon=read.ENVI(file)
Im2<-as.matrix(resonon)
wl.data<-scan("./Base_de_datos/wavelength.txt",skip=1)
wavelength0<-c(wl.data[2:169])
#####
#PRETRATAMIENTO
#####
# eliminacion de bordes
# normalizado y centrado SNV
# suavizado
#####
#ELIMINACION DE BORDES
#####
#binicio=45 #inclusive
#bfinal=140#inclusive
binicio<-min(which(wavelength0>Linicio))
bfinal<-min(which(wavelength0>Lfinal))
newban1=(bfinal-binicio+1)
wavelength1<-wavelength0[c(binicio:bfinal)]
pixeles<-nr*nc
elers2<-pixeles*newban1
Im1.bordes<-1:elers2
dim(Im1.bordes) <- c(pixeles,newban1)
for ( k in 1:pixeles) {
for ( l in 1:newban1) {
m=1+binicio-1
Im1.bordes[k,l] <- Im2[k,m ]
}
}

```

```

}
#####
#Normalizado SNV
#####
Im1.snv<-Im1.bordes
#for(i in 1:pixeles){ Im1.snv[i,<-(Im1.bordes[i,]-mean(Im1.bordes[i,]))/(sd(Im1.bordes[i,]))}
for(i in 1:pixeles){ Im1.snv[i,<-snv(Im1.bordes[i,])}
#Im1.snv <- apply_spectra(Im1.bordes[1:pixeles,], snv)
matplot(wavelength1,t(Im1.snv[1:2,]),type='l',xlab="",ylab='Reflectance')
Im1.snvimg <- array (0 , dim = c(nr,nc,newban1))
Im1.snvimg [,,]<- Im1.snv[,]
#####
#Suavizado
#####
#ww<-21
newban<-newban1-ww+1;w1<-1+(ww-1)/2;w2<-1+newban1-w1
#npol<-5
matplot(wavelength1,t(Im1.snv[1:2,]),type='l',xlab="",ylab='Reflectance')
Im1<- savitzkyGolay(Im1.snv,0,npol,ww,delta.wav=2)
Im1.der <- savitzkyGolay(Im1.snv,1,npol,ww,delta.wav=2)
Im1.img <- array (0 , dim = c(nr,nc,newban))
Im1.img [,,]<- Im1[,]
wavelength<-wavelength1[c(w1:w2)]
#matplot(wavelength,t(Im1.der[1:2,]),type='l',xlab="",ylab='Reflectance')
#####
# Function for contrast
#####
contr <- function ( x ) {
out <- ( x - min ( x ) + 1 ) / ( max ( x ) - min ( x ) + 1 )
out
}
"%^%" <- function(x, n)
with(eigen(x), vectors %*% (values^n * t(vectors)))
#imagen promedio

```

```
#####
Im1_mean<-matrix(rowMeans(Im1),nr,nc)
image(Im1_mean,col=gray((0:256)/256))
#####
#MNF2 Ruido a partir de  $Z^{-1}$ 
#####
Im1.cov<-cov(Im1)
invIm1<-ginv(Im1.cov)
D<-1:eleres2
D <- matrix(0, newban, newban)
for(i in 1:newban){
D[i,i]<-1/invIm1[i,i]
}
D.cov<-cov(D)
invSN<-ginv(D.cov)
Imnf<-Im1.cov%*%invSN
Imnf.cov<-cov(Imnf)
Imnf.eval<-eigen(Imnf.cov)$values
Imnf.evec<-eigen(Imnf.cov)$vectors
tmnf<-Im1%*%Imnf.evec
imgtmnf<- array (0 , dim = c(nr,nc,newban))
imgtmnf[.,]<- tmnf[,]
#Display the 1 st to 20 bands
###display ( contr ( imgtmnf [ , , c (1:4)] ) , all = T , meth = 'r')
#####
#Eleccion de la dimension
#####
#vd=40
th<-vd+1 #pixeles endmembers
tmnf.vd<-tmnf[,c(1:th)]
endmember<-1:th
dim(endmember)<-th
#####
euclides<-1:pixeles
```

```
uno<-1:pixeles
dim(euclides)<-c(pixeles)
dim(uno)<-c(pixeles)
#####
#Primer endmember: Mayor módulo
#####
maxMod<-0
for(i in 1:pixeles){
vecPrueba<-tmnf[i,]
# maxPrueba<-vecPrueba%*%vecPrueba
maxPrueba<-sqrt(t(vecPrueba)%*%invIm1%*%vecPrueba)
if(maxPrueba>maxMod){
maxMod<-maxPrueba
endmember[1]<-i
}
}
cat("endmember ",1," ",endmember[1],"\n")
#####
#Segundo endmember: mayor distancia euclidiana
#####
maxDis<-0
for(i in 1:pixeles){
vecDis<-tmnf[i,]-tmnf[endmember[1],]
#disPrueba<-vecDis%*%vecDis
disPrueba<-sqrt(t(vecDis)%*%invIm1%*%vecDis)
if(disPrueba>maxDis){
maxDis<-disPrueba
endmember[2]<-i
}
}
cat("endmember ",2," ",endmember[2],"\n")
#####
# SGA : Otros endmembers
```

```

# NON square matrix: MNF reduce dimension->Square Matrix
#####
endmember[3:(vd+1)]<-0
for (j in 3:(vd+1)){
  Vn<-tmnf[endmember[1],1:j-1]
  for (n in 2:(j-1)){
    Vn<-cbind(Vn,tmnf[endmember[n],1:j-1])
  }
  Vtest.max<-0
  for(i in 1:pixeles){
    Vtest<-cbind(Vn,tmnf[i,1:(j-1)])
    first.row<-rep(1,j)
    Vtest<-rbind(first.row,Vtest)
    Vtest.Vol<-abs(det(Vtest)/factorial(j-1))
    if ((Vtest.Vol>Vtest.max)&!(i%in%endmember)){
      Vtest.max<-Vtest.Vol
      endmember[j]<-i
    }
  }
  cat("endmember ",j," ",endmember[j]," ",Vtest.max,"\n")
}
VPri<-cbind(Vn,tmnf.vd[endmember[vd+1],1:vd])
#####
#Abundancias
#####
#VE Matriz endmembers:vd+1
first.row<-rep(1,th)
VE<-rbind(first.row,VPri)
volE<-det(VE)#Factorial se cancela
alfa<-matrix( rep( 0, len=th*pixeles), nrow = pixeles)
for(i in 1:pixeles){
  for(j in 1:th){
    probe<-1:th
    probe[1]<-1
  }
}

```

```

probe[2:th]<-tmnf.vd[i,c(1:vd)]
VY<-VE
VY[,j]<-probe
volY<-det(VY)#factorial se cancela
alfa[i,j]<-volY/volE
if (alfa[i,j]<0) alfa[i,j]<-0#Fuera del simplex
}
}
abundancia1<-colSums(alfa)
abundancia1<-abundancia1/sum(abundancia1)
emab<-cbind(endmember,abundancia1)
#####
#banda m Vs banda n
#####
"
sx<-1
sy<-2
sz<-3
x1.Im1<-min(Im1.bordes[,sx])
x2.Im1<-max(Im1.bordes[,sx])
y1.Im1<-min(Im1.bordes[,sy])
y2.Im1<-max(Im1.bordes[,sy])
plot(Im1.bordes[,sx],Im1.bordes[,sy], pch=20, cex=0.2,col='red',
xlim=c(x1.Im1,x2.Im1),ylim=c(y1.Im1,y2.Im1),xlab="", ylab=")
par(new=TRUE)
plot(Im1.bordes[endmember,sx],Im1.bordes[endmember,sy],pch=20,col='blue',
xlim=c(x1.Im1,x2.Im1),ylim=c(y1.Im1,y2.Im1),xlab="", ylab=")
par(new=TRUE)
plot(Im1.bordes[endmember[1],sx],Im1.bordes[endmember[1],sy],type='p',
xlim=c(x1.Im1,x2.Im1),ylim=c(y1.Im1,y2.Im1),col='green',xlab="", ylab=")
#x<-Im1.bordes[endmember[1],]
#y<-Im1.bordes[endmember[4],]
#z<-Im1.bordes[endmember[19],]
#limites<-c(endmember[12],endmember[11], endmember[13])

```

```

x1.mnf<-min(tmnf[,sx])
x2.mnf<-max(tmnf[,sx])
y1.mnf<-min(tmnf[,sy])
y2.mnf<-max(tmnf[,sy])
z1.mnf<-min(tmnf[,sz])
z2.mnf<-max(tmnf[,sz])
plot (tmnf[,sx], tmnf[,sy], pch=20,cex=0.3, col = 'red',
xlim=c(x1.mnf,x2.mnf), ylim=c(y1.mnf,y2.mnf),
xlab='Normalized Intensity of e1',
ylab='Normalized Intensity of e2')
par(new=TRUE)
plot (tmnf[endmember,sx], tmnf[endmember,sy], pch=20, col = 'blue',
xlim=c(x1.mnf,x2.mnf), ylim=c(y1.mnf,y2.mnf),xlab=' ', ylab='')
par(new=TRUE)
plot(tmnf[endmember[1],sx],tmnf[endmember[1],sy],type='p',
xlim=c(x1.mnf,x2.mnf),ylim=c(y1.mnf,y2.mnf),col='green',xlab="", ylab=")
par(mfrow=c(1,2))

scatter3D(tmnf[,sx], tmnf[,sy], tmnf[,sz],
phi = 0, theta=10, pch=20, xlab="",
ylab="", zlab=","bty = 'f',
main='phi=0, Theta=10',
xlim = c(x1.mnf, x2.mnf),
ylim = c(y1.mnf, y2.mnf), zlim = c(z1.mnf, z2.mnf),
cex = 0.5,ticktype = 'detailed', nticks=5,col='blue',alpha=0.01)

scatter3D(tmnf[endmember,sx], tmnf[endmember,sy], tmnf[endmember,sz],
add = TRUE,size=0.5, pch=20, type = 'p',
ticktype = 'detailed',
cex=0.5, col='red')

scatter3D(tmnf[,sx], tmnf[,sy], tmnf[,sz],
phi = 40,theta=40, pch=50, xlab="",
ylab="", zlab=","bty = 'f',

```



```
main='phi=40, Theta=40',  
xlim = c(x1.mnf, x2.mnf),  
ylim = c(y1.mnf, y2.mnf), zlim = c(z1.mnf, z2.mnf),  
cex = 0.5,ticktype = 'detailed', nticks=5,col='blue',alpha=0.01)
```

```
scatter3D(tmnf[endmember,sx], tmnf[endmember,sy], tmnf[endmember,sz],  
add = TRUE,size=0.5, pch=20, type = 'p',  
ticktype = 'detailed',  
cex=0.5, col='red')
```

```
par(mfrow=c(1,2))
```

```
scatter3D(tmnf[,sx], tmnf[,sy], tmnf[,sz],  
phi = 40, theta=80, pch=20, xlab="",  
ylab="", zlab="",bty = 'f',  
main='phi=40, Theta=80',  
xlim = c(x1.mnf, x2.mnf),  
ylim = c(y1.mnf, y2.mnf), zlim = c(z1.mnf, z2.mnf),  
cex = 0.5,ticktype = 'detailed', nticks=5,col='blue',alpha=0.01)
```

```
scatter3D(tmnf[endmember,sx], tmnf[endmember,sy], tmnf[endmember,sz],  
add = TRUE,size=0.5, pch=20, type = 'p',  
ticktype = 'detailed',  
cex=0.5, col='red')
```

```
scatter3D(tmnf[,sx], tmnf[,sy], tmnf[,sz],  
phi = 40,theta=120, pch=50,  
xlab="",ylab="",zlab="",bty = 'f',  
main='phi=40, Theta=120',  
xlim = c(x1.mnf, x2.mnf),  
ylim = c(y1.mnf, y2.mnf), zlim = c(z1.mnf, z2.mnf),  
cex = 0.5,ticktype = 'detailed', nticks=5,col='blue',alpha=0.05)
```

```
scatter3D(tmnf[endmember,sx], tmnf[endmember,sy], tmnf[endmember,sz],
```

```

add = TRUE,size=0.5, pch=20, type = 'p',
ticktype = 'detailed',
cex=0.5, col='red')
"

#colvar = NULL
#plot3d(Im1.bordes[,1], Im1.bordes[,2], Im1.bordes[,3], type="s", size=0.4,
#   lit=TRUE,cex=0.3,xlab="MNF1", ylab="MNF2",zlab="MNF3",
#   cex.axis=0.5)
#plot3d(x, y, z, type="s", size=1,
#   lit=TRUE, main = "bandas 1,2,6",sub="3-D Plot",
#   xlab="MNF10", ylab="MNF50",zlab="MNF100")
#####
#Píxeles de los endmembers sobre imagen promedio y espectros
#####
#par(mfrow=c(1,2))
yc<-1.0-seq(0,1,length.out =nr)
xc<-seq(0,1,length.out =nc)
col_set <- rainbow(th)
image(Im1_mean,axes="FALSE",main=file,col=gray((0:256)/256))
row_n<-1:th;col_n<-1:th
dim(row_n)<-c(th);dim(col_n)<-c(th)
for (i in 1:th){
row_n[i]<-(endmember[i]%/%nc +1)
col_n[i]<-(endmember[i]%nc)
if (col_n[i]==0) {row_n[i]<-row_n[i]-1;col_n[i]<-nc}
}
points(xc[col_n],yc[row_n],pch=20, lwd=2, cex=1,col=col_set)
matplot(wavelength1,t(Im1.bordes[endmember[1:th],]),type='l',
xlab='Longitud de onda(nm)',ylab='Reflectance',col=col_set)
#matplot(wavelength,t(Im1.nc[endmember[1:th],]),type='l',
#   xlab='Longitud de onda(nm)',ylab='Reflectance',col=col_set)
#legend("left", lty=c(1,1) , lwd=c(2.5,2.5), col = col_set)
#write.table(maxproy, file="Max.txt")
#write.table(minproy, file="Min.txt")

```

```
#####
#Ajuste al rango de trabajo
#####
alineas<-function(data.s,wl,n){
r<-1:n;dim(r)<-c(n)
y<-1:n;dim(y)<-c(n)
m<-(data.s[n]-data.s[1])/(wl[n]-wl[1])
b<-data.s[1]-m*wl[1]
y<-m*wl+b
# r<-(data.s-y)*((1+m^2)^(-0.5))#rcos(theta)
r<-(data.s-y)
cero<-mean(r)
r<-r-cero
out<-r
out
}
#####
#Grupos de pixeles por Endmember
#####
#migruo<-
Im1.bordesA<-Im1.bordes
for(i in 1:pixeles){
Im1.bordesA[i,<-alineas(Im1.snv[i,],wavelength1,newban1)
}
matplot(wavelength1,t(Im1.bordesA[endmember[1:th],]),type='l',
xlab='Longitud de onda(nm)',ylab='Reflectance',col=col_set)
for(i in 1:th){
em.group<-cbind((1:pixeles),rowSums((Im1.bordesA-Im1.bordesA[endmember[i,])^2))
em.group.ord<-em.group[order(em.group[,2],decreasing = FALSE),]
em.pix.total<-c(em.group.ord[,1])
em.pix<-em.pix.total[c(1:migruo)]
espectro<-colMeans(Im1.bordes[em.pix[1:migruo],])
if (i!=1){
em.spectrat<-cbind(em.spectrat,espectro)}
}
```

```

else{em.spectrat<-espectro}
}
em.spectra<-t(em.spectrat)
em.spectra.sua <- savitzkyGolay(em.spectra,0,npol,ww,delta.wav=2)
em.spectra.der <- savitzkyGolay(em.spectra,1,npol,ww,delta.wav=2)
em.spectra.der.sua <- savitzkyGolay(em.spectra.der,0,npol,ww,delta.wav=2)
#matplot(wavelength1,em.spectra[1,],ylim=c(0.67, 0.69),lwd=2,
# type='l',xlab='Wavelength (nm)',ylab='Reflectance (a.u)',cex.lab=1.3)
#matplot(wavelength,em.spectra.sua[1,],ylim=c(0.67, 0.69),lwd=2,
# type='l',xlab='Wavelength (nm)',ylab='Reflectance (a.u)',cex.lab=1.3)
#####
#BASE DE DATOS 1
#####
#elementos
#ntbase<-4
#name<-c(1:10)
el.name<-1:ntbase
dim(el.name)<-ntbase
el.name.der<-1:ntbase
dim(el.name.der)<-ntbase
for(i in 1:ntbase){
el.name[i]<-paste("./Base_de_datos/",name[i],".txt",sep="")
el.name.der[i]<-paste("./Base_de_datos/",name[i],"der.txt",sep="")}
#####
el.w<-matrix(rep( 0, len=(ntbase*newban)), nrow = ntbase)
el.r<-matrix(rep( 0, len=(ntbase*newban)), nrow = ntbase)
el.rA<-matrix(rep( 0, len=(ntbase*newban)), nrow = ntbase)
el.r.der<-matrix( rep( 0, len=(ntbase*newban)), nrow = ntbase)
nini<-binicio+w1-1
nini.der<-binicio+10
for (i in 1:ntbase){
el.data<-scan(el.name[i],skip=1)
el.data.der<-scan(el.name.der[i],skip=1)
el.w[i,1:newban]<-wavelength

```

```

inidat<-nini
findat<-newban+inidat-1
el.r[i,1:newban]<-el.data[c(inidat:findat)]
el.r.der[i,1:newban]<-el.data.der[c(inidat:findat)]
}

for(i in 1:ntbase){
el.rA[i,]<-alineael(el.r[i,],wavelength,newban)
}
em.spectraA.sua<-1:(th*newban)
dim(em.spectraA.sua)<-c(th,newban)
for (i in 1:th){
em.spectraA.sua[i,]<-alineael(em.spectra.sua[i,],wavelength,newban)
}
matplot(wavelength,em.spectraA.sua[6,],ylim=c(-0.01, 0.01),lwd=2,
type='l',xlab='Wavelength (nm)',ylab='Reflectance (a.u)',cex.lab=1.3)
#####
#em.spectraA.sua<-em.spectraA.sua
#####
par(mfrow=c(1,1))
matplot(wavelength,t(el.r[1:ntbase,]),ylim=c(0.4, 1),col = c("red", "blue", "green","black"),
type='l',lwd=2, xlab='Wavelength (nm)',ylab='Reflectance',cex.lab=1.3)
legend("topright", legend = c(name),
col = c("red", "blue", "green"), lty = 1:ntbase, cex = 0.8,bty = "n")
#####
#Simulated annealing
#Píxeles puros ajustados con la base de datos
#####
#valor máximo en el ajuste
#####
esp.max<-max(abs(em.spectraA.sua))
base.max<-max(abs(el.rA))
por.max<-esp.max/base.max

```

```
matplot(wavelength,t(el.rA[1:ntbase,]),ylim=c(-base.max, base.max),col = c("red", "blue",
"green"),
type='l',lwd=2, xlab='Wavelength (nm)',ylab='Normalized Reflectance (a.u.)',cex.lab=1.3)
legend("topright", legend = c(name),
col = c("red", "blue", "green","black"), lty = 1:ntbase, cex = 0.8,bty = "n")
#####
base.s<-t(el.rA)
ajustar<-em.spectraA.sua
#base.s<-t(el.r)
#ajustar<-em.spectraA.sua
base<-base.s
#####
pe<-1:(ntbase*th)
pj<-c(1:ntbase)
pk<-c(1:ntbase)
pef<-1:(ntbase*th)
p_nor<-1:(ntbase*th)
dim(pe)<-c(ntbase,th)
dim(pef)<-c(ntbase,th)
dim(p_nor)<-c(ntbase,th)
#ciclos<-500#numero de ciclos
#pruebas<-10#numero de pruebas
#p1<-0.7#probabilidad de aceptar la peor solucion al inicio
#p50<-0.000005#probabilidad de aceptar la peor solucion al final
t1<- -1.0/log(p1)#temperatura inicial
t50<- -1.0/log(p50)#temperatura final
frac<- (t50/t1)^(1/(ciclos-1))#Reduccion de temperatura en cada ciclo
for (l in 1:th){
u<-ajustar[l,]
nacep<-0#soluciones aceptadas
pe[,l]<-0.1#parametros iniciales
t<-t1
DeltaE<-0#Promedio de energia
DeltaE_avg<-0#Promedio de energia
```

```

for (p in 1:ntbase){
  pef[p,l]<-1}
for(m in 1:mm){
  for (i in 1:ciclos){
  for (j in 1:pruebas){
  omega1<-c(u-base%*%pe[,l])
  for (p in 1:ntbase){
  pj[p]<-pe[p,l]
  pk[p]<-runif(1,0 ,2)
  }
  for (p in 1:ntbase){
  pe[p,l]<-pk[p]}
  omega2<-c(u-base%*%pe[,l])
  SumSpec1<-omega1%*%omega1
  SumSpec2<-omega2%*%omega2
  dif_prueba<-(SumSpec2-SumSpec1)
  DeltaE<-abs(dif_prueba)
  if (SumSpec2>SumSpec1){
  if (i==1 && j==1){DeltaE_avg<-DeltaE}
  prob<-exp(-DeltaE/(DeltaE_avg*t))
  dado<-runif(1,0,1)
  if (dado>prob) {
  for (p in 1:ntbase){
  pe[p,l]<-pj[p]}
  }
  }
  }
  nacep<-nacep+1
  DeltaE_avg<- (DeltaE_avg * (nacep-1.0) + DeltaE) / nacep
  t<-frac*t
  }
  }
  for (p in 1:ntbase){
  pef[p,l]<-pe[p,l]}

```

```

p_nor[,l]<-pe[,l]/sum(pe[,l])
cat("t=",t," endmember: ",l, " ci: ",p_nor[,l] ,"\n")
}

#####
VDS<-round(vd/5)
rms_avg_VD<-1:VDS
dim(rms_avg_VD)<-c(VDS)
ajustar<-ajustar
for (j in 1:VDS){
thh<-j*5
rms<-0
for (i in 1:thh){
ajustad[i,]<-base.s%*%pef[,i]
rms<-rms+sqrt(sum((ajustar[i,]-ajustad[i,])^2)/newban)
}
rms_avg<-rms/thh
rms_avg_VD[j]<-rms_avg
}
plot(wavelength,t(ajustar[6,]),ylim=c(-esp.max,esp.max),
type='l',xlab='Wavelength (nm)', lty=1,
ylab=expression(paste("Normalized reflectance (a.u.)")),cex.lab=1.3)
lines(wavelength,ajustad[6,],ylim=c(-esp.max,esp.max),lty=2,
type='l',col="red",cex.lab=1.3)
legend("topright", legend = c("Endmember 6", "Adjusted Endmember"),
col = c("black","red"), lty=1:2,cex = 0.8,bty = "n")
#####
#Abundancias sobre la base
#####
abundancia<-1:(ntbase*VDS)
dim(abundancia)<-c(VDS, ntbase)
abundancia.var<-1:th
dim(abundancia.var)<-c(th)
abundancia.var<-c(1:th)*0

```



```

for(i in 1:VDS){
thh<-i*5
for (j in 1:thh){
abundancia.var[j]<-abundancia1[j]
}
abundancia[i,<-abundancia.var%*%t(p_nor)
abundancia[i,<-abundancia[i,]/sum(abundancia[i,])
}
#abundancia<-abundancia1%*%t(p_nor)
#abundancia<-abundancia/sum(abundancia)
otros<-c(Linicio,Lfinal,migrupo,ciclos,pruebas,m,p1,p50)
time<-Sys.time()
hora<-as.character(time)
file_out<-paste("out_",hora,".txt",sep="")
cat(file=file_out, "FILE:", File_name, sep = " ", "\n")
cat(file=file_out, hora, append = TRUE, "\n")
write("*****", file = file_out, append = TRUE)
write("Endmembers", file = file_out, append = TRUE)
write("*****", file = file_out, append = TRUE)
write.table(emab, file = file_out, col.names=F, append = TRUE)
write.table(t(p_nor), file = file_out, col.names=F, append = TRUE, sep = " ")
write("*****", file = file_out, append = TRUE)
write("Eigenvalues", file = file_out, append = TRUE)
write("*****", file = file_out, append = TRUE)
write.table(Imnf.eval, file = file_out, col.names=F, append = TRUE)
write("abundancias", file = file_out, append = TRUE)
write("*****", file = file_out, append = TRUE)
for(i in 1:VDS){
reporte<-t(rbind("vd=",i*5,name,round(abundancia[i,]*100, digits=2)))
write.table(reporte, file = file_out, row.names=F, col.names=F, append = TRUE ,
quote=FALSE)
}
write("*****", file =
file_out, append = TRUE)

```

```
write("rms, inicio, bfinal, migrupa, ciclos, pruebas, m, p1, p50", file = file_out, append =
TRUE)
write("*****", file =
file_out,append = TRUE)
write(rms_avg, file = file_out,append = TRUE, sep = " ")
write(otros, file = file_out,append = TRUE, sep = " ")
```