

Nivel 2: Herramientas de teledetección cuantitativa
*Guía de actividades: Uso del suelo en el departamento de
Iguazú, provincia de Misiones*

Francisco Nemiña¹

*Unidad de Educacion y Formacion Masiva
Comisión Nacional de Actividades Espaciales*

2017-1

¹fnemina@conae.gov.ar

Índice general

| | |
|--|-----------|
| Introducción | v |
| 0.1. Organización del curso | v |
| 0.1.1. Forma de aprobación | vi |
| 0.2. Material del curso | vi |
| | |
| I Variables continuas | 1 |
| 1. Reflectancia y espacio espectral | 3 |
| 1.1. Exploración de imágenes con el QGIS | 3 |
| 1.2. Creación de capas vectoriales | 5 |
| 1.3. Exploración raster en R | 6 |
| 1.4. Manejo vectorial en R | 8 |
| | |
| 2. Corrección radiométrica | 13 |
| 2.1. Cálculo de reflectancia a tope de la atmósfera | 13 |
| 2.2. Cálculo de reflectancia corregida atmosféricamente por métodos estadísticos | 15 |
| 2.3. 6S | 17 |
| | |
| 3. Cálculo de índices espectrales | 19 |
| 3.1. Cálculo de índices entre bandas | 19 |
| 3.2. Cálculo de la línea de suelo | 21 |
| 3.3. Estimación de parámetros biofísicos | 22 |
| | |
| 4. Rotaciones espectrales | 25 |
| 4.1. Cálculo de la transformada tasseled cap para una imagen landsat | 25 |
| 4.2. Cálculo de la transformada por componentes principales | 26 |
| 4.3. Algunas ideas sobre series temporales | 27 |
| | |
| II Variables discretas | 29 |
| | |
| 5. Clasificación no supervisada | 31 |
| 5.1. Clasificación mediante el método k-means | 31 |
| 5.2. Información espacial y temporal | 34 |
| | |
| 6. Clasificación supervisada | 37 |
| 6.1. Clasificación por máxima verosimilitud | 37 |
| 6.2. Entropía de la clasificación | 39 |
| | |
| 7. Técnicas pos-clasificación | 41 |
| 7.1. Filtrado de clasificaciones | 41 |
| 7.2. Validación de la clasificación | 42 |

| | |
|---|-----------|
| III Apéndice | 47 |
| A. Cronograma | 49 |
| B. Categorías de uso y cobertura del suelo | 51 |
| C. Instalacion de qgis y R | 53 |
| C.1. Instalación en Windows 10 | 53 |
| C.2. Instalación GNU/Linux | 53 |
| C.3. Máquina virtual | 54 |

Introducción

La utilización de imágenes satelitales permite analizar grandes extensiones del territorio, contando con un registro histórico con el cual realizar comparaciones.

En la provincia de Misiones, el departamento de Iguazú es lindante a Brasil y Paraguay siendo parte de la zona conocida como triple frontera perteneciente a la ecorregión conocida como *selva paranaense* (Figura 1). Allí podemos encontrar Represa de Urugua-í y el Parque Nacional Iguazú. El departamento tiene un área de $2,736 km^2$ y una población de 82,227 según el último censo del INDEC.

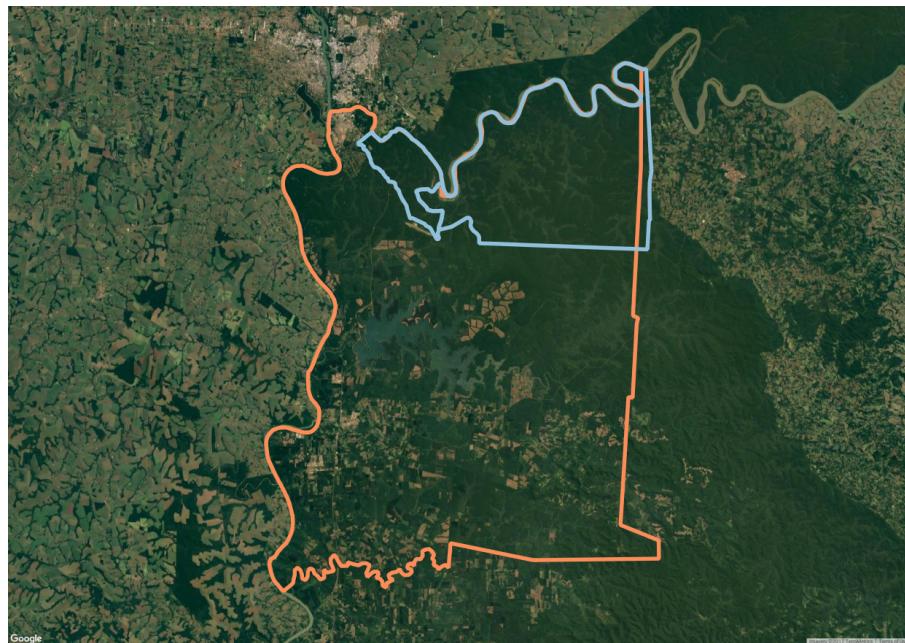


Figura 1 – Departamento de Iguazú, en rosa, y parque natural Iguazú, en celeste.

Tomaremos al departamento como área de estudio durante este curso con el objetivo de obtener un mapa de uso y cobertura que nos permita estimar y validar sus superficies.

Utilizaremos imágenes de los satélites Landsat 8, Landsat 7 y el producto de MOD13Q1 obtenido de los satélites TERRA y AQUA durante el periodo 2000-2016.

0.1. Organización del curso

El curso se divide en dos partes. En la primera trabajaremos con la compresión del espacio espectral y el uso de las imágenes satelitales para extraer valores continuos de las variables biofísicas.

Capítulo 1, **Reflectancia y espacio espectral**, estudiaremos las firmas espectrales de distintas

coberturas y como se relacionan con las propiedades biofísicas. Introduciremos el concepto de espaciopectral como el lugar natural donde realizar el análisis en teledetección.

Capítulo 2, **Corrección radiométrica**, estudiaremos distintas formas de obtener la reflectancia de las coberturas a partir de los datos satelitales. Analizaremos métodos de corrección atmosférica basados en propiedades estadísticas de las imágenes y en el modelado de la atmósfera.

Capítulo 3, **Cálculo de índices espectrales**, veremos como a partir de los valores de reflectancia para una cobertura y operaciones matemáticas, podemos obtener los valores de variables biofísicas continuas como son el contenido de clorofila o humedad.

Capítulo 4, **Rotaciones espectrales**, empezaremos a utilizar herramientas geométricas en el espaciopectral para resaltar distintas propiedades de las imágenes y poner en evidencia cuales son las zonas del espectro que más información aportan sobre nuestra zona de estudio.

En la segunda parte del curso trabajaremos más en detalle con el espaciopectral usando sus propiedades para extraer información categórica de las imágenes.

Capítulo 5, **Clasificación no supervisada**, veremos como utilizar herramientas que no requieren de conocimiento previo del área de estudio para realizar segmentaciones en el espacio de fases de la imagen y obtener mapas de uso y cobertura.

Capítulo 6, **Clasificación supervisada**, veremos otros métodos para extraer información categórica sobre las imágenes satelitales al estudiar distintas formas de clasificación supervisada.

Capítulo 7, **Técnicas pos-clasificación**, veremos algunas técnicas de postprocesamiento que nos permitirán analizar el contexto espacial de nuestras clasificaciones y calcular superficies de coberturas para las imágenes y su correspondiente incertezza.

Puede consultar el cronograma detallado en el apéndice A.

0.1.1. Forma de aprobación

Para aprobar el curso se deben reunir al menos 100 puntos entre las distintas actividades. Además deberán completarse dos cuestionarios *obligatorios* al comenzar y finalizar el curso.

La nota final del curso estará dada por el siguiente rango

- 0 - 99 - No aprobó
- 100-109 - Seis
- 110-129 - Siete
- 130-169 - Ocho
- 170-189 - Nueve
- 190-200 - Diez

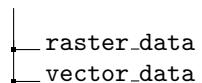
Los puntos se determinan de la siguiente manera

- Cada cuestionario: 0 a 10 puntos. Máximo 70.
- Cada tarea: 0 a 50 puntos. Máximo 100.
- Participar en la plataforma: 0 a 10 puntos. Sin máximo.

0.2. Material del curso

Todo el material del curso se encuentra disponible para descargar en el aula virtual del curso. Está organizado de la siguiente manera

```
material
└── aux_data
└── example_scripts
```



conteniendo la carpeta `aux_data` archivos adicionales para trabajar, `example_scripts` todos los scripts que se encuentran en esta guia, `raster_data` los archivos raster cada uno en una carpeta y `vector_data` los archivos vectoriales para usar durante el curso.

Todos los materiales del curso, con sus correspondientes editables, pueden encontrarse en el repositorio de github <https://github.com/fnemina/curso-sopi-herramientas-cuantitativas>.

Parte I

Variables continuas

Capítulo 1

Reflectancia y espacio espectral

En esta primera práctica nos familiarizaremos con las interfaces gráficas del QGIS y de R-studio analizando la imagen Landsat 8 de noviembre de 2016, desde el punto de vista espectral. Son nuestros objetivos:

- Abrir una imagen en QGIS.
- Crear archivos vectoriales y digitalizar coberturas en QGIS.
- Abrir un archivo raster y vectorial en R.
- Realizar un análisis estadístico de la imagen y de las distintas coberturas digitalizadas en R.

1.1. Exploración de imágenes con el QGIS

Comenzamos abriendo la imagen LC82240782016304LGN00.vrt que se encuentra en la carpeta `raster_data/LC82240782016304`, la cual corresponde al departamento de Iguazú en la provincia de Misiones. Fue obtenida por el satélite Landsat 8 durante el mes de noviembre de 2016.

En el menú *Capa → Añadir capa → Añadir capa ráster*. Navegamos hasta la carpeta `raster_data/LC8224078201630` y abrimos la imagen LC82240782016304LGN00.vrt. La encontraremos en el *Panel de capas* de QGIS. Podemos usar las herramientas para movernos en la imagen (Figura 1.1).



Figura 1.1 – Herramientas para moverse dentro de la imagen. De izquierda a derecha: 1. Desplazar mapa, 2. Desplazar mapa a la selección, 3. Acerca zoom, 4. Alejar zoom, 5. zoom a la resolución nativa, 6. zoom general, 7. zoom a la selección, 8. zoom a la capa, 9. zoom anterior, 10. zoom siguiente, 11. Actualizar.

Para realizar cambios en la visualización y explorar las propiedades de una capa, hacemos click derecho sobre ella y luego seleccionamos la opción *Propiedades*. Allí podemos ir a la pestaña *General* para ver datos como el nombre de la capa¹, la cantidad de filas y columnas del archivo, el valor digital no válido, el sistema de coordenadas entre otros (Figura 1.2).

En la pestaña *Estilo* podemos cambiar la visualización de la capa. Allí elegimos de qué color mostrar cada una de las bandas. Para cambiar el realce hacemos click en el botón *Cargar* para seleccionar los valores máximos y mínimos (Figura 1.3).

¹Es un buen momento para ponerle uno más sencillo

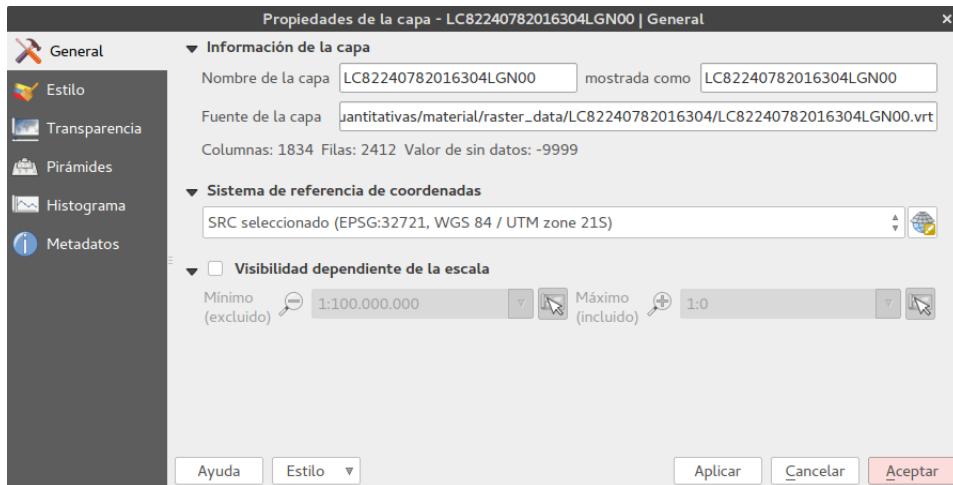


Figura 1.2 – Pestaña general de propiedades de una capa. Podemos ver los datos más importantes como la cantidad de filas y columnas, el nombre y el sistema de referencia.

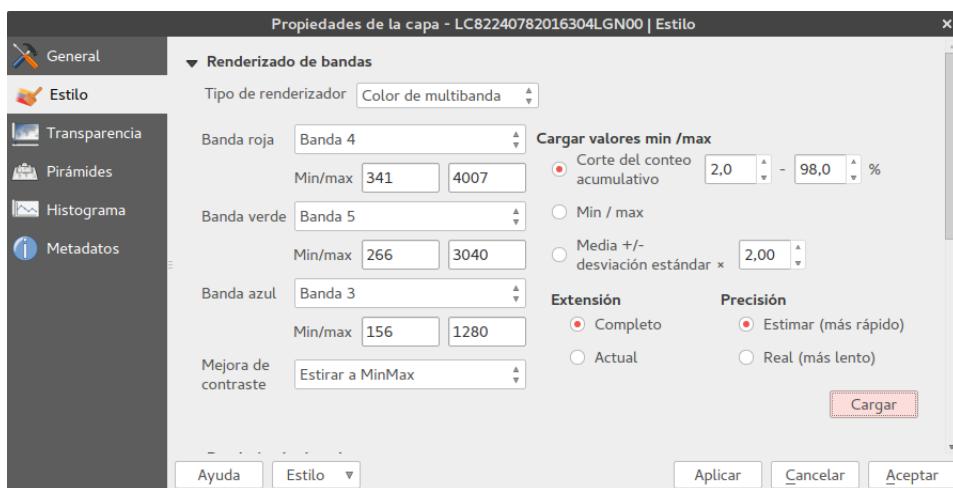


Figura 1.3 – Estilos de visualización de una capa raster.

La herramienta *Identificar un objeto espacial* nos permite extraer valores de la imagen. Al habilitarla veremos datos como el valor de reflectancia del píxel seleccionado que pueden mostrarse como Árbol, Tabla o Grafo según uno deseé (Figura 1.4).

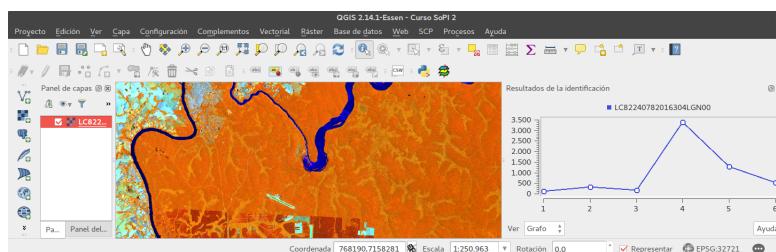


Figura 1.4 – Identificación de un píxel correspondiente a la selva paranaense mostrada como grafo.

Actividad 1.1.1. Cambie la combinación de bandas de la imagen Landsat 8 a color real y explórela. Identifique zonas de coberturas uniformes. Pruebe cambiar de combinación de bandas y decida si las zonas siguen siendo uniformes después de cada cambio.

Actividad 1.1.2. Encuentre el sistema de coordenadas en el cual se obtenga la imagen. ¿Cuántas filas y columnas tiene?

Actividad 1.1.3. Utilizando la herramienta identificar objetos espaciales encuentre los valores de reflectancia de distintas coberturas. Grafique estos valores en función de la longitud de onda y en el espacio espectral.

1.2. Creación de capas vectoriales

La capas vectoriales nos será de utilidad en este curso, para extraer datos cuantitativos de las capas raster.

Con la herramienta *nueva capa de archivo shape* es posible crear una nueva capa vectorial. Para esto hacemos click en el botón que se encuentra en el panel lateral. Podemos agregar los campos que sean necesarios para nuestra capa vectorial. En este caso, serán los campos: MC_ID, como entero de longitud 1 y Comment, como texto de 80 caracteres. Elegimos el sistema de coordenadas correspondiente a la imagen anterior, guardamos en la carpeta `vector_data/`, con el nombre `firmas.shp` (Figura 1.5).

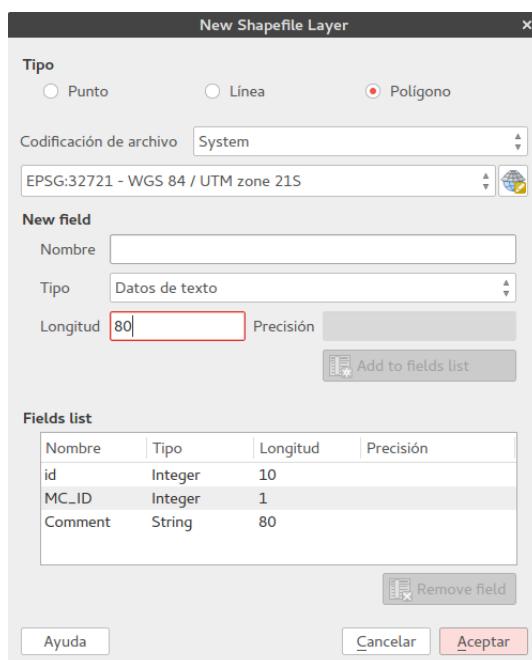


Figura 1.5 – Creación de una nueva capa vectorial.

Una vez creada, utilizamos la barra de herramientas de QGIS, (Figura 1.6), para agregarle geometrías. Para esto hacemos click en el botón de agregar geometría y digitalizamos una zona uniforme dentro de la imagen.

Al terminar, QGIS pedirá un número de ID para la capa que debe ser correlativo. Además podremos ingresar en este momento los valores del resto de los campos de nuestro objeto espacial (Figura 1.7).

Es importante recordar que debemos estar en el modo de edición para poder hacerlo. Al terminar debemos desactivar esa opción.



Figura 1.6 – Herramientas de edición vectorial. De izquierda a derecha: 1. Comutar edición, 2. Guardar cambios a la capa, 3. Añadir objeto espacial, 4. Añadir cadena circular, 5. Mover objeto espacial, 6. Herramienta de nodos, 7. Borrar lo seleccionado, 8. Cortar objetos espaciales, 9. Copiar objetos espaciales, 10. Pegar objetos espaciales.

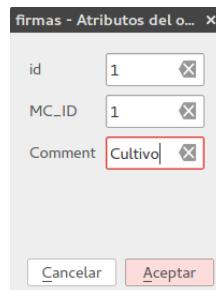


Figura 1.7 – Valores de los campos del nuevo polígono creado.

Actividad 1.2.1. Digitalice coberturas uniformes dentro de la imagen. Recuerde obtener al menos un polígono por cada categoría de uso y cobertura presente.

En caso de necesitar cambiar la visualización de la capa vectorial, podemos entrar a sus propiedades². Podemos acceder a la tabla de datos de la capa vectorial haciendo click derecho sobre ella y eligiendo la opción *Abrir tabla de atributos*.

1.3. Exploración raster en R

Veamos como abrir y trabajar con las imágenes satelitales en R. La forma de realizar operaciones es escribir comandos en la consola de R-studio y ejecutarlos presionando enter. Para trabajar con imágenes satelitales debemos utilizar algunas librerías adicionales. Para cargarlas usamos el comando `library(raster)`. De esta forma agregamos funciones que nos facilitaran el trabajo raster.

Además, deberemos situar nuestra carpeta de trabajo donde se encuentran las carpetas que descargamos. Para esto nos movemos en el explorador de archivos hasta ella y hacemos click en usar la carpeta como carpeta de trabajo (Figura 1.8).

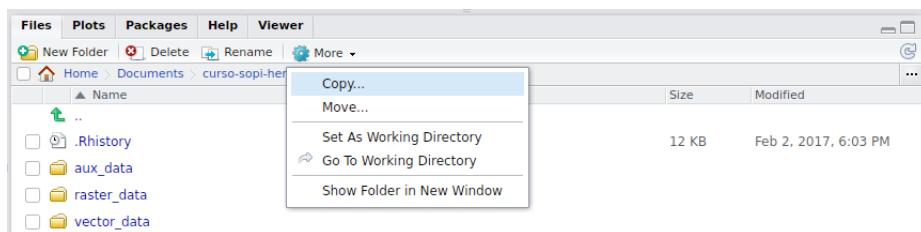


Figura 1.8 – Configuración del directorio de trabajo desde la interfaz gráfica.

También podemos utilizar el comando `setwd(.)` para configurar el directorio de trabajo. En este caso hay que especificar la ruta completa hasta él.

²Puedes utilizar el estilo precargado ubicado en la carpeta `aux_data`

Una vez en la carpeta, existen varias maneras de abrir una imagen. Con el comando `raster`, abrimos una única banda; `brick`, abrimos un archivo multibanda, ;y `stack`, abrimos distintas bandas por separado. Veamos algunos ejemplo:

Ejemplo 1.3.1. Abrimos la imagen completa del archivo de Landsat 8 y consultamos sus propiedades.

```
1 ref.2016 <- brick("raster_data/LC82240782016304/LC82240782016304LGN00.vrt")
2 ref.2016
```

obtenemos de resultado el siguiente texto

```
class      : RasterBrick
dimensions : 2412, 1834, 4423608, 6 (nrow, ncol, ncell, nlayers)
resolution : 30.00402, 30.00265 (x, y)
extent     : 731118.6, 786146, 7101531, 7173897 (xmin, xmax, ymin, ymax)
coord. ref. : +proj=utm +zone=21 +south +datum=WGS84 +units=m +no_defs
              +ellps=WGS84 +towgs84=0,0,0
data source: ./material/raster_data/LC82240782016304/LC82240782016304LGN00.vrt
names      : LC82240782016304LGN00.1, LC82240782016304LGN00.2, ...
min values : -33, 192, ...
max values : 2774, 3265, ...
```

En él, podemos ver la clase a la que corresponde el archivo, en este caso un *RasterBrick*, las dimensiones, el tamaño de píxel, extensión de la capa, proyección, cual es la ruta al archivo, las bandas y sus valores máximos y mínimos.

Cambiemos el nombre a las bandas y la convertimos a reflectancia entre 0 y 1.

```
1 names(ref.2016) <- c("blue", "green", "red", "nir", "swirl1", "swirl2")
2 ref.2016 <- ref.2016/1e4
3 rasterOptions(addheader = "ENVI")
4 writeRaster(ref.2016, "raster\_data/processed/ref2016")
```

Analicemos el código linea por linea.

- La primera abre la imagen como un raster de múltiples bandas.
- La segunda, cambia los nombres de cada banda a los que figuran en la lista entre paréntesis. Es importante resaltar que el número de nombres debe ser el mismo que el de bandas.
- En tercer lugar, convertimos el archivo de números enteros entre 0 y 10000 a valores entre 0 y 1.
- La cuarta linea es necesaria correrla una sola vez por sesión. La misma agrega el header de ENVI a nuestro output para poder abrir el archivo desde QGIS
- La quinta linea guarda el archivo raster con el nombre `ref2016` . En este caso estamos usando el formato nativo de R.

Podemos graficar una combinacion de bandas con el comando `plotRGB(ref.2016,r=4,g=5,b=3,stretch='lin')`(Figura 1.9)

Para graficar las bandas por separador hacemos `plotRGB(ref.2016)` (Figura 1.10).

Actividad 1.3.1. Abra el archivo guardado en QGIS y vuelva a mirar la firma espectral para distintas coberturas. ¿Entre que valores se encuentra ahora?

Veamos como trabajar más en detalle con los valores de nuestra imagen.

Ejemplo 1.3.2. Hagamos un análisis estadístico de la imagen ejecutando el comando `summary(ref.2016)`



Figura 1.9 – Combinación de bandas nir-swir1-red en R.

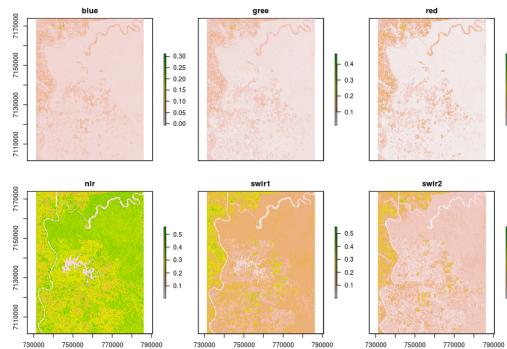


Figura 1.10 – Gráfico de bandas con realce automático para cada una.

| | blue | gree | red | nir | swir1 | swir2 |
|---------|---------|--------|--------|---------|---------|---------|
| Min. | -0.0278 | 0.0000 | 0.0000 | -0.0128 | -0.0069 | -0.0038 |
| 1st Qu. | 0.0128 | 0.0328 | 0.0184 | 0.2763 | 0.1198 | 0.0493 |
| Median | 0.0138 | 0.0362 | 0.0203 | 0.3287 | 0.1365 | 0.0572 |
| 3rd Qu. | 0.0170 | 0.0450 | 0.0329 | 0.3557 | 0.1644 | 0.0749 |
| Max. | 0.5548 | 0.8257 | 0.8034 | 0.7542 | 0.9181 | 0.9446 |
| NA's | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 | 0.0000 |

Calculamos los histogramas de todas las bandas con el comando `hist(ref.2016)` y el scatter plot entre dos bandas como `plot(ref.2016$red, ref.2016$nir)`.

En caso de querer todos los scatterplots e histogramas en un solo gráfico usamos el comando `pairs(ref.2016)`.

1.4. Manejo vectorial en R

Hasta ahora estamos analizando la imagen completa, pero podemos analizar determinadas zonas utilizando un archivo vectorial.. También será posible muestrear la imagen usando otro raster, pero lo veremos más adelante.

Para trabajar con vectores en R utilizaremos la librería `library(rgdal)`.

Ejemplo 1.4.1. Veamos como realizar el análisis básico de un vector en R. Comenzamos leyéndolo

```
1  firmas <- readOGR(dsn="vector\data\", layer="firmas")
```

Notamos en este caso que debemos indicar por separado la carpeta que contiene al shapefile en *dsn* y el nombre de la capa que queremos abrir como *layer*.

Podemos mostrar las propiedades del vector llamando a la variable **firmas** obteniendo

```
class      : SpatialPolygonsDataFrame
features   : 8
extent     : 738692.8, 767774.6, 7133396, 7165265 (xmin, xmax, ymin, ymax)
coord. ref.: +proj=utm +zone=21 +south +datum=WGS84 +units=m +no_defs
             +ellps=WGS84 +towgs84=0,0,0
variables  : 3
names      : id, MC_ID,      Comment
min values : 0,      1,      Alto
max values : 9,      8, Suelo desnudo
```

Para graficar los vectores y la imagen juntos hacemos

```
1 plotRGB(ref.2016, stretch="lin")
2 plot(firmas, add=TRUE, col='red')
```

donde la primera línea grafica la imagen de fondo y la segunda agrega el shapefile sobre ella.

Actividad 1.4.1. Muestre las propiedades de la capa raster y vectorial y verifique que se encuentren en el mismo sistema de coordenadas.

Veamos como extraer datos de un archivo raster con un vector con la función **extract**. Esta toma dos argumentos, el vector que queremos utilizar y la capa raster sobre la cual hacer la consulta.

Ejemplo 1.4.2. Graficar en un scatterplot de dos bandas mostrando la zona del espacio ocupada por una cobertura.

```
1 datos <- extract(ref.2016, firmas)
```

de esta forma realizamos la extracción de todos los datos de la imagen a una lista

```
1 plot(ref.2016$red, ref.2016$nir)
2 points(as.data.frame(datos[1])$red, as.data.frame(datos[1])$nir, col="green",
3         pch = ".")
```

Agregamos el scatterplot al muestreo obteniendo

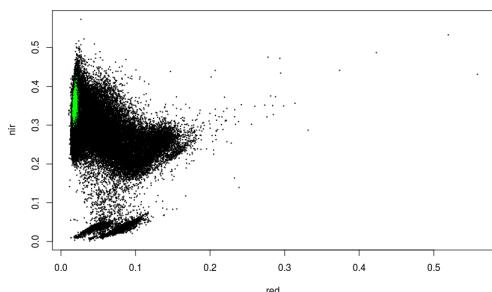


Figura 1.11 – Resultado del scatterplot para las bandas roja y nir. Se muestra en verde datos correspondientes a la selva paranaense.

La función `extract` nos permite también aplicar una función a los datos extraídos antes de entregarlos al usuario. Veamos como usarla para calcular datos de interés sobre las coberturas y guardarlos en un archivo vectorial.

Ejemplo 1.4.3. Extraer los promedios y desvío standar de un raster y agregarlos a un vector. Primero extraemos los valores de promedio y desvío

```
1 promedio <- extract(ref.2016, firmas, fun=mean)
2 desvio <- extract(ref, firmas, fun=sd)
```

renombramos luego las columnas como promedio y desvío seguido de la banda a la que pertenecen,

```
1 colnames(promedio) <- paster("mean", colnames("promedio"), sep="_")
2 colnames(desvio) <- paster("sd", colnames("desvio"), sep="_")
```

finalmente agregamos los archivos a un nuevo shapefile

```
1 firmas@data <- cbind(firmas@data, promedio, desvio)
2 writeOGR(firmas, sdn="vector_data/processed", "firmas_datos",
3           driver="ESRI Shapefile")
```

Finalmente, veamos como usar una capa vectorial para graficar las firmas espectrales y graficarlas para distintas coberturas

Ejemplo 1.4.4. Graficar las firmas espectrales en función de la longitud de onda para cada polígono. Utilizaremos dos nuevas librerías, `reshape2` y `lattice`

Comenzamos convirtiendo en dataframe a nuestros promedios, donde cada columna corresponde a una firma espectral

```
1 df <- t(promedio)
2 colnames(df) <- vector@data$Comment
```

Agregamos luego una columna con las longitudes de onda en nanómetros. Luego reformamos el dataframe para que podamos subsequeirlo, poniendo finalmente los nombres a cada columna

```
1 df$wl <- as.matrix(c(485, 560, 660, 830, 1650, 2215))
2 df <- melt(df, id.vars="wl", variable.name="cobertura")
3 names(df) <- c("wl", "Cobertura", "Reflectancia")
```

El dataframe resultante debería ser:

| | wl | Cobertura | Reflectancia |
|---|------|-----------|--------------|
| 1 | 485 | Alto | 0.012926561 |
| 2 | 560 | Alto | 0.034730646 |
| 3 | 660 | Alto | 0.018491884 |
| 4 | 830 | Alto | 0.354564681 |
| 5 | 1650 | Alto | 0.133750642 |
| | ... | | |

Repetimos el proceso para el desvío standar

```
1 dfd <- t(desvio)
2 colnames(dfd) <- vector@data$Comment
3 dfd$wl <- as.matrix(c(485, 560, 660, 830, 1650, 2215))
4 dfd <- melt("wl", "Cobertura", "Desvio")
5 df$desvio <- dfd$desvio
6 df$MC_ID <- as.character(vector@data$MC_ID[match(df$Cobertura,
7           vector@data$Comment)])
```

El resultado será ahora

| wl | Cobertura | Reflectancia | Desvio |
|--------|-----------|--------------|--------------|
| 1 485 | Alto | 0.012926561 | 0.0007772473 |
| 2 560 | Alto | 0.034730646 | 0.0018113004 |
| 3 660 | Alto | 0.018491884 | 0.0011561294 |
| 4 830 | Alto | 0.354564681 | 0.0166801398 |
| 5 1650 | Alto | 0.133750642 | 0.0075157929 |
| ... | | | |

En primer lugar pondremos todas las firmas espectrales juntas, separadas por color, con la librería **lattice**

```
1 xyplot( Reflectancia ~ wl, data=df, groups = Cobertura,
2         auto.key=list(space="top", columns=4),
3         ty=c("l", "p"))
```

Aquí la primer línea dice que grafiquemos la reflectancia como función de la longitud de onda, obteniendo los datos del dataframe DF y agrupandolos según la columna cobertura. La siguiente línea agrega la leyenda en la parte superior de la figura con 4 columnas. Por último en la tercera línea pedimos que el gráfico tenga líneas y puntos (Figura 1.12).

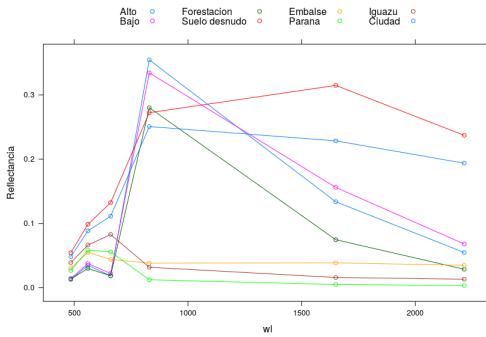


Figura 1.12 – Firmas espectrales

Si queremos agruparlo por categoría de uso y cobertura cambiamos la formula Reflectancia wl por Reflectancia wl | MC_ID

```
1 xyplot( Reflectancia ~ wl | MC_ID, data=df, groups = Cobertura,
2         auto.key=list(space="top", columns=4),
3         ty=c("l", "p"))
```

Si queremos graficar solo un subset de datos (Figura 1.13).

```
1 xyplot( Reflectancia ~ wl | MC_ID, data=df, groups = Cobertura,
2         auto.key=list(space="top", columns=4), ty=c("l", "p"),
3         subset = Cobertura %in% c("Alto", "Bajo"))
```

Actividad 1.4.2. Grafique la media y el desvío standar para las distintas coberturas que pudo identificar en el punto uno.

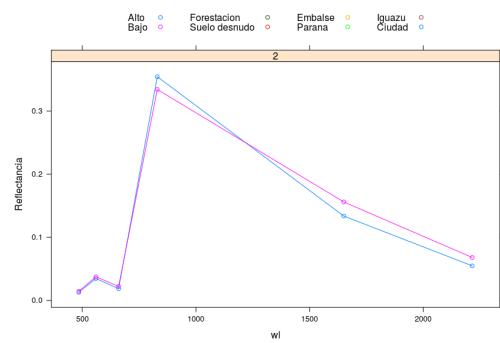


Figura 1.13 – Firmas espectrales

Capítulo 2

Corrección radiométrica

En esta segunda actividad práctica nos centraremos en la corrección radiométrica de imágenes satelitales. Son nuestros objetivos:

- Abrir una imagen satelital desde el metadato.
- Convertir los valores de la imagen a reflectancia a tope de la atmósfera.
- Corregir la imagen satelital utilizando los métodos de *dos* y *cost*
- Corregir la imagen satelital utilizando el *6S* en su versión web.

2.1. Cálculo de reflectancia a tope de la atmósfera

Para poder convertir una imagen a reflectancia a tope de la atmósfera vamos a necesitar la información adicional que hallaremos en su metadato. Para abrirla desde el metadato utilizaremos las funciones disponibles en la librería **RStoolbox**.

Ejemplo 2.1.1. Abrimos la imagen Landsat 7 del año 2000 desde el metadato y la mostraremos en combinación color real y analicemos sus propiedades.

```
1 meta.2000 <- readMeta("raster_data/LE72240782000188EDC00/LE72240782000188EDC00_MTL.txt")
```

Podemos mostrar las distintas variables incluidas en el objeto usando el signo \$ y su nombre. Por ejemplo, `meta.2000$SOLAR_PARAMETERS` da como resultado

```
azimuth elevation distance
37.38251 31.14409 1.01670
```

Usando el metadato podemos cargar la imagen completa con el comando `stackMeta`. Eliminamos, en este caso, las bandas 6 y 7 por ser térmicas.

```
1 dn.2000 <- stackMeta(meta.2000)
2 dn.2000 <- dn.2000[[-6:-7,]]
3 dn.2000
```

El resultado es un objeto *raster stack* como el que sigue

```
class      : RasterStack
dimensions : 2412, 1834, 4423608, 6 (nrow, ncol, ncell, nlayers)
resolution : 30.00402, 30.00265 (x, y)
extent     : 731118.6, 786146, 7101531, 7173897 (xmin, xmax, ymin, ymax)
```

```

coord. ref. : +proj=utm +zone=21 +south +datum=WGS84 +units=m +no_defs
              +ellps=WGS84 +towgs84=0,0,0
names       : B1_dn, B2_dn, B3_dn, B4_dn, B5_dn, B7_dn
min values  :      0,      0,      0,      0,      0,      0
max values  :    255,    255,    255,    255,    255,    255

```

Mostramos la imagen en combinación color real con `plotRGB(dn.2000, r=3, g=2, b=1, stretch="lin")` (Figura 2.1).



Figura 2.1 – Imagen en combinación de bandas color real de la zona de interés.

De esta forma tenemos el archivo en número digital con todos sus metadatos para convertirlo a reflectancia y realizar las distintas correcciones.

Existen dos formas de convertirla a reflectancia a tope de la atmósfera: a mano, utilizando las herramientas algebraicas de R o con la función específica de `RStoolbox`.

Ejemplo 2.1.2. Cálculo de reflectancia a tope de la atmósfera utilizando el metadato paso por paso

```

1 dn2ref.2000 <- meta.2000$CALREF[1:6,]
2 elev.2000 <- pi*meta.2000$SOLAR_PARAMETERS['elevation']/180

```

extraemos primero del metadato los parámetros de calibración en reflectancia y el ángulo de elevación solar. Convertimos luego la imagen a reflectancia y la dividimos por el seno del ángulo solar. Luego cambiamos los nombres de las bandas

```

1 toam.2000 <- (dn.2000*dn2ref.2000$gain+dn2ref.2000$offset)/sin(elev.2000)
2 names(toam.2000) <- c("blue","green","red","nir","swirl1","swirl2")

```

Otra forma de realizarlo es utilizando la función `radCor`. En este caso, debemos tener la imagen en DN, el metadato y cual es la cantidad que queremos calcular.

2.2. CÁLCULO DE REFLECTANCIA CORREGIDA ATMOSFÉRICAMENTE POR MÉTODOS ESTADÍSTICOS

```
1     toa.2000 <- radCor(dn.2000, metaData = meta.2000, method = "apref")
```

Podemos comparar los resultados de ambos métodos inspeccionando los objetos `toa.2000` y `toa.2000`.

```
class      : RasterBrick
dimensions : 2412, 1834, 4423608, 6 (nrow, ncol, ncell, nlayers)
resolution : 30.00402, 30.00265 (x, y)
extent     : 731118.6, 786146, 7101531, 7173897 (xmin, xmax, ymin, ymax)
coord. ref. : +proj=utm +zone=21 +south +datum=WGS84 +units=m +no_defs
              +ellps=WGS84 +towgs84=0,0,0
data source : in memory
names       : blue, green, red, nir, swir1, ...
min values  : -0.01976113, -0.02181530, -0.02029439, 0.01934678, -0.02781926, ...
max values  : 0.6106812, 0.5609009, 0.6079443, 0.8696885, 0.8640919, ...

y

class      : RasterStack
dimensions : 2412, 1834, 4423608, 6 (nrow, ncol, ncell, nlayers)
resolution : 30.00402, 30.00265 (x, y)
extent     : 731118.6, 786146, 7101531, 7173897 (xmin, xmax, ymin, ymax)
coord. ref. : +proj=utm +zone=21 +south +datum=WGS84 +units=m +no_defs
              +ellps=WGS84 +towgs84=0,0,0
names       : B1_tre, B2_tre, B3_tre, B4_tre, B5_tre, ...
min values  : 0.00000000, 0.00000000, 0.00000000, 0.01934678, 0.00000000, ...
max values  : 0.6106812, 0.5609009, 0.6079443, 0.8696885, 0.8640919, ...
```

Actividad 2.1.1. Inspeccione la reflectancia a tope de la atmósfera para todas las bandas. Para esto realice los histogramas, graficos de dispersión, calcule la media, el desvío standar y cualquier otra medida estadística que deseé.

2.2. Cálculo de reflectancia corregida atmosféricamente por métodos estadísticos

La función `radCor` dispone de un parámetro para hacer distintos tipos de correcciones atmosféricas. Ya utilizamos `apref`, que nos permitió calcular la reflectancia a tope de la atmósfera, veamos ahora como aplicar el método de substracción de cuerpo oscuro.

Ejemplo 2.2.1. Apliquemos el metodo de *simple dos* para corregir la imagen. En este caso solamente restaremos el mínimo en cada banda a la imagen para las bandas donde existe haze, es decir en la zona del visible y del infrarrojo cercano.

Estimamos el haze primero y luego corregimos la imagen haciendo

```
1 haze.2000 <- estimateHaze(dn.2000, darkProp = 0.01, hazeBands = 1:4, plot=TRUE)
2 sdos.2000 <- radCor(dn.2000, metaData = meta.2000,
3                       hazeValues = haze.2000,
4                       hazeBands = c("B1_dn", "B2_dn", "B3_dn", "B4_dn"),
5                       method="sdos")
```

en este caso los valores de haze estimados son

| B1_dn | B2_dn | B3_dn | B4_dn |
|-------|-------|-------|-------|
| 41 | 27 | 20 | 15 |

Para hacer un análisis de lo que pasa, vamos a graficar los histogramas de cada banda para la imagen en reflectancia TOA y corregida por el método *simple dos*. Usaremos el paquete `rasterVis`

```

1 B1 <- densityplot(~B1_tre+B1_sre, data=toa.boa, xlab="Reflectancia",
2                      ylab="", main="Banda azul", plot.points=FALSE, xlim=c(0,0.3),
3                      key=simpleKey(text=c("Tope de la atmósfera",
4                               "Corrección Simple DOS"),
5                               lines=TRUE, points=FALSE))
6 B2 <- densityplot(~B2_tre+B2_sre, data=toa.boa, xlab="Reflectancia",
7                      ylab="", main="Banda verde", plot.points=FALSE, xlim=c(0,0.3),
8                      ,
9                      key=simpleKey(text=c("Tope de la atmósfera",
10                         "Corrección Simple DOS"),
11                         lines=TRUE, points=FALSE))
12 B3 <- densityplot(~B3_tre+B3_sre, data=toa.boa, xlab="Reflectancia",
13                      ylab="", main="Banda roja", plot.points=FALSE, xlim=c(0,0.3),
14                      key=simpleKey(text=c("Tope de la atmósfera",
15                         "Corrección Simple DOS"),
16                         lines=TRUE, points=FALSE))
17 B4 <- densityplot(~B4_tre+B4_sre, data=toa.boa, xlab="Reflectancia",
18                      ylab="", main="Banda nir", plot.points=FALSE, xlim=c(0,0.3),
19                      key=simpleKey(text=c("Tope de la atmósfera",
20                         "Corrección Simple DOS"),
21                         lines=TRUE, points=FALSE))
22 print(B1, split = c(1, 1, 2, 2), more=TRUE)
23 print(B2, split = c(2, 1, 2, 2), more=TRUE)
24 print(B3, split = c(1, 2, 2, 2), more=TRUE)
25 print(B4, split = c(2, 2, 2, 2), more=FALSE)

```

las primeras 4 funciones crean los histogramas para cada banda corregida, mientras que las últimas 4 líneas los imprimen en una grilla (Figura 2.2). Notamos en este caso que la corrección se vuelve

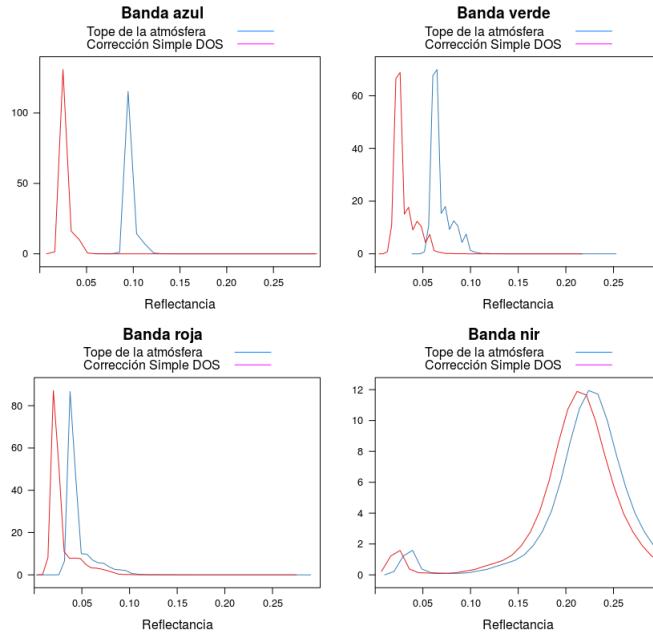


Figura 2.2 – Graficos de densidad para las distintas bandas donde se muestra el nivel de corrección en cada una.

menos importante a medida que crece la longitud de onda. Además, la corrección solo cambia la posición de la distribución y no su forma.

Actividad 2.2.1. Analice los valores de haze obtenidos por la función stimate haze y grafíquelos como función de la longitud de onda en escala logarítmica. ¿Qué observa?

Actividad 2.2.2. Utilice el método *costz* para corregir la imagen a reflectancia a tope de la superficie. Puede ayudarse con el comando `?radCor`

Actividad 2.2.3. Guarde el archivo raster generado por cada uno de los métodos de corrección. Abralos en QGIS y compárelos visualmente. Obtenga firmas espectrales con los distintos métodos de corrección.

2.3. 6S

Veamos ahora como operar con el 6S para obtener una estimación de los parámetros atmosféricos. Para ello utilizaremos la versión web del 6S que se encuentra disponible en <http://6s.ltdri.org/pages/run6SV.html>.

Ingresaremos a la página y haremos click en el botón *Submit query*. iremos luego configurando paso a paso nuestro modelo de la atmósfera, haciendo siempre click en el botón *submit query* para ir al paso siguiente.

Los parametros para nuestro modelo son

1. Geometrical conditions

- TM (Landsat)
- Month: 4, Day:13, GTM decimal hour: 13.60, Longitude: -63.8606, Latitude: -24.9937.

2. Atmospheric Model

- Select Atmospheric Profile: Mid latitude summer
- Select aerosol model: Continental Model
- Visibility: 60

3. Target & sensor altitude

- Select targe altitude: sea level
- Select sensor altitude: satellite level

4. Spectral conditions

- Select spectral conditions: choose band
- Select band: 1st band of tm (landat 5)

5. Ground reflectance

- Ground reflectance type: homogeneous surface
- Directional effect: no directional effect
- Specify surface reflectance: input constant value of ro
- input constant value for ro: 0

6. Signal

- Atmospheric correction mode: no atmospheric correction

Todos estos valores los encontramos en el metadato de la imagen.

En *7.Results* podemos ver el resultado haciendo click en *Output file*. Extraemos los valores de

- `global gas. trans. - total`
- `total sca. trans. - total`

- spherical albedo - total
- reflectance I - total

Una vez ejecutado el proceso puede usarse el siguiente código para corregir todas las bandas utilizando R.

```

1 a <- c(0.98,0.90,...) # Global gas transmitance
2 b <- c(0.81,0.90,...) # Total scattering transmitance
3 g <- c(0.15,0.10,...) # Spherical albedo
4 r <- c(0.08,0.05,...) # Reflectance I
5 sss.2000 <- (toa.2000/(a*b)-r/b)/(1+g*(toa.2000/(a*b)-r/b))

```

Actividad 2.3.1. Realice una extracción de firmas espectrales para distintas coberturas de cada uno de los archivos raster obtenidos y grafíquelos juntos. Compare los resultados con la firma espectral realizada a partir de la imagen corregida por el USGS.

Actividad 2.3.2. Haga un gráfico de densidades que muestre los distintos métodos de corrección atmosféricos para cada banda.

Actividad 2.3.3. Para cada banda calcule la diferencia promedio entre las imágenes en reflectancia a tope de la atmósfera, las distintas correcciones y la imagen en reflectancia entregada por el USGS.

Capítulo 3

Cálculo de índices espetrales

En esta tercer práctica comenzaremos a trabajar con operaciones matemáticas entre las bandas y relacionaremos los resultados con variables biofísicas medibles en el terreno. Son nuestros objetivos:

- Calcular los índices de vegetación a partir de las imágenes en reflectancia.
- Calcular la línea de suelo como parámetro para obtener índices de vegetación.
- Realizar modelos empíricos que relacionen variables biofísicas medidas a campo con los índices espetrales.
- Construir mapas a partir de los modelos empíricos antes mencionados.

3.1. Cálculo de índices entre bandas

Utilizaremos las librerías `raster` y `RStoolbox`. Para poder usar mejores paletas de colores utilizaremos la librería `RColorBrewer`. Por último, puede ayudar cargar la librería `rasterVis` para realizar algunos de los gráficos.

Comenzamos primero cargando la imagen desde el metadato y convirtiéndola a reflectancia entre cero y uno.

```
1  xml.2016 <- readMeta("raster_data/LC82240782016304/LC82240782016304LGN00.xml")
2  ref.2016 <- stackMeta(xml.2016, quantity = "sre")
3  scaleF <- getMeta(ref.2016,xml.2016, what = "SCALE_FACTOR")
4  ref.2016 <- ref.2016 * scaleF
5  ref.2016 <- ref.2016[[-1,]]
6  names(ref.2016) <- c("blue","green","red","nir","swirl","swir2")
```

Luego podemos realizar operaciones entre las bandas llamando a cada una por separado. Veamos como ejemplo el calculo de NDVI.

Ejemplo 3.1.1. Cálculo de NDVI a mano.

```
1  ndvi.2016 <- (ref.2016$nir-ref.2016$red)/(ref.2016$nir+ref.2016$red)
2  cols = colorRampPalette(brewer.pal(9,"YlGn"))(256)
3  plot(ndvi.2016, col=cols, zlim = c(0,1))
```

En este caso estamos

- Calculando el NDVI a mano usando la formula $(\rho_n - \rho_r)/(\rho_n + \rho_r)$.
- Obteniendo una rampa de color entre amarillo y verde.

- Graficando el NDVI, usamos como colores la rampa anterior y ajustandolo entre 0 y 1, es decir, los valores menores a 0 se mostrarán todos del mismo color.

Obtenemos entonces el resultado de la figura 3.1

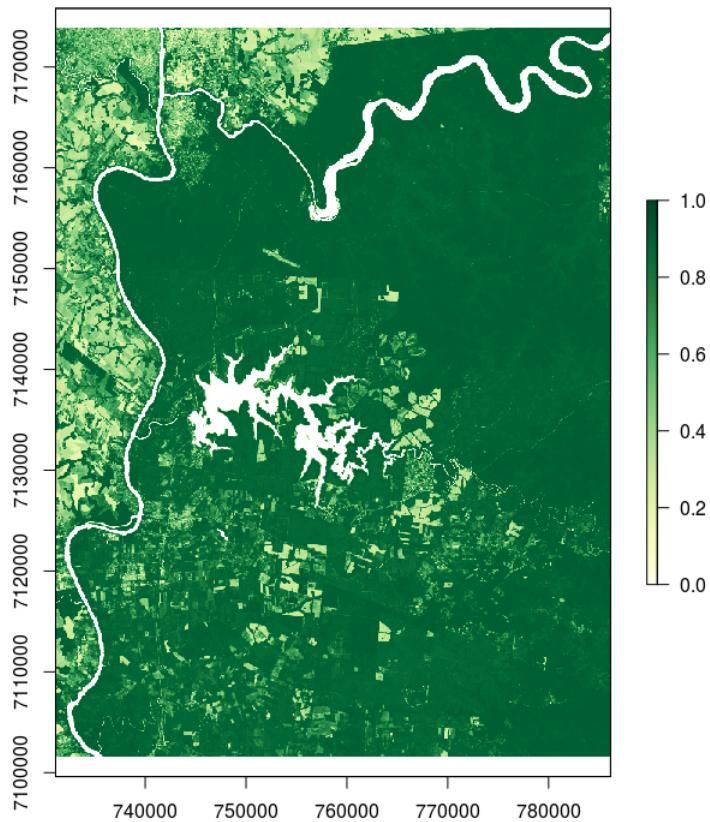


Figura 3.1 – Mapa del NDVI en escala de verdes.

El paquete `RStoolbox` tiene varias herramientas que nos ayudan a calcular los índices espectrales. Veamos por ejemplo como calcular el NDVI y el EVI.

Ejemplo 3.1.2. Para calcular los índices mediante la función `spectralIndices` debemos especificar con que raster trabajamos y que bandas corresponden a cada longitud de onda

```

1  indices.2016 <- spectralIndices(ref.2016,
2                                     blue="blue", red="red", nir="nir",
3                                     indices=c("NDVI", "EVI"))
4  plot(indices.2016, col=cols, zlim=c(0,1))

```

En este caso, vemos que la función `spectralIndices` necesita al menos 3 parámetros

- Una imagen.
- A que zona del espectro corresponde cada banda.


```
4 masked.2016[masked.2016<=0] <- 255
```

de esta forma enmascaramos todos los valores con nubes, agua y donde la reflectancia obtenida es cero con el valor 255. Calculamos ahora la línea de suelo y la mostramos en un scatterplot

```
1 bsl.2016 <- BSL(as.matrix(masked.2016$red), as.matrix(masked.2016$nir),
2                         method="quantile")
3 plot(ref.2016$red, ref.2016$nir)
4 abline(bsl.2016$BSL, col="red")
```

Obtenemos como resultado el gráfico de (Figura 3.3). Podemos consultar los demás parámetros imprimiendo la variable `bsl.2016`.

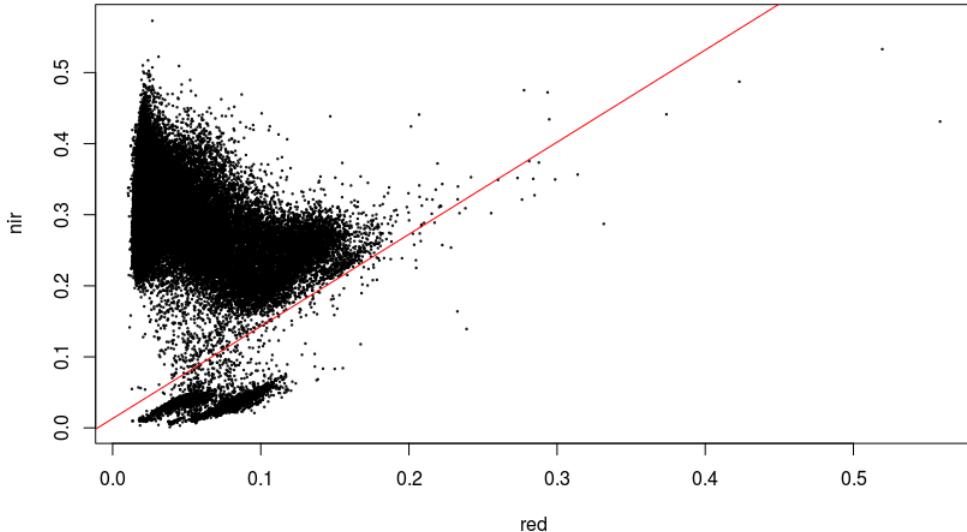


Figura 3.3 – Línea de suelo sobre el scatterplot nir-red

Actividad 3.2.1. Calcule el tSAVI utilizando la línea de suelo.

Actividad 3.2.2. Calcule la línea de suelo sin enmascarar la imagen y dibuje el scatterplot conto con ella.

Actividad 3.2.3. Obtenga la línea de suelo y calcule el índice tSAVI para la imagen del año 2000.

3.3. Estimación de parámetros biofísicos

Finalmente, veamos como se puede obtener datos biofísicos a partir de los índices de vegetación para realizar mapas de porcentaje de cobertura, productividad, etc.

Ejemplo 3.3.1. Comenzamos calculando el NDVI para el año 2016, y utilizando la capa muestreo, hacemos una extracción estadística sobre la misma.

```

1  vector <- readOGR(dsn="vector_data/", layer="muestreo")
2  datos <- extract(ndvi.2016, vector)
3  DF <- data.frame(vector@data, datos)
4  pairs(DF)

```

Obtendremos un gráfico que presenta los scatterplots entre las bandas, su correlación e histogramas. Vemos que la superficie cubierta por vegetación varía linealmente con el NDVI. Por lo tanto, los utilizaremos para hacer un ajuste de nuestro modelo

```

1  lm.2016 <- lm(fcover ~ ndvi, data=muestreo)
2  plot(muestreo$ndvi, muestreo$fcover)
3  abline(lm.2016, col="red")
4  summary(lm.2016)

```

de esta forma obtenemos los parámetros de nuestro ajuste.

Para aplicar el modelo a nuestro raster hacemos

```

1  fcover.2016 <- predict(ndvi.2016, lm.2016)
2  plot(fcover.2016)

```

Actividad 3.3.1. Genere los modelos de lai, fapar y fcover para el año 2016 y realice mapas de dichas variables.

Actividad 3.3.2. Utilizando los modelos obtenidos para 2016 realice los mapas de lai, fapar y fcover del año 2000. ¿Que suposición está haciendo? Compare distintas zonas de los modelos en el qgis.

Capítulo 4

Rotaciones espectrales

En nuestra cuarta práctica trabajemos con rotaciones en el espacio espectral. Apuntamos a poder incorporar conceptos como dimensionalidad y correlación entre bandas, que nos ayuden a utilizar mejor la información satelital.

Son nuestros objetivos:

- Aplicar la transformada tasseled cap a una imagen multiespectral e interpretar el significado de cada banda.
- Aplicar la transformada por componentes principales a una imagen multiespectral e interpretar el resultado.
- Aplicar la transformada por componentes principales a un stack de bandas multiespectrales de distintas fechas e interpretar los resultados.
- Extraer información de series temporales de índices espectrales.

4.1. Cálculo de la transformada tasseled cap para una imagen landsat

Comencemos calculando la transformada tasseled cap para una imagen Landsat 8. Ésta será la primer rotación que utilizaremos en el curso cuya la interpretación es bastante sencilla. Usaremos el paquete **RStoolbox**.

Ejemplo 4.1.1. Para cacular la transformada por componentes principales comenzamos abriendo la imagen Landsat 8 desde el metadado y convirtiéndola a reflectancia a tope de la cobertura, como vimos en las clases anteriores.

```
1  xml.2016 <- readMeta("raster_data/LC82240782016304/LC82240782016304LGN00.xml")
2  ref.2016 <- stackMeta(xml.2016, quantity = "sre")
3  scaleF <- getMeta(ref.2016,xml.2016, what = "SCALE_FACTOR")
4  ref.2016 <- ref.2016 * scaleF
5  ref.2016 <- ref.2016[[-1,]]
6  names(ref.2016) <- c("blue","red","nir","swirl","swir2")
```

Analizamos ahora el espacio rojo-infrarrojo cercano y rojo-verde para la imagen

```
1  B1 <- xyplot(nir~red,data=ref.2016)
2  B2 <- xyplot(red~green,data=ref.2016)
3  print(B1,split=c(1,1,2,1),more=TRUE)
4  print(B2,split=c(2,1,2,1),more=FALSE)
```

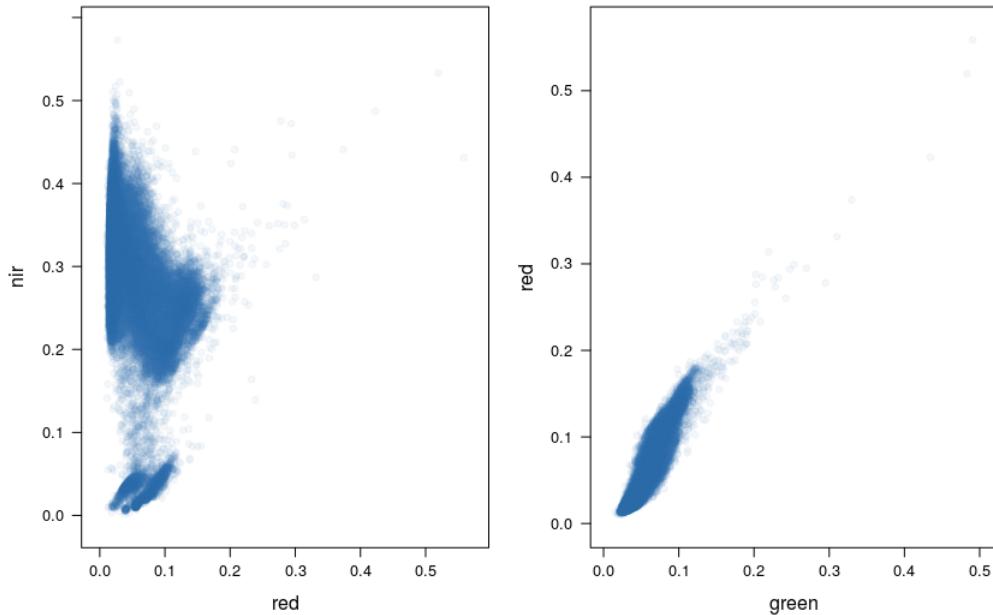


Figura 4.1 – Scatterplot verde-rojo y nir-red.

obteniendo como scatterplots (Figura 4.1).

Calculemos ahora la transformada tasseled cap. Para eso usamos la función `tasseledCap` del paquete `RStoolbox`.

```
1 tsc.2016 <- tasseledCap(r.ref.2016, sat="Landsat8OLI")
```

Tendremos una imagen de tres bandas, *brillo*, *verdor* y *humedad*. Podemos graficar cada una de las bandas por separado con el comando `plot(tsc.2016)` o todas juntas con `plotRGB(tsc.2016, r=1, g=2, b=3, stretch="lin")`

Actividad 4.1.1. Calcule la transformada tasseled cap para la imagen Landsat 7 del año 2000.

4.2. Cálculo de la transformada por componentes principales

Veamos ahora como calcular la transformada por componentes principales. Utilizaremos la herramienta `rasterPCA` del paquete `RStoolbox`.

Ejemplo 4.2.1. Comencemos analizando la transformada por componentes principales de la imagen de 2016. Miremos primero los scatterplots con el comando `pairs(ref.2016)` obteniéndolos (Figura 4.2)

Mirando el resumen de la imagen vemos que hay varias bandas muy correlacionadas entre sí, como las del visible, mientras que otras lo están poco, como el infrarrojo cercano y el infrarrojo de onda corta. Por lo tanto, esperamos que no todas las bandas sean necesarias para explicar el comportamiento de la imagen, al menos en el nivel de detalle más bajo.

Apliquemos entonces la transformada por componentes principales y veamos que sucede

```
1 pca.2016 <- rasterPCA(r.ref.2016)
2 summary(pca.2016$model)
```

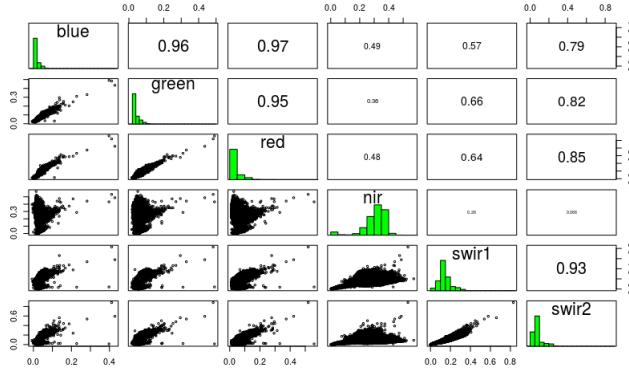


Figura 4.2 – Scaterplots y coeficientes de correlación para la imagen Landsat 8.

El sumario del modelo obtenidos,

Importance of components:

| | Comp. 1 | Comp. 2 | Comp. 3 | Comp. 4 | ... |
|------------------------|------------|------------|------------|-------------|-----|
| Standard deviation | 0.08079854 | 0.07808556 | 0.01242745 | 0.006488765 | ... |
| Proportion of Variance | 0.50850204 | 0.47492732 | 0.01202957 | 0.003279516 | ... |
| Cumulative Proportion | 0.50850204 | 0.98342936 | 0.99545892 | 0.998738441 | ... |

Al observar las varianzas, vemos que las 3 primeras explican más que el 99.5% de la variabilidad de la imagen. Es decir que, de las 6 bandas de Landsat 8 en esta imagen, 3 nos alcanza para explicar casi todo el comportamiento. Analisemos la primera, usando el comando `loadings(pca.2016$model)` para ver como son las componentes.

```
loadings:
  Comp.1 ...
blue
green
red -0.128 ...
nir -0.575 ...
swir1 -0.663 ...
swir2 -0.451 ...
```

La primer componente pesa siempre con el mismo signo, a todas las bandas. Podemos interpretarla como un brillo negativo y esperar que sea más alta cuando miramos zonas de la imagen que tengan menos reflectancia en promedio. La segunda componente, presenta diferencia entre los valores del infrarrojo cercano y el resto de la bandas. Esta será alta en presencia de vegetación y baja en su ausencia. La componente tres se comporta de forma similar pero con las bandas del infrarrojo medio, por lo tanto podemos interpretarla como una componente que varía según el contenido de humedad.

Actividad 4.2.1. Calcule y analice la transformada por PCA de la imagen Landsat 7 del año 2000.

4.3. Algunas ideas sobre series temporales

Empecemos esta sección con una actividad

Actividad 4.3.1. Aplique la transformada por componentes principales al stack de bandas del año 2000 y 2016.

Veamos que pasa al trabajar con series temporales de índices.

Ejemplo 4.3.1. Otra aplicación de la transformada por componentes principales es el análisis de series temporales.

```
1 ndvi.list <- list.files("raster_data/MOD13Q1/NDVI/", pattern = "*.tif$",
2                           full.names = TRUE)
3 ndvi.stack <- stack(ndvi.list)
```

una vez abierta la imagen la convertimos a valores entre -1 y 1 e interpolamos los valores faltantes.

```
1 ndvi.stack <- ndvi.stack/1e4
2 ndvi.stack <- approxNA(ndvi.stack)
3 writeRaster(ndvi.stack, "ndvi-series.tif")
```

Una vez interpoladas las fechas donde no había datos de NDVI, podemos abrir la en el QGIS y analizar distintas zonas de la imagen.

Utilizando la herramienta de identificar objetos espaciales podemos consultar como es el comportamiento de la serie temporal para cada píxel.

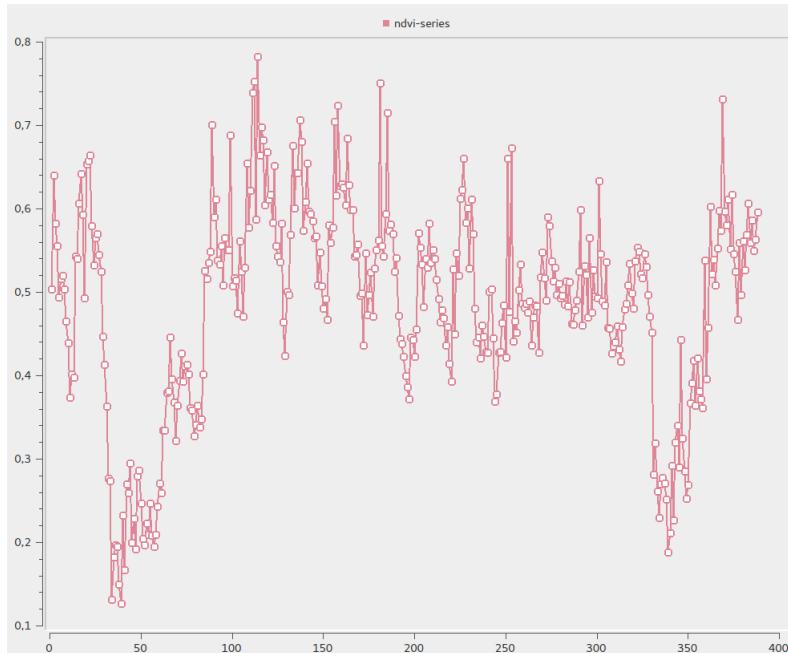


Figura 4.3 – Serie temporal de valores de NDVI.

Vemos que distintas zonas tienen distintos comportamientos intra e interanual. Podemos analizar el promedio y el desvío standar para cada píxel de la imagen

```
1 ndvi.mean <- mean(ndvi.stack)
2 plot(ndvi.mean)
3 ndvi.sd <- calc(ndvi.stack, fun=sd)
4 plot(ndvi.sd)
```

Actividad 4.3.2. Grafique las primeras 4 componentes de la transformada por componentes principales de la imagen del stack de NDVI. ¿Qué zonas puede identificar en la primera? ¿Qué zonas se distinguen en la segunda? ¿Qué comportamiento encuentra en la tercera y cuarta?

Parte II

Variables discretas

Capítulo 5

Clasificación no supervisada

En la quinta clase del curso, trabajaremos con clasificación no supervisada de imágenes satelitales. Son nuestros objetivos:

- Usar la herramienta de clasificación no supervisada.
- Aprender a indentificar clases espectrales y asignarlas a categorías de uso y cobertura.
- Incorporar información no espectral a las clasificaciones como pueden ser datos temporales o información espacial.
- Aplicar la transformada por componentes principales para reducir la dimensionalidad y seleccionar los datos mas relevantes para realizar las clasificaciones.

5.1. Clasificación mediante el método k-means

Cargaremos primero la imagen Landsat 8 y habilitaremos la opción para escribir el header de ENVI.

```
1 rasterOptions(addheader = "ENVI")
2 xml.2016 <- readMeta("raster_data/LC82240782016304/LC82240782016304LGN00.xml")
3 ref.2016 <- stackMeta(xml.2016, quantity = "sre")
4 scaleF <- getMeta(ref.2016,xml.2016, what = "SCALE_FACTOR")
5 ref.2016 <- ref.2016 * scaleF
6 ref.2016 <- ref.2016[[-1,]]
7 names(ref.2016) <- c("blue","green","red","nir","swirl","swir2")
```

Veamos como clasificar una imagen usando el método k-means en R. Vamos a usar los paquetes `raster` y `RStoolbox`

Ejemplo 5.1.1. Elegimos la semilla para el geneador de números aleatorios con el comando `set.seed(42)`. De esta forma, la serie de números aleatorios es la misma para todos.

Luego, clasificamos la imagen y la guardamos como vimos en las clases anteriores.

```
1 set.seed(42)
2 kmeans.2016 <- unsuperClass(ref.2016, nClasses = 5, nStarts = 100,
3                                nSamples = 100)
4 writeRaster(kmeans.2016$map, "raster_data/processed/kmeans2016",
5             datatype="INT1U")
```

En este caso estamos usando solamente 5 clases espectrales. Podemos ahora graficar por separado cada una de las clases (figura 5.1).

```

1  clases.2016 <- layerize(kmeans.2016$map)
2  plot(clases.2016)

```

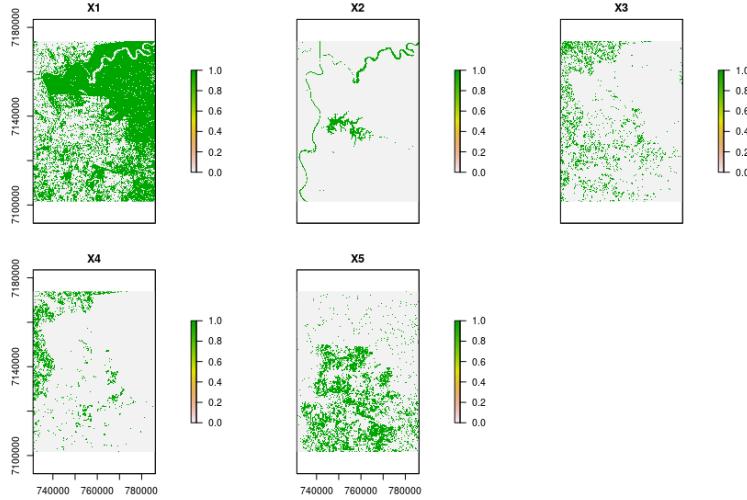


Figura 5.1 – Clases generadas por el algoritmo kmeans para 5 clases.

Abrimos la imagen en el QGIS e identificamos cada una de las clases. Para esto primero vamos al menú *propiedades de la imagen* → *Estilo* → *Tipo de renderización* → *Unibanda pseudocolor*. Elegimos de modo *Intervalo Igual* y en número de clases ponemos con el mínimo en 1 y el máximo en 5. En estilo de color elegimos colores aleatorios.

Construimos una tabla como la siguiente en un editor de texto.

| id | class |
|----|-------|
| 1 | 2 |
| 2 | 7 |
| 3 | 1 |
| 4 | 1 |
| 5 | 1 |

La guardamos en un archivo de texto con el nombre `class.txt`.

Una vez conocidas las categorías de uso y cobertura correspondientes a cada clase espectral podemos combinarlas

```

1  clases.2016 <- read.delim("aux_data/class.txt")
2  reclas.2016 <- subs(kmeans.2016$map, clases.2016)

```

y graficarlas (figura 5.2)

```

1  colores = c('#b2df8a', '#33a02c',
2    '#fdbf6f', '#ff7f00',
3    '#fb9a99', '#e31a1c',
4    '#a6cee3', '#1f78b4')
5  plot(reclas.2016, col=colores, zlim=c(1,8))

```

Ejemplo 5.1.2. Hagamos un análisis espectral de las imágenes clasificadas antes y después de la fusión. Utilizaremos a la imagen clasificada como máscara para asignar los colores al scatterplot.

Agregamos la imagen clasificada a las bandas de la imagen en reflectancia y luego hacemos los scatterplot según la clase espectral o de información como

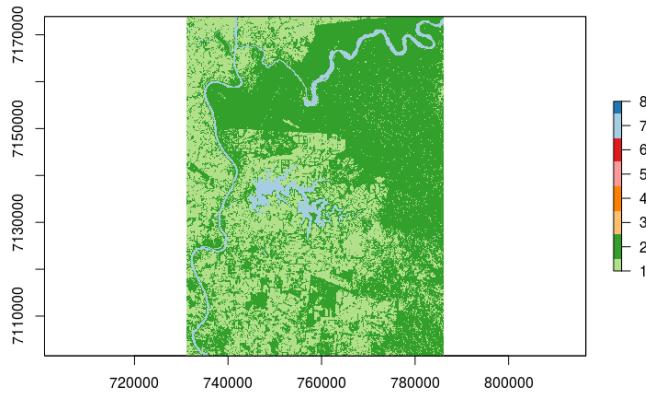


Figura 5.2 – Imagen clasificada por el método kmeans con 5 clases espectrales.

```

1 stack.2016 <- stack(ref.2016, reclas.2016)
2 xyplot(nir+swirl~red, groups=MC_ID, data=stack.2016)

```

Vemos que las clases espectrales pueden unirse en clases de información y que algunas clases de información no se separan en distintas clases espectrales (figuras 5.3).

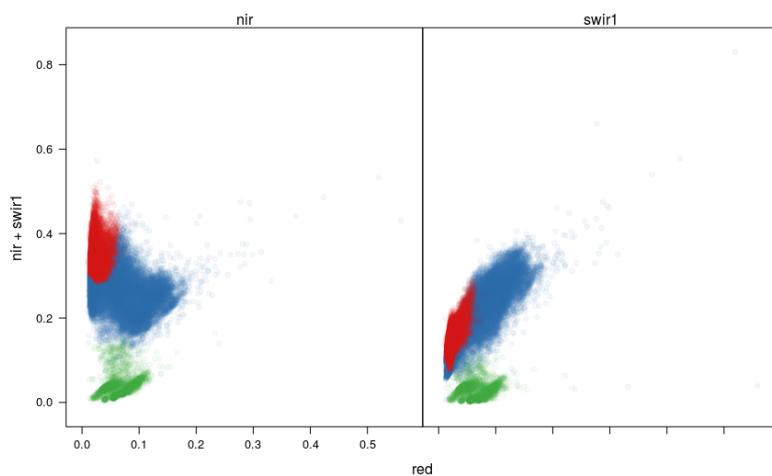


Figura 5.3 – Espaciopectral clasificado por kmeans.

Ejemplo 5.1.3. Repitamos este análisis pero con 100 clases espectrales. El algoritmo es el mismo pero ahora vamos a usar el parametro nClasses igual a 100.

```

1 set.seed(42)
2 kmeans.2016b <- unsuperClass(ref.2016, nClasses = 100, nStarts = 100,
3                                     nSamples = 1000)
4 writeRaster(kmeans.2016b$map, "raster_data/processed/kmeans2016b",
5             datatype="INT1U", overwrite=TRUE)
6 clasesb.2016 <- read.delim("aux_data/class100.txt")
7 reclasb.2016 <- subs(kmeans.2016b$map, clasesb.2016)
8 plot(reclasb.2016, col=colores, zlim=c(1,8))

```

Comparamos las clasificaciones obtenidas con 5 y 100 classes espetrales (figura 5.4)

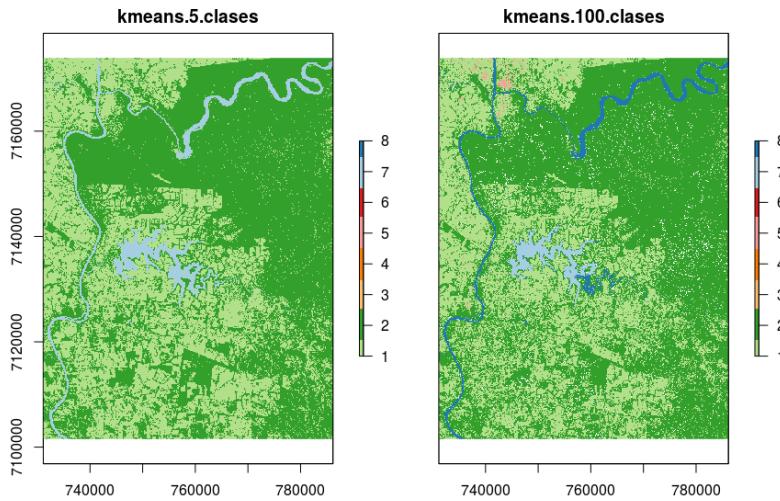


Figura 5.4 – Comparación entre la clasificación con 5 clases y 100 clases espetrales.

Comparamos nuevamente los espacios de fase entre la clasificaciones obtenidas a partir de 5 y 100 clases espetrales

```

1 stackb.2016 <- stack(ref.2016, reclasb.2016)
2 xyplot(nir+swirl~red, groups=MC_ID, data=stack.2016)
3 xyplot(nir+swirl~red, groups=MC_ID, data=stackb.2016)

```

Actividad 5.1.1. Repita la clasificación para la imagen landsat 7 del año 2000.

5.2. Información espacial y temporal

Con el fin de mejorar la clasificación podemos incorporar información espacial y temporal a la información radiométrica. De esta forma, al momento de clasificar los píxeles, ya no estaremos trabajando en un espacio espectral, sino en uno más amplio. Veamos como hacerlo.

Ejemplo 5.2.1. Para incorporar información espacial sobre el contexto del píxel, podemos hacerlo antes o después de la clasificación. Aquí nos concentraremos en el primer caso

Calculamos primero la variabilidad local de brillo para la banda pancromática de la imagen Landsat 8 como

```

1 pan.2016 <- raster("raster_data/LC82240782016304LGN00/LC82240782016304LGN00_B8.
2 TIF")
3 window <- matrix(1,nrow=5, ncol=5)
4 sd.2016<-focal(pan.2016,w=window,fun=sd)
5 plot(log(sd.2016,base = 10), zlim=c(1,4))

```

Vemos que las zonas con mayor presencia de urbanizacion, presentan mayor variabilidad que aquellas con menor (figura 5.5).

Una vez obtenida la banda de desvio standar, podemos agregarla a las demás, luego de remuestreala haciendo

```

1 sd.2016 <- aggregate(sd.2016, fact=2, fun=mean)
2 stack(ref.2016, sd.2016)
3 pca.2016 <- rasterPCA(stack(ref.2016, sd.2016), spca = TRUE)

```

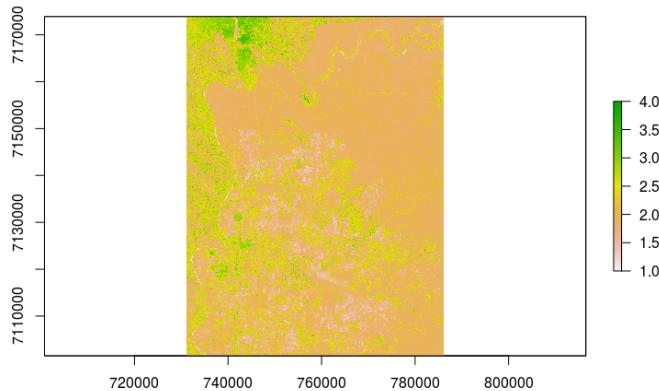


Figura 5.5 – Desvío standar para una ventana de 5 píxeles por 5 píxeles en escala logarítmica.

Importance of components:

| | Comp.1 | Comp.2 | Comp.3 | Comp.4 | Comp.5 | ... |
|------------------------|-----------|-----------|-----------|------------|-------------|-----|
| Standard deviation | 2.1592402 | 1.1821570 | 0.8392878 | 0.40888073 | 0.196591748 | ... |
| Proportion of Variance | 0.6660455 | 0.1996422 | 0.1006292 | 0.02388335 | 0.005521188 | ... |
| Cumulative Proportion | 0.6660455 | 0.8656876 | 0.9663168 | 0.99020013 | 0.995721318 | ... |

Actividad 5.2.1. Repita el ejemplo para la imagen landsat 7 del año 2000. Clasifique por el metodo de kmeans las imagenes de los años 2000 y 2016.

Ejemplo 5.2.2. Para incorporar información del contexto temporal, agregaremos el promedio y el desvío estandar del NDVI a lo largo del 2016. Primero cargamos las imágenes MODIS del 2016

```

1  list.2016 <- list.files("raster_data/MOD13Q1/NDVI/", pattern = "MOD13Q1.A2016+.
2  *tif", full.names = TRUE)
3  ndvi.2016 <- stack(list.2016)/1e4
4  ndvi.2016 <- approxNA(ndvi.2016)
5  sd.2016 <- calc(ndvi.2016, fun = sd)
6  me.2016 <- calc(ndvi.2016, fun = mean)
7  plot(stack(me.2016, sd.2016))

```

Una vez hecho esto, resampleamos el promedio y el desvío de la serie temporal y lo unimos a las imágenes en reflectancia, sobre las cuales calcularemos la transformada por componentes principales.

```

1  sd.2016 <- resample(sd.2016, ref.2016, method="ngb")
2  me.2016 <- resample(me.2016, ref.2016, method="ngb")
3  pca.2016 <- rasterPCA(stack(ref.2016, me.2016, sd.2016), spca = TRUE)
4  summary(pca.2016$model)

```

Actividad 5.2.2. Repita el ejemplo para la imagen Landsat 7 del año 2000. Clasifique por el metodo de kmeans las imagenes de los años 2000 y 2016.

Capítulo 6

Clasificación supervisada

En la sexta clase del curso, trabajaremos con algoritmos de clasificación supervisados. Son nuestros objetivos:

- Poder realizar clasificaciones supervisadas utilizando los distintos algoritmos que se encuentran en R.
- Comparar, utilizando la entropía de un píxel, qué coberturas presentan mayor confusión al momento de la clasificación.

Cargaremos primero la imagen Landsat 8 y habilitaremos la opción para escribir el header de ENVI. Usaremos en primer lugar los paquetes `raster`, `rgdal`, `RStoolbox` y `rasterVis`.

```
1 rasterOptions(addheader = "ENVI")
2 xml.2016 <- readMeta("raster_data/LC82240782016304/LC82240782016304LGN00.xml")
3 ref.2016 <- stackMeta(xml.2016, quantity = "sre")
4 scaleF <- getMeta(ref.2016,xml.2016, what = "SCALE_FACTOR")
5 ref.2016 <- ref.2016 * scaleF
6 ref.2016 <- ref.2016[[-1,]]
7 names(ref.2016) <- c("blue","green","red","nir","swirl1","swirl2")
8 vector <- readOGR(dsn="vector_data", layer="entrenamiento")
```

6.1. Clasificación por máxima verosimilitud

Empecemos con la clasificación por el método de máxima verosimilitud (figura 6.1).

```
1 library(e1071)
2 colores = c('#b2df8a', '#33a02c',
3             '#fdbf6f', '#ff7f00',
4             '#fb9a99', '#e31a1c',
5             '#a6cee3', '#1f78b4')
6 sup.2016 <- superClass(ref.2016, vector, responseCol = "MC_ID",
7                         model = "mlc")
8 plot(sup.2016$map, col=colores, zlim=c(1,8))
```

Cambiando el algoritmo de clasificación en el parámetro `model` podemos calcular distintas clasificaciones supervisadas. Algunas de las vistas en clase son `mlc`, `rf` y `svmRadial`. Cada una de ellas usa alguna librería adicional.

Ejemplo 6.1.1. Una forma de mejorar las clasificaciones supervisadas es clasificar por separado distintas clases espectrales y luego unirlas en la misma clase de información. Veamos como hacerlo.

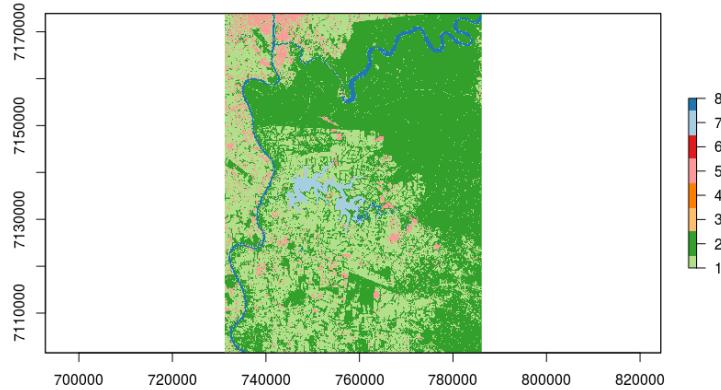


Figura 6.1 – Clasificación por máxima verosimilitud sin separar en clases espectrales.

```

1 sup.2016b <- superClass(ref.2016, vector, responseCol = "C_ID",
2                           model = "mlc")

```

En este caso, la columna de respuesta es C_ID y no MC_ID. Una vez realizada la clasificación, debemos substituir los valores de cada píxel por el de la clase de información correspondiente. Para ello:

```

1 subs.2016 = vector@data[c(3,1)]
2 sub.2016 <- reclassify(sup.2016b$map, subs.2016)
3 writeRaster(sub.2016, "raster_data/processed/mlc2016",
4             datatype="INT1U")

```

Podemos finalmente comparar las dos imágenes clasificadas lado a lado ejecutando el comando `plot(stack(sup.2016$map, sub.2016), col=colores, ylim=c(1,8))` (figura 6.2).

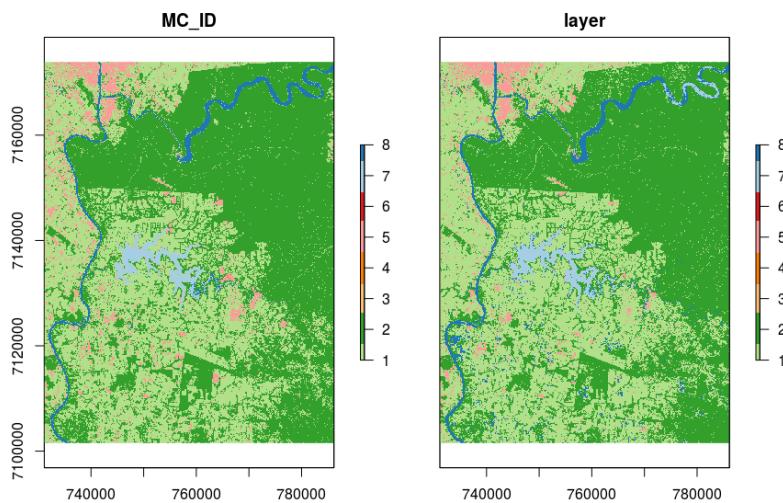


Figura 6.2 – Comparación entre la clasificación utilizando clases de información y clases espectrales.

Actividad 6.1.1. Realice clasificaciones por los distintos métodos y comparelas visualmente.

Actividad 6.1.2. Agregue las bandas de textura y vuelva a clasificar las imágenes.

6.2. Entropía de la clasificación

Para poder comparar en que zonas los clasificadores presentan mas o menos dispersión podemos calcular la entropía de las distintas clasificaciones en cada píxel. Para esto utilizaremos la función `rasterEntropy`.

Ejemplo 6.2.1. Comenzamos corriendo la clasificación para distintos modelos.

```

1  set.seed(42)
2  library(e1071)
3  sup.2016 <- superClass(ref.2016, vector, responseCol = "C_ID",
4                           model = "mlc")
5  mlc.2016 <- reclassify(sup.2016$map, subs.2016)
6
7  library(randomForest)
8  sup.2016 <- superClass(ref.2016, vector, responseCol = "C_ID",
9                           model = "rf")
10 rf.2016 <- reclassify(sup.2016$map, subs.2016)
11
12 library(kernlab)
13 sup.2016 <- superClass(ref.2016, vector, responseCol = "C_ID",
14                           model = "svmLinear")
15 svm.2016 <- reclassify(sup.2016$map, subs.2016)
```

Los apilamos y calculamos la entropía en cada píxel.

```

1 prediction_stack <- stack(mlc.2016, rf.2016, svm.2016)
2 names(prediction_stack) <- c("mlc", "rf", "svm")
3 model_entropy <- rasterEntropy(prediction_stack)
```

Podemos graficar la entropía de las clasificaciones como `model_entropy` y ver que zonas presentan mayor o menor diferencia, a la hora de la clasificación (figura 6.3).

```
1 plot(stack(prediction_stack, model_entropy), col=colores)
```

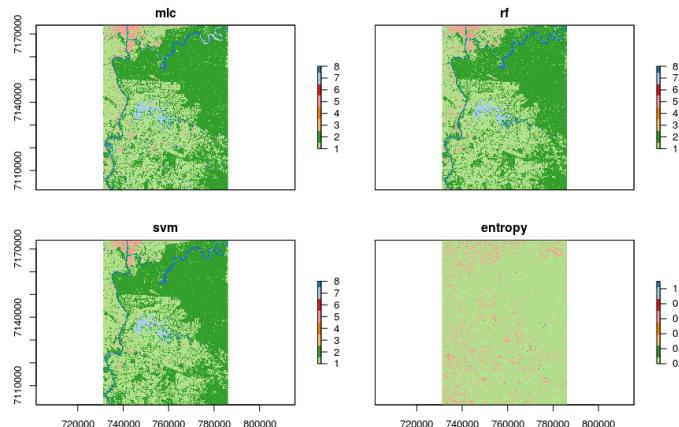


Figura 6.3 – Comparación entre los distintos métodos de clasificación supervisada usando la entropía de la clasificación.

Actividad 6.2.1. Repita las clasificaciones por los métodos de arriba agregando (maxima verosimilitud, random forest, support vector machine) la banda textura. Apílela junto con las clasificaciones por k-means de la clase anterior. ¿A que cobertura pertenecen las zonas con mayor variabilidad?

Capítulo 7

Técnicas pos-clasificación

Veamos finalmente, algunas técnicas de posprocesamiento para imágenes satelitales que nos ayudaran a mejorar los valores extraidos y conocer la incertezza en su estimación. Son nuestros objetivos:

- Aplicar filtros modales a las clasificaciones para eliminar píxeles aislados.
- Calcular la presición total para la clasificación y las precisiones del usuario y productor.
- Estimar el error de las superficies calculadas a partir de la imagen clasificada.

Vamos a usar los paquetes `raster`, `rgdal`, `RStoolbox` y `rasterVis`.

7.1. Filtrado de clasificaciones

Luego de clasificar una imagen es necesario aplicar un filtro que permita eliminar píxeles aislados. Al hacerlo, podremos descartar, píxeles de ciudad en el medio de la selva o de selva en medio de la ciudad. Ésta es otra forma de incorporar el contexto espacial a nuestras clasificaciones

Ejemplo 7.1.1. Veamos como aplicar un filtro por moda a una imagen clasificada. Comencemos cargando una imagen clasificada en R

```
1 mlc.2016 = raster("raster_data/processed/mlc2016")
```

Para aplicar el filtro de 3x3 creamos una matriz de dicha dimensión y usamos el comando `focal` con la función `modal`

```
1 colores = c('#b2df8a', '#33a02c',
2           '#fdbf6f', '#ff7f00',
3           '#fb9a99', '#e31a1c',
4           '#a6cee3', '#1f78b4')
5 window <- matrix(1, nrow=3, ncol=3)
6 mlc.3x3.2016 <- focal(mlc.2016, w=window, fun=modal)
7 writeRaster(mlc.3x3.2016, "raster_data/processed/mlc3x3-2016", datatype = "INT1U"
8           , overwrite=TRUE)
9 plot(mlc.3x3.2016, col=colores)
```

En el caso de un filtro por moda, estamos cambiando el valor del píxel por la moda entre los que lo rodean. Podemos en este caso mostrar las imágenes con y sin el filtro juntas (figura 7.1).

```
1 plot(stack(mlc.2016, mlc.3x3.2016), col=colores)
```

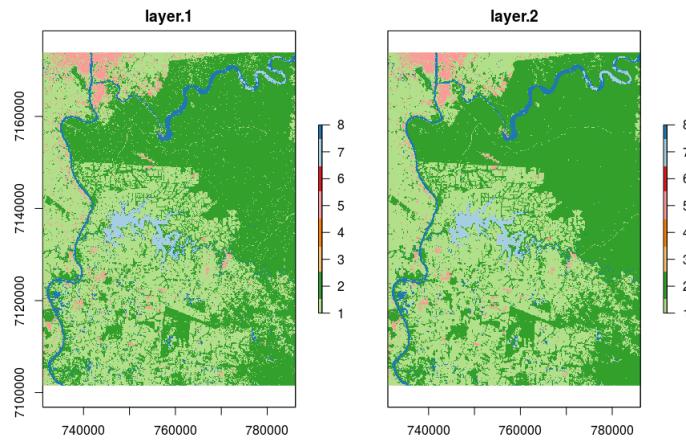


Figura 7.1 – Imagen de la clasificación con y sin filtro con una ventana de 3x3.

Actividad 7.1.1. Aplique filtros de 5x5 y 7x7 a la imagen. ¿Qué problemas desaparecen? ¿Qué dificultad introducen?

Actividad 7.1.2. Aplique el filtro de 3x3 a la imagen correspondiente al año 2000.

7.2. Validación de la clasificación

El segundo procesamiento pos-clasificación, y tal vez el mas importante, es el cálculo de la matriz de confusión para nuestra clasificación. Lo haremos en dos partes: crearemos primero el set de datos para la validación y luego lo usaremos para calcular la matriz de confusión.

Ejemplo 7.2.1. Veamos como crear un set de puntos aleatorios de muestreo con QGIS. En éste caso, utilizaremos como unidad de muestreo al píxel, pero de forma similar pueden usarse polígonos o grupos de polígonos.

Comenzamos abriendo la imagen `mlc3x3-2016` en QGIS. Luego vamos a vectorizar la clasificación utilizando la herramienta *Raster → Conversión → Poligonizar*. Guardamos el archivo como `2016-3x3.shp` en la carpeta de `vector_data` poniendo en nombre del campo `MC_ID` (figura 7.2).

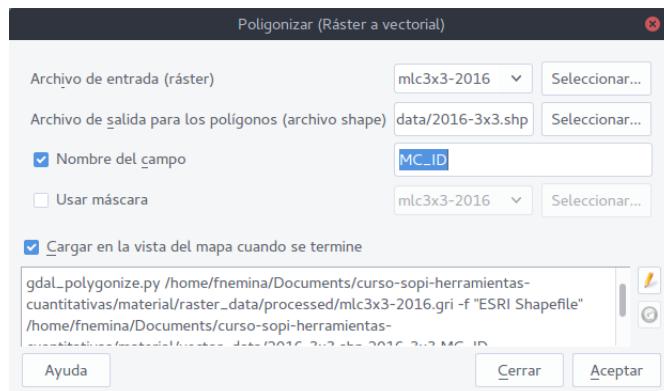


Figura 7.2 – Herramienta de poligonización

Vamos luego a *Herramientas de gestión de datos → Dividir capa vectoria* y guardamos los vectores en la carpeta `vector_data/split` (figura 7.3).

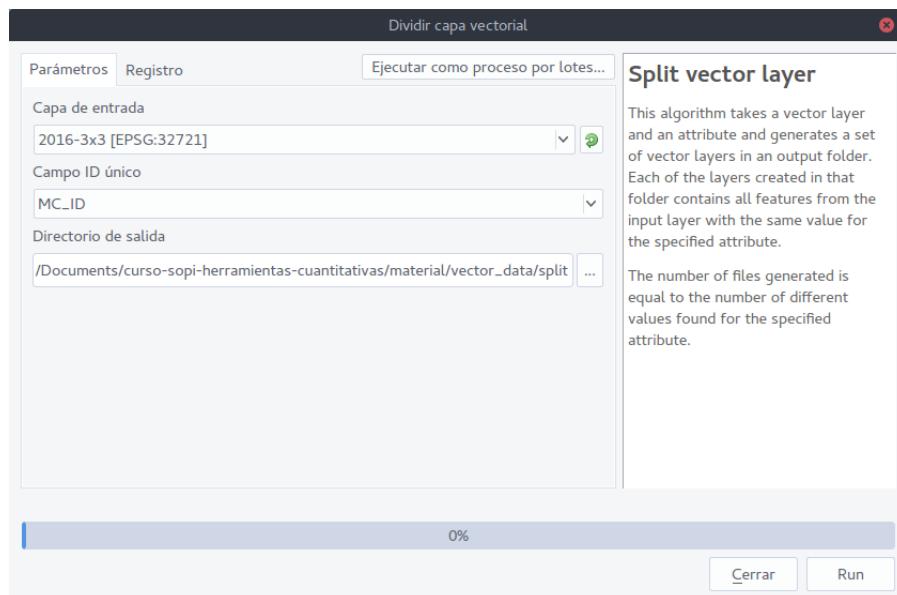


Figura 7.3 – Herramienta de división de capa vectorial.

Cargamos las capas en QGIS. En *Herramientas de investigación → Puntos aleatorios en los límites de la capa* hacemos click en el botón *Ejecutar proceso por lotes*. Luego hacemos click en el botón con los tres puntos y elegimos *Seleccionar del sistema de archivos*. Seleccionamos cada capa. En número de puntos ponemos la cantidad correspondiente a cada categoría.

Para calcular dicha cantidad usamos el comando `freq.2016 <- freq(mlc.3x3.2016)` para conocer las frecuencias de aparición de cada categoría y luego con el comando `freq.2016[,2] <- round(freq.2016[,2]/sum(freq.2016[,2])*250+50)` distribuimos los píxeles en cada categoría con 50 fijos y 250 según la frecuencia de aparición. Los repartimos en la imagen con una distancia de separación mínima de 100m (figura 7.4).

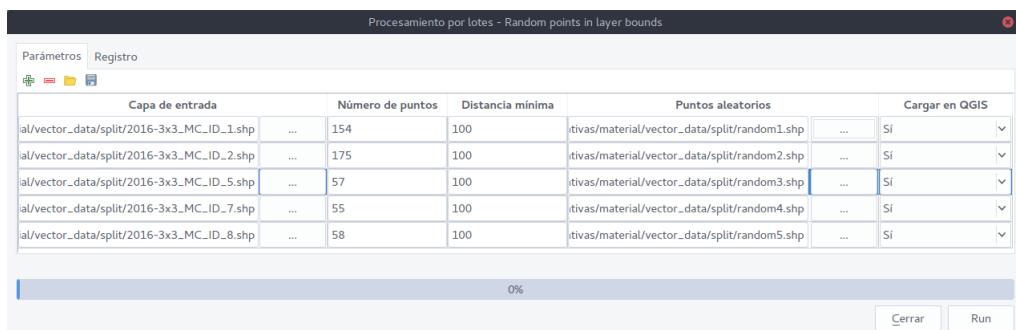


Figura 7.4 – Herramienta de creación de puntos aleatorios.

Elegimos luego en que carpeta guardar los puntos aleatorios y hacemos click en aceptar. Finalmente, una vez creadas las capas de puntos aleatorios las unimos utilizando la herramienta *Herramientas de gestión de datos → Combinar capas vectoriales* con el nombre *val-2016.shp* (figura 7.5).

Luego editamos su tabla de atributos para agregar el campo *MC_ID* y, realizando un análisis visual en distintas combinaciones de bandas, asignamos a cada punto el *MC_ID* de la clase de información a la que pertenece.

Podemos utilizar la misma imagen que usamos para clasificar o una de mayor resolución espacial. Una vez terminada y guardada tendremos nuestra capa de validación.

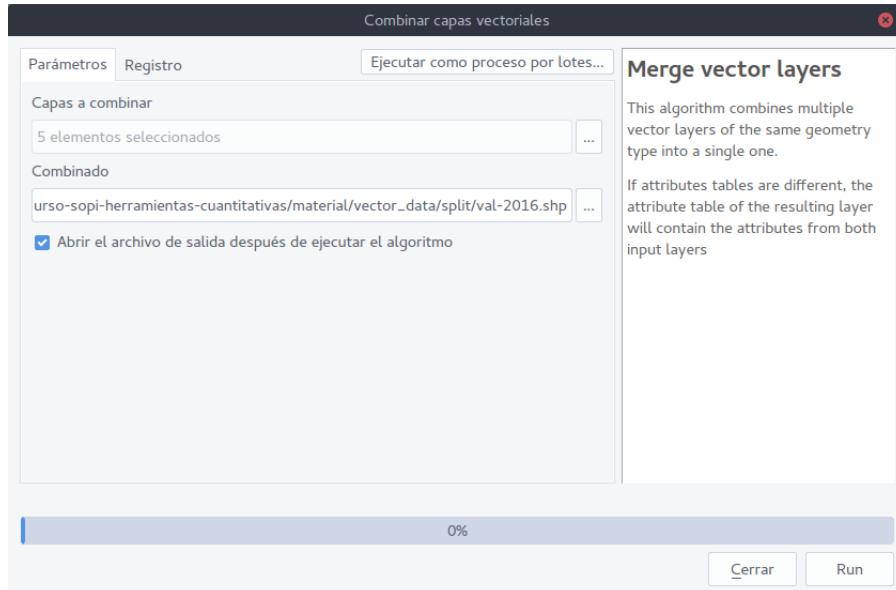


Figura 7.5 – Combinacion de polígonos en una capa vectorial.

Actividad 7.2.1. Construya de esta forma una serie de puntos de validación para la imagen Landsat 7 del año 2000.

Ejemplo 7.2.2. Una vez obtenidos los puntos de validación podemos calcular la matriz de confusión. Para esto debemos cargar el polígono de validación y la calculamos con la función `validateMap`.

```
1   valid.2016 <- readOGR(dsn="vector_data", layer="validacion")
2   val.2016 <- validateMap(mlc.3x3.2016, valData = valid.2016,
3                           responseCol = "MC-ID")
```

Al inspeccionar el elemento `val.2016$performance` obtenemos la matriz de confusión, la presición global como *Acuracy*, la presición del productor como *Sensitivity* y la del usuario como *Pos Pred Value*.

Confusion Matrix and Statistics

| | | Reference | | | | | |
|------------|-----|-----------|----|----|----|---|--|
| Prediction | | 1 | 2 | 5 | 7 | 8 | |
| 1 | 150 | 27 | 14 | 6 | 2 | | |
| 2 | 5 | 164 | 0 | 0 | 0 | | |
| 5 | 6 | 0 | 21 | 0 | 0 | | |
| 7 | 1 | 0 | 0 | 35 | 5 | | |
| 8 | 3 | 0 | 0 | 6 | 54 | | |

Overall Statistics

```
Accuracy : 0.8497
95% CI  : (0.8153, 0.8799)
No Information Rate : 0.3828
P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.7888
McNemar's Test P-Value : NA
```

Statistics by Class:

| | Class: 1 | Class: 2 | Class: 5 | Class: 7 | Class: 8 |
|----------------------|----------|----------|----------|----------|----------|
| Sensitivity | 0.9091 | 0.8586 | 0.60000 | 0.74468 | 0.8852 |
| Specificity | 0.8533 | 0.9838 | 0.98707 | 0.98673 | 0.9795 |
| Pos Pred Value | 0.7538 | 0.9704 | 0.77778 | 0.85366 | 0.8571 |
| Neg Pred Value | 0.9500 | 0.9182 | 0.97034 | 0.97380 | 0.9839 |
| Prevalence | 0.3307 | 0.3828 | 0.07014 | 0.09419 | 0.1222 |
| Detection Rate | 0.3006 | 0.3287 | 0.04208 | 0.07014 | 0.1082 |
| Detection Prevalence | 0.3988 | 0.3387 | 0.05411 | 0.08216 | 0.1263 |
| Balanced Accuracy | 0.8812 | 0.9212 | 0.79353 | 0.86570 | 0.9323 |

Una vez obtenida la matriz de confusión y las superficies mediante el comando `freq()`, podemos usar el script de R `areas.R`. Nos devolverá un archivo csv con las superficies de cada categoría de uso y cobertura, *adj_area*, y su error *CI_adj_area* como se ve al ejecutar el comando `ov[c(1:5,11,12)]`.

| 1 | 2 | 5 | 7 | 8 | adj_area | CI_adj_area |
|---|-------|-------|-------|-------|----------|-------------|
| 1 | 0.318 | 0.053 | 0.030 | 0.013 | 0.004 | 135500.03 |
| 2 | 0.015 | 0.488 | 0.000 | 0.000 | 0.000 | 215369.55 |
| 5 | 0.006 | 0.001 | 0.021 | 0.000 | 0.000 | 20265.40 |
| 7 | 0.000 | 0.000 | 0.000 | 0.016 | 0.003 | 12318.18 |
| 8 | 0.002 | 0.000 | 0.000 | 0.003 | 0.028 | 13907.63 |

Actividad 7.2.2. Construya la matriz de confusión y obtenga la presición global para todas las clasificaciones del año 2000 y 2016. ¿Qué algoritmo funciona mejor? Obtenga estimaciones de las superficies para los años 2000 y 2016 y utilícela para estimar la perdida de selva paranaense durante este período.

Parte III

Apéndice

Apéndice A

Cronograma

Cronograma del curso.

25/4 Instalar el QGIS y el R. Ver apendice C.

26/4 Reflectancia y espacio espectral

- Clase teórica: radiancia y reflectancia. Firmas espectrales y espacio espectral.
- Clase práctica: capítulo 1 de la guía práctica.
- Lectura recomendada: Remote sensing digital Image Analysis - John A. Richards. Capítulos 1 y 3

2/5 Entrega del cuestionario 1. Encuesta de inicio del curso.

3/5 Corrección radiométrica

- Clase teórica: Transferencia radiativa. Métodos estadísticos y modelado de la atmósfera.
- Clase práctica: capítulo de 2 la guía práctica.
- Lectura recomendada: Remote sensing digital Image Analysis - John A. Richards. Capítulos 2

9/5 Entrega del cuestionario 2.

10/5 Cálculo de índices espetrales

- Clase teórica: Cálculo de índices espetrales
- Clase práctica: capítulo de 3 la guía práctica.
- Lectura recomendada: Quantitative Remote Sensing - ShunLin Liang. Capítulos 8

16/5 Entrega del cuestionario 3.

17/5 Rotaciones espetrales

- Clase teórica: Transformada Tasseled Cap y Análisis por componentes principales.
- Clase práctica: capítulo de 4 la guía práctica.
- Lectura recomendada: Remote sensing digital Image Analysis - John A. Richards. Capítulos 6

23/5 Entrega del cuestionario 4.

24/5 Clase de consulta.

30/5 Entrega del trabajo práctico 1.

31/5 Clasificación no supervisada

- Clase teórica: Métodos de clasificación no supervisados.
- Clase práctica: capítulo de **5** la guia práctica.
- Lectura recomendada: Remote sensing digital Image Analysis - John A. Richards. Capitulos 9

6/6 Entrega del cuestionario 5.

7/6 Clasificación supervisada

- Clase teórica: Métodos de clasificación supervisados.
- Clase práctica: capítulo de **6** la guia práctica.
- Lectura recomendada: Remote sensing digital Image Analysis - John A. Richards. Capitulos 8

13/6 Entrega del cuestionario 6.

14/6 Técnicas pos-clasificación

- Clase teórica: Tecnicas pos-clasificación.
- Clase práctica: capítulo de **7** la guia práctica.
- Lectura recomendada: Making better use of accuracy data in land change studies: Estimating accuracy and area and quantifying uncertainty using stratified estimation - Olofsson et al.

20/6 Entrega del cuestionario 7.

21/6 Clase de consulta.

27/6 Entrega del trabajo práctico 2.

30/6 Encuesta de fin del curso.

Apéndice B

Categorías de uso y cobertura del suelo

Categorías de uso y cobertura según el esquema LCCS2 de la FAO. Los colores son sugerencias por categoría.

| Nombre | Código | Color |
|---|--------|---------|
| Áreas terrestres cultivadas y manejada | A11 | #b2df8a |
| Vegetación natural y semi-natural | A12 | #33a02c |
| Áreas acuáticas o regularmente inundadas cultivadas | A23 | #fdbf6f |
| Vegetación natural y semi-natural acuática o regularmente inundadas | A24 | #ff7f00 |
| Superficies artificiales y áreas asociadas | B15 | #fb9a99 |
| Áreas descubiertas o desnudas | B16 | #e31a1c |
| Cuerpos artificiales de agua, nieve y hielo | B27 | #a6cee3 |
| Cuerpos naturales de agua, nieve y hielo | B28 | #1f78b4 |

Tabla B.1 – Categorías usos del suelo según el esquema LCCS2 de la FAO.

Apéndice C

Instalacion de qgis y R

Veamos como instalar Q-GIS y R en Windows 10 y Ubuntu 16.04. De la misma forma deberia poder instalarse en otro sistemas operativos (Windows 7, 8 y 8.1) y GNU/Linux.

C.1. Instalación en Windows 10

Para realizar la instalacion en Windows 10, descargamos e instalamos en el siguiente orden cada uno de los programas a utilizar.

1. Q-GIS 2.14 o 2.18 de la pagina oficial de QGIS <http://www.qgis.org/es/site/>.
2. R 3.3 o 3.4 de la pagina de oficial del proyecto R <https://cran.r-project.org/bin/windows/base/>.
3. R-Studio en su version gratuita de la pagina oficial <https://www.rstudio.com/products/rstudio/download/>.

Siempre en sus instalaciones por defecto y debemos instalarlas en el orden indicado.

Una vez finalizada, abrimos R-Studio y ejecutamos el siguiente comando en la consola.

```
1 install.packages(c("raster", "lattice", "RStoolbox", "rasterVis", "rgdal", "e1071", "randomForest", "kernlab"))
```

El mismo descargara todos los paquetes que utilizaremos durante el curso y los instalara. Si la instalación termina sin ningun error, estamos listos para comenzar a trabajar.

C.2. Instalación GNU/Linux

Para instalar los programas necesarios en linux debemos seguir los siguientes tutoriales para instalas Q-GIS, R y R-Studio.

1. Q-GIS seguimos las instrucciones para instalar Q-GIS 2.14 o 2.18 <http://www.qgis.org/es/site/forusers/alldownloads.html#linux>.
2. R segun nuestra distribucion. Para el caso de Ubuntu podemos utilizar `sudo apt install r-base` y para Fedora podemos utilizar `yum install R`. En caso de tener otra distribucion preguntar en el foro.
3. R-Studio en su version gratuita de la pagina oficial <https://www.rstudio.com/products/rstudio/download/>.

Una vez finalizada la instalacion, abrimos R-Studio y ejecutamos el siguiente comando en la consola.

```
1 install.packages(c("raster","lattice","RStoolbox","rasterVis","rgal","e1071","randomForest","kernlab"))
```

En este caso descargara y recompilara cada uno de los paquetes necesarios para nuestra distribución.

C.3. Máquina virtual

En caso de no querer realizar la instalación, es posible utilizar la máquina virtual provista por el [Boston Education in Earth Observation Data Analysis](#) siguiendo las instrucciones en su repositorio de github <https://github.com/bgeeoda/opengeo-vm>.