

Introducción

1. Un viaje del sol a los pixeles.

En esta primera práctica nos familiarizaremos con las interfaces gráficas del qgis y de R-studio. Para esto analizaremos 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 archivos raster y vectoriales 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`. Esta imagen corresponde al departamento de Iguazú en la provincia de Misiones. Esta fue obtenida por el satélite Landsat 8 durante el mes de noviembre de 2016.

Para esto vamos al menú *Capa → Añadir capa → Añadir capa ráster*. Navegamos hasta la carpeta `raster_data/LC82240782016304` y abrimos el archivo `LC82240782016304LGN00.vrt`. Una vez abierto el mismo podremos encontrarlo en el *Panel de capas* de qgis desde donde podremos cambiar las opciones de visualización y estudiar sus propiedades.

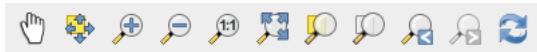


Figura 1 – Herramientas para moverse dentro de la imagen. De izquierda a derecha: 1. Desplazar mapa, 2. Desplazar mapa a la selección, 3. Acercar zoom, 4. Alejar zoom, 5. Zoom a la resolución nativa, 6. Zoom general, 7. Zoom a la selección, 8. Zoom a la capa, 9. Zoom anterior, 10. Zoom siguiente, 11. 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*. Dentro de las propiedades 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 referencias de coordenadas entre otros.

Podemos ir luego a la pestaña de *Estilo* para cambiar la visualización de la capa. Allí podemos elegir de qué color mostraremos cada una de las bandas además de cambiar el realce. Una vez elegidas las bandas debemos hacer click en el botón *Cargar* para seleccionar los valores máximos y mínimos de las bandas para el realce.

La herramienta *Identificar un objeto espacial* nos permite extraer valores de una coordenada espacial de nuestra imagen. Al habilitarla y hacer click sobre un punto de la imagen veremos datos de la misma como por ejemplo los valores de reflectancia del pixel seleccionado. Dichos valores pueden mostrarse como Árbol, Tabla o Grafo según como sea más útil.

Actividad 1.1. Cambie la combinación de bandas de la imagen L8 a color real y explorela. Identifique zonas de coberturas uniformes. Pruebe cambiar la combinación de bandas y decida si dichas zonas siguen siendo uniformes después de cada cambio.

Actividad 1.2. Encuentre el sistema de coordenadas en el cual se encuentra la imagen. ¿Cuántas filas y columnas tiene la imagen?

Actividad 1.3. Utilizando la herramienta identificar objetos espaciales encuentre los valores de reflectancia de distintas coberturas. Grafique estos valores en una firma espectral y en el espacio de fases nirrojo.

¹Es un buen momento para ponerle uno más sencillo

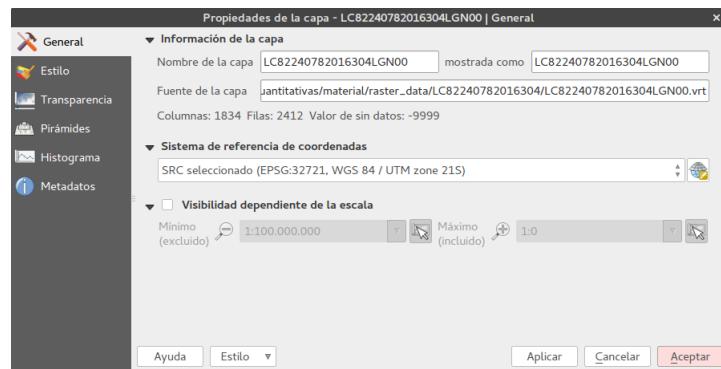


Figura 2 – Pestaña general de propiedades de una capa. En la misma se pueden ver los datos mas importantes sobre la misma como la cantidad de filas y columnass, el nombre y el sistema de referencia de coordenadas.

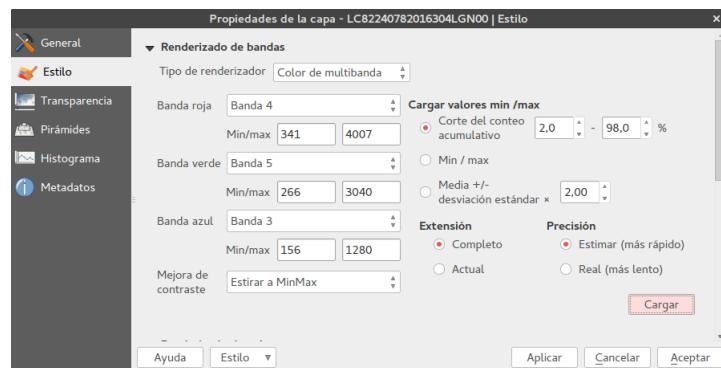


Figura 3 – Estilos de visualizacion de una capa raster. Los estilos posibles son: 1. Color de multibanda, 2. En paleta, 3. Unibanda gris, 4. Unibanda pseudocolor. Puede explorar cada uno por separado ya que todos tendran distintas utilidades.

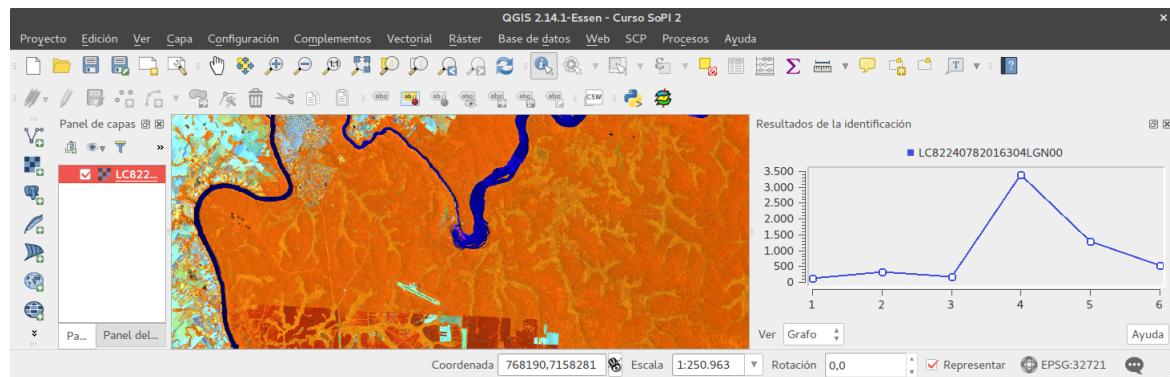


Figura 4 – Identificacion de un pixel correspondiente a la selva paranaense mostrada como grafo.

1.2. Creacion de capas vectoriales

Veamos ahora como crear capas vectoriales. Nos van a ser principalmente de utilidad 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 boton del mismo nombre que se encuentra en el panel lateral. Podemos agregar los campos que sean necesarios para nuestra capa vectorial. En este caso crearemos 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. La guardamos en la carpeta `vector_data/` con el nombre `firmas.shp`.

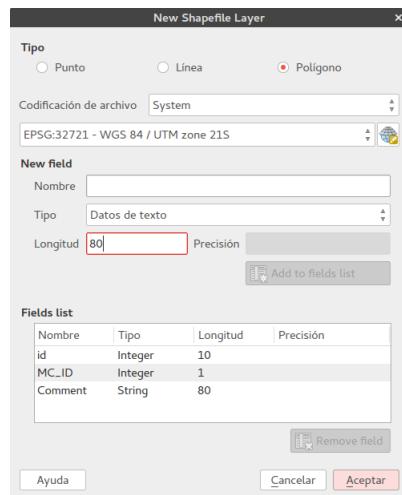


Figura 5 – Creacion de una nueva capa vectorial. Se agregan campos que seran de interes para comparar las firmas espectrales.

Una vez creada la nueva capa podemos utilizar la barra de herramientas de qgis para agregar nuevas geometrias a la misma. Para esto hacemos click en el boton de agregar geometrica y digitalizamos una zona uniforme dentro de la imagen.



Figura 6 – Herramientas de edición vectorial. De izquierda a derecha: 1. Conmutar 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.

Al terminar de acerlo qgis pedira un numero 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.

Es importante recordar que debemos estar en el modo de edicion para poder hacer esto y salir de al terminarla.



Figura 7 – Valores de los campos del nuevo polígono creado.

Actividad 1.4. Digitalize coberturas uniformes dentro de la imagen. Recuerde obtener al menos una por cada categoria de uso y cobertura presente dentro de la misma.

En caso de desear cambiar la visualizacion de la capa vectorial, podemos entrar a las propiedades de la misma². Ademas podemos acceder a la tabla de datos de la capa vectorial haciendo click derecho sobre la misma y eligiendo la opcion *Abrir tabla de atributos*.

1.3. Exploracion raster en R

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

Ademas, deberemos situar nuestra carpeta de trabajo donde se encuentran las carpetas que descargamos. Para esto nos movemos en el explorador de archivos hasta la misma y hacemos click en usar la carpeta como carpeta de trabajo.

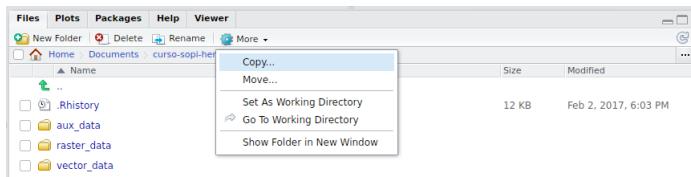


Figura 8 – Configuracion del directorio de trabajo desde la interfaz grafica.

Tambien podemos utilizar el comando `setwd(.)` para configurar el directorio de trabajo.

Una vez en dicha carpeta, existen varias maneras de abrir una imagen segun queramos hacerlo solo para una banda, varias bandas en archivos separados o un solo archivo multibanda.

Los comandos para esto son `raster`, para abrir una unica banda, `brick`, para abrir un archivo multibanda, y `stack` para abrir distintas bandas por separado. Veamos algunos ejemplo de esto:

Ejemplo 1.1. 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 text

```
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 el podemos ver la clase a la que corresponde el archivo abierto, en este caso un *RasterBrick*, las dimensiones, el tamaño de pixel, extension de la capa, proyeccion, cual es la ruta al archivo que abrimos, las bandas y sus valores maximos y minimos.

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

Trabajemos ahora con este raster. Vamos a cambiarle el nombre a las bandas y convertirla a reflectancia entre 0 y 1.

```
1 ref.2016 <- brick(filename)
2 names(ref.2016) <- c("blue", "gree", "red", "nir", "swirl", "swir2")
3 ref.2016 <- ref.2016/1e4
4 rasterOptions(addheader = "ENVI")
5 writeRaster(ref.2016, "raster\_data/processed/ref2016")
```

Analicemos el codigo linea por linea.

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

podemos ademas graficar tanto una combinacion de bandas en qgis

```
1 plotRGB(ref.2016, r=4,g=5,b=3, stretch='lin')
```

Obtenemos como resultado como tambien todas las bandas por separado



Figura 9 – Combinacion de bandas nir-swirl-red en R.

```
1 plotRGB(ref.2016)
```

obtenemos como resultado

Actividad 1.5. Abra el archivo vrt en qgis y vuelva a mirar la firma espectral para distintas coberturas. ¿Entre que valores se encuentra ahora las mismas?

Ejemplo 1.2. Hagamos un poco de analisis ahora sobre la imagen. En primer lugar podemos calcular la estadistica basica sobre la imagen. Para ello ejecutamos el comando `summary(ref.2016)` obtenemos como resultado

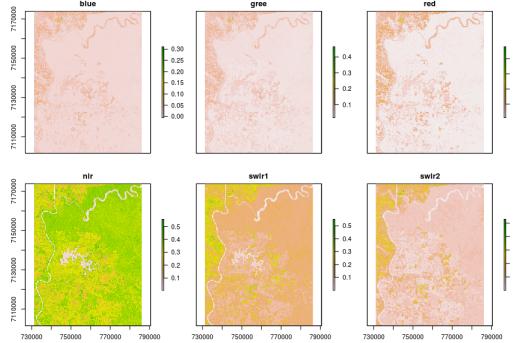


Figura 10 – Grafico de bandas con realce automatico para cada una.

	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

Para comenzar podemos calcular los histogramas de todas las bandas con el comando `hist(ref.2016)` y el scatter plot entre dos bandas como `plot(l8red, l8blue)`

En caso de querer todos los scatterplots e histogramas en un solo grafico podemos hacerlo con el comando `pairs(18)`.

1.4. Manejo vectorial en R

Hasta ahora estamos analizando la imagen completa. Podemos sin embargo analizar solo sectores concretos de la imagen muestreandola en funcion de un archivo vectorial. Tambien sera posible muestrear la imagen pos zonas definidas por otro raster pero veremos esto mas adelante.

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

Ejemplo 1.3. Veamos como realizar el análisis básico de un vector en R. Comenzamos leyendolo

```
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 como resultado

```
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
```

Podemos graficar los vectores obtenidos en R junto aa la imagen de base como

```

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

```

donde la primera linea grafica la imagen de fondo y la segunda agrega el el shapefile sobre la misma.

Actividad 1.6. Muestre las propiedades de la capa raster y el vector abiertos y verifique que los mismos se encuentren en el mismo sistema de coordenadas.

Por ultimo mostremos como extraer datos de un archivo raster y veamos un par de ejemplo concretos. La funcion que nos permite extraer datos de un raster segun un vector es **extract** que toma dos argumentos, el vector que queremos utilizar y la capa raster sobre la cual hacer la consulta.

Veamos algunos ejemplos

Ejemplo 1.4. 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 extraccion 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 = ".")

```

obteniendo como resultado

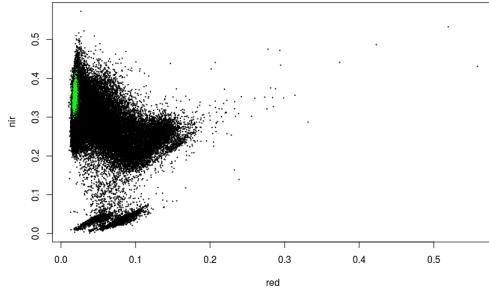


Figura 11 – Resultado del scatterplot para las bandas roja y nir. Se muestra en verde datos correspondientes a la selva paranaense.

La funcion **extract** nos permite tambien aplicar una funcion a los datos extraidos antes de entregarlos al usuario. Veamos como usarla para calcular datos de interes sobre las coberturas y guardarlos en un archivo vectorial.

Ejemplo 1.5. Extraer los promedios y desvios standar de un raster y agregarlos a un vector. Primero extraemos los valores de promedio y desvio

```

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

```

renombramos luego las columnas como promedio y devio 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")
```

Por ultimo, veamos como usar una capa vectorial para graficar para extraer las firmas especiales y graficarlas para distintas coberturas

Ejemplo 1.6. Graficar las firmas espectrales en funcion de la longitud de onda para cada geometria de un vector. Utilizaremos en este caso dos nueva libreria, `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 nanometros. Luego reformamos el dataframe para que podamos subsetearlo, 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")
```

si mostramos el dataframe, el resultado debería ser similar al siguiente

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 los desvios 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 sera 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
...			

Veamos algunas opciones para generar ahora los graficos. En primer lugar pondremos todas las firmas juntas, separadas por color, usando la libreria **lattice**

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

Aqui la primer linea dice que grafiquemos la reflectancia como funcion de la longitud de onda, obteniendo los datos del dataframe df y agrupandolos segun la columna cobertura. La siguiente linea agrega la leyenda en la parte superior de la figura y con 4 columnas. Por ultimo en la tercera linea pedimos que el grafico tenga lineas y puntos. Si queremos agruparlo por categoria de uso y

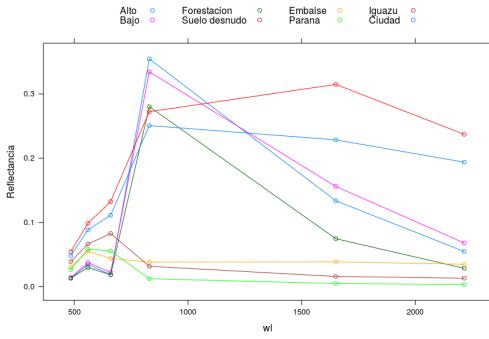


Figura 12 – Firmas espetrales

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"))
```

y finalmente si queremos graficar solo un subset de los daatos

```
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"))
```

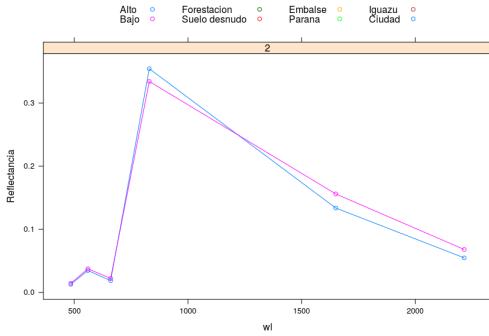


Figura 13 – Firmas espetrales

Actividad 1.7. Grafique la media y el desvio standar para las distintas coberturas que pudo identificar en el punto uno.

2. Rebotando por la atmosfera

En esta segunda actividad practica nos centraremos en la corrección radiométrica de imágenes satelitales. Son objetivos de la misma

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

2.1. Calculo de reflectancia a tope de la atmósfera

Para poder convertir una imagen a reflectancia a tope de la atmósfera vamos a necesitar no solo la imagen sino también la información adicional que hallaremos en su metadato.

Para abrir una imagen satelital desde el metadato utilizaremos las funciones disponibles en **RStoolbox**. Este incluye diversas herramientas para trabajar con imágenes satelitales.

Ejemplo 2.1. Comencemos analizando un ejemplo sencillo, abriremos la imagen Landsat 7 del año 2000 desde el metadato y la mostraremos en combinación de bandas de falso color compuesto, además de analizar las propiedades básicas de la misma.

```
1 meta.2000 <- readMeta("raster_data/LE72240782000188EDC00/LE72240782000188EDC00_.MTL.txt")
```

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

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

A partir del metadato podemos cargar la imagen completa con el comando `stackMeta`. Además eliminaremos 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
```

obtenemos como resultado 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
```

y podemos mostrar la iamgen como hicimos antes

```
1 plotRGB(dn.2000, r=3, g=2, b=1, stretch="lin")
```



Figura 14 – Imagen en combinacion color real de la zona de interes sobre la imagen en DN Landsat 7.

De esta forma podemos tener el archivo cargado en DN con todos sus metadatos para convertirlo a reflectancia reflectancia y realizar distintas correcciones. Para pasar nuestra imagen a reflectancia a tope de la atmosfera tenemos dos maneras de hacerlo. Podemos hacerlo a mano utilizando las herramientas algebraicas de R o podemos hacerlo con la funcion especifica de **RStoolbox**.

Ejemplo 2.2. Calculo de reflectancia a tope de la atmosfera 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 metadatos los parametros de calibracion en reflectancia y el angulo de elevacion solar.

Convertimos luego la imagen a reflectancia y la dividimos luego por el angulo 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","swirl","swir2")
```

Otra forma forma de realizar este proceso es utilizando la funcion **radCor**. En este caso debemos dar la imagen en DN, el metadato y cual es la cantidad que queremos calcular.

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

podemos comparar los resultados de ambos metodos inspeccionando los objetos **toam.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, swir2
min values  : -0.01976113, -0.02181530, -0.02029439, 0.01934678, -0.02781926, -0.02678077
max values  : 0.6106812, 0.5609009, 0.6079443, 0.8696885, 0.8640919, 0.8263815

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, B7_tre
min values  : 0.00000000, 0.00000000, 0.00000000, 0.01934678, 0.00000000, 0.00000000
max values  : 0.6106812, 0.5609009, 0.6079443, 0.8696885, 0.8640919, 0.8263815

```

Actividad 2.1. Inspeccione la reflectancia a tope de la atmosfera para todas las bandas. Para esto realice los histogramas, graficos de dispersion, calcule la media, el desvio standar y cualquier otra medida estadistica que le guste.

2.2. Calculo de reflectancia corregida atmosfericamente por metodos estadisticos

La funcion `radCor` dispone de un parametro para hacer distintas tipos de correcciones atmosfericas. Ya vimos `apref` que nos permitio calcular la reflectancia a tope de la atmosfera. Veamos como aplicar el metodo de substraccion de cuerpo oscuro.

Ejemplo 2.3. Apliquemos el metodo de *simple dos* para corregir la imagen. En este caso solamente restaremos el minimo 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 corregimos la imagen luego 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 analisis de lo que pasa en la situacion, vamos a graficar los histogramas de cada banda para la imagen en reflectancia TOA y corregida por el metodo simple dos. Para esto 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 atmosfera",
4                                         "Correccion 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                     key=simpleKey(text=c("Tope de la atmosfera",

```

```

9           "Correccion Simple DOS") ,
10          lines=TRUE, points=FALSE))
11 B3 <- densityplot(~B3_tre+B3_sre , data=toa.boa, xlab="Reflectancia",
12                      ylab="", main="Banda roja", plot.points=FALSE, xlim=c(0,0.3),
13                      key=simpleKey( text=c("Tope de la atmosfera",
14                               "Correccion Simple DOS"),
15                               lines=TRUE, points=FALSE))
16 B4 <- densityplot(~B4_tre+B4_sre , data=toa.boa, xlab="Reflectancia",
17                      ylab="", main="Banda nir", plot.points=FALSE, xlim=c(0,0.3),
18                      key=simpleKey( text=c("Tope de la atmosfera",
19                               "Correccion Simple DOS"),
20                               lines=TRUE, points=FALSE))
21 print(B1, split = c(1, 1, 2, 2), more=TRUE)
22 print(B2, split = c(2, 1, 2, 2), more=TRUE)
23 print(B3, split = c(1, 2, 2, 2), more=TRUE)
24 print(B4, split = c(2, 2, 2, 2), more=FALSE)

```

En este caso las primeras 4 funciones crean los histogramas para cada banda corregida mientras que las ultimas 4 lineas los imprimen en una grilla.

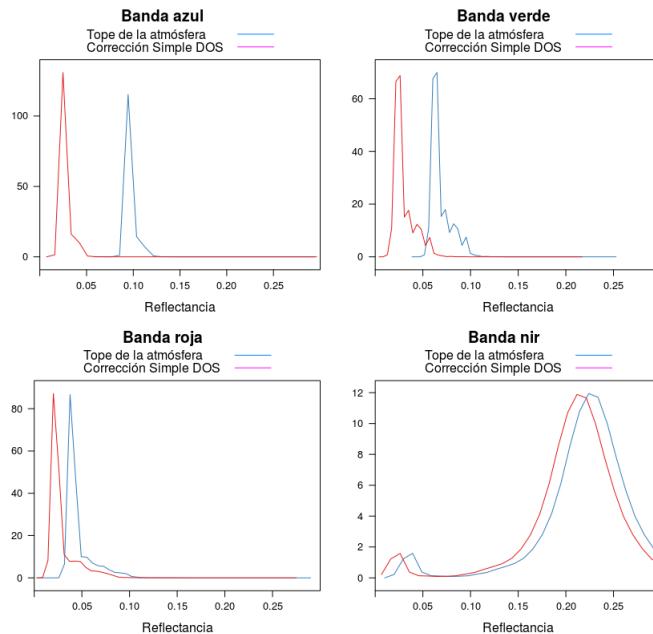


Figura 15 – Graficos de los histogramas para las distintas bandas donde se muestra el nivel de correccion en cada una.

Notamos en este caso que la correccion se vuelve menos importante a medida que crece la longitud de onda.

Actividad 2.2. Analice los valores de haze obtenidos por la funcion stimate haze y grafiquelos como funcion de la longitud de onda en escala logaritmica. ¿Que observa?

Actividad 2.3. Utilice el metodo costz para corregir la imagen a reflectancia a tope de la superficie.

Actividad 2.4. Guarde los archivos raser generado por cada uno de los metodos de correccion. Abralos en qgis y comparelos visualmente. Obtenga firmas espectrales con los distintos metodos de correccion.

2.3. 6S

Veamos ahora como operar con el 6S para obtener una estimacion de los parametros atmosfericos. Para esto utilizaremos la version web del 6S que se encuentra disponible en <http://6s.ltdri.org/pages/run6SV.html>.

Para utilizarla ingresaremos a la pagina y haremos click en el boton *Submit query*. Iremos luego configurando paso a paso nuestro modelo de la atmosfera haciendo siempre luego click en el boton *submit query* para pasar 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 target 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*. Del mismo debemos extraer 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 codigo 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 2.5. Realice una extraccion de firmas espectrales para distintas coberturas de cada uno de los archivos raster obtenidos y grafiquelos en el mismo grafico. Comparela con la firma espectral obtenida a partir de la imagen corregida por el usgs.

Actividad 2.6. Haga un grafico de densidades que muestre los distintos metodos de correccion atmosfericos para cada banda.

Actividad 2.7. Calcule la diferencia promedio para cada banda entre las imagenes en reflectancia a tope de la atmosfera y las distintas correcciones y la imagen en reflectancia entregada por el USGS.

3. Un abaco espectral

En esta tercer practica comenzaremos a trabajar con operaciones matematicas entre las bandas del sateltica y en el uso de los resultados para obtener relaciones empiricas con las variables biofisicas medibles en el terreno. Son objetivos de las mismas

- Poder calcular los indices de vegetacion a partir de las imágenes en reflectancia.
- Poder calcular la linea de suelo como parametro para calcular índices de vegetación.
- Poder realizar modelos empiricos que relacionen variables biofisicas medidas a campo con los indices calculados.
- Construir mapas a partir de los modelos empiricos antes mencionados.

3.1. Calculo de indices entre bandas

Comenzamos trabajando con el calculo de indices entre bandas. Para ello utilizaremos lass lirberias `raster` y `RStoolbox`. Ademas para poder usar mejores paletas de colores utilizaremos la libreria `RColorBrewer`, sin embargo el uso de la misma es optativo. Por ultimo puede ayudar cargar la libreria `rasterVis` para realizar algunos de los graficos.

Comenzamos primer cargando la imagen desde el metadato y convirtiendola 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","swirl1","swirl2")
```

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

Ejemplo 3.1. Calculo de NDVI a mano y grafico del mismo.

```
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

- Primero calculando el ndvi a mano.
- Segundo obteniendo la rampa de color con 16 valores entre amarillo y verde.
- Por ultimo graficando el ndvi, utilizando como colores dicha rampa y ajustandolo entre 0 y 1, es decir, los valores menores a 0 se mostraran todos del mismo color.

Obtenemos entonces

El paquete `RStoolbox` tiene varias herramientas que nos ayudan a calcular los indices especiales. Veamos por ejemplo como calcular el NDVI y el EVI utilizando dicho paquete

Ejemplo 3.2. Para calcular los indices mediante la funcion `spectralIndices` debemos especificar con que raster trabajamos y que bandas corresponden a cada longitud de onda

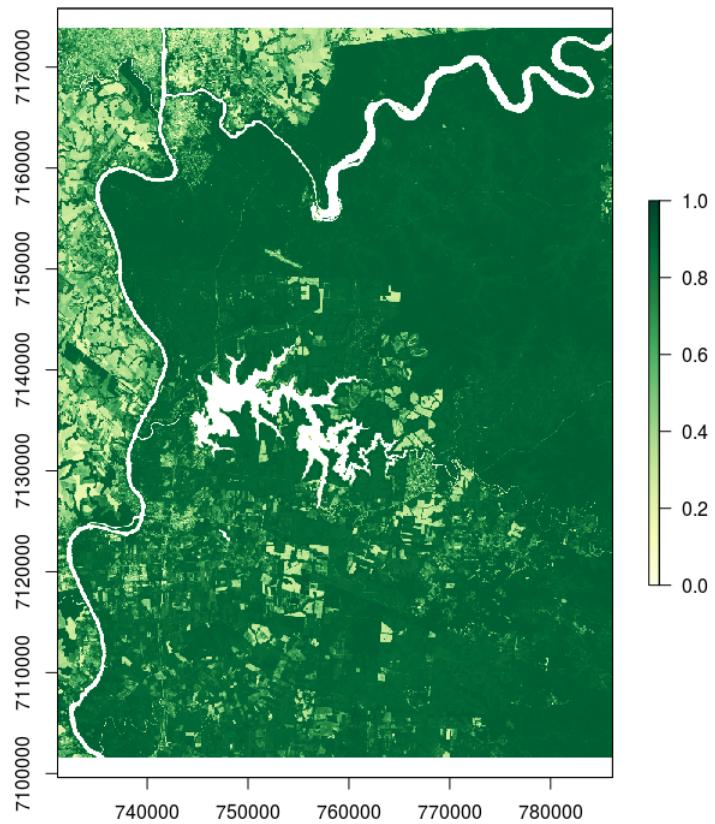


Figura 16 – Mapa del NDVI en escala de verdes.

```

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

- La imagen con la cual vamos a trabajar.
- A qué zona del espectro corresponde cada banda.
- Los índices que queremos calcular

Actividad 3.1. Calcule el NDVI y el EVI para el año 2000 utilizando la imagen landsat 7.

Actividad 3.2. Calcule y grafique todos los índices posibles que involucren a las bandas roja y nir de landsat 8. Puede ayudarse con el comando `?spectralIndices`

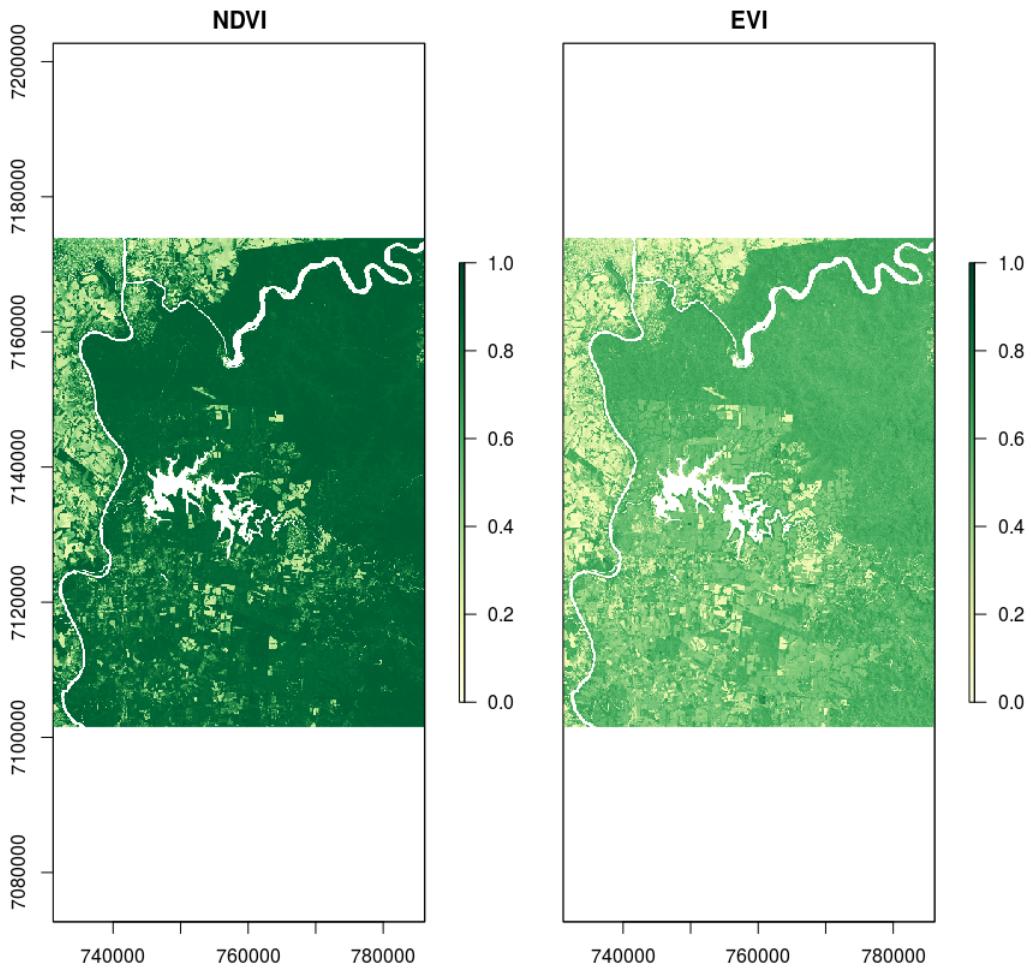


Figura 17 – Graficos del EVI y el NDVI para la imagen seleccionada.

3.2. Calculo de la linea de suelo

La linea de suelo es una cantidad que definimos en teledeteccion que aporta informacion sobre las imagenes que estamos analizando. Ademas sirva para incorporar los efectos de la reflectancia del suelo en el calculo de indices. Veamos como hacerlo utilizando la libreria `landsat`

Ejemplo 3.3. Calculemos el tSAVI utilizando la linea de suelo obtenida a partir de la imagen. Para esto necesitaremos enmascarar las zonas con cobertura de agua y nubes. Veamos primer como hacer esto.

```

1 mask.2016 <- raster("raster_data/LC82240782016304/LC82240782016304LGN00_cfmask.tif")
2 masked.2016 <- mask(ref.2016, mask=mask.2016, inverse=TRUE,
3                         maskvalue=0, updatevalue=255)
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 linea 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 grafico de la linea de suelo. Podemos consultar los demas parametros imprimiendo la variable `bsl.2016`.

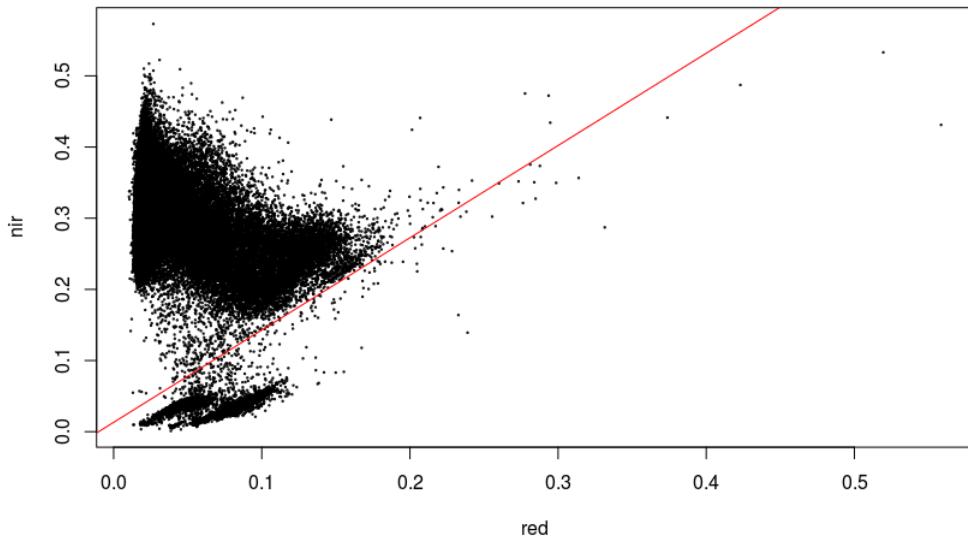


Figura 18 – Linea de suelo sobre el scatterplot nir-red

Actividad 3.3. Calcule el tSAVI utilizando la linea de suelo obtenida arriba.

Actividad 3.4. Vuelva a obtener la linea de suelo sin enmascarar la imagen y dibuje el scatterplot con la misma y la anterior. Que problema encuentra.

Actividad 3.5. Obtenga la linea de suelo y calcule el indice tSAVI para la imagen del año 2000.

3.3. Estimacion de parametros biofisicos

Finalmente, veamos como se puede obtener datos biofisicos a partir de los indices de vegetacion calculados. De esta forma podremos generar mapas de porcentaje de cobertura, productividad, etc.

Ejemplo 3.4. Comenzamos calculando el ndvi para el año 2016, y utilizando la capa muestreo hacemos una extraccion estadistica sobre la misma.

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

Obtendremos un grafico que presenta los scatterplots entre las bandas, su correlacion e histogramas. Veamos en el mismo que la superficie cubierta por vegetacion varia linealmente con el NDVI. Por lo tanto utilizaremos estos para hacer un ajuste de nuestro modelo.

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

de esta forma veremos los parametros de nuestro ajuste, y graficaremos al mismo en un scatterplot.

Para aplicar el modelo a nuestro raster hacemos

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

Obteniendo el mapa de abajo.

Actividad 3.6. Genere los modelos de lai, fapar y fcover para el año 2016 y con los mismos realice mapas de dichas variables.

Actividad 3.7. Utilizando los modelos obtenidos para 2016 aplique los mismos para obtener los mapas de lai, fapar y fcover del año 2000. ¿Que suposicion esta haciendo? Compare distintas zonas de los modelos en el qgis.

4. Rotaciones espectrales

En nuestra cuarta practica trabajemos con rotaciones en el espacio espectral. A puntamos con las mismas a poder incorporar conceptos como dimensionalidad y correlacion entre bandas que nos ayuden a utilizar mejor la informacion satelital.

Son objetivos de la misma

- Aplicar la transformada tasseled cap a una imagen multiespectral e interpretar el significado de cada banda.
- Poder aplicar la transformada por componentes principales a una imagen multiespectral e interpretar el resultado.
- Poder aplicar la transformada por componentes principales a un stack de bandas multiespectrales de distitnas fechas e interpretar los resultados.
- Utilizar la transformada por componentes principales para extraer informacion de series temporales de indices.

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

Comencemos calculados la transformada tasseled cap para una imagen landsat. La misma sera la primer rotacion que utilizaremos en el curso y en ella la interpretacion es bastante sencilla. Para ello usaremos el paquete **RStoolbox**.

Ejemplo 4.1. Para cacular la transformada por componentes principales comenzamos abriendo la imagen landsat desde el metadado y convirtiendola a reflectancia a tope de la atmosfera 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","green","red","nir","swirl1","swirl2")
```

calculamos analizamos ahora el espacio red-nir y green-nir 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)
```

obteniendo como scatterplots

Calculemos ahora la transformada tasseled cap. Para eso usamos la funcion **tasseledCap** del paquete **RStoolbox**.

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

Obtenemos una imagen de tres bandas, *brillo, verdor y humedad*.

podemos graficar cada una de las bandas por separado con el comando

```
1  plot(tsc.2016)
```

o todas juntas con

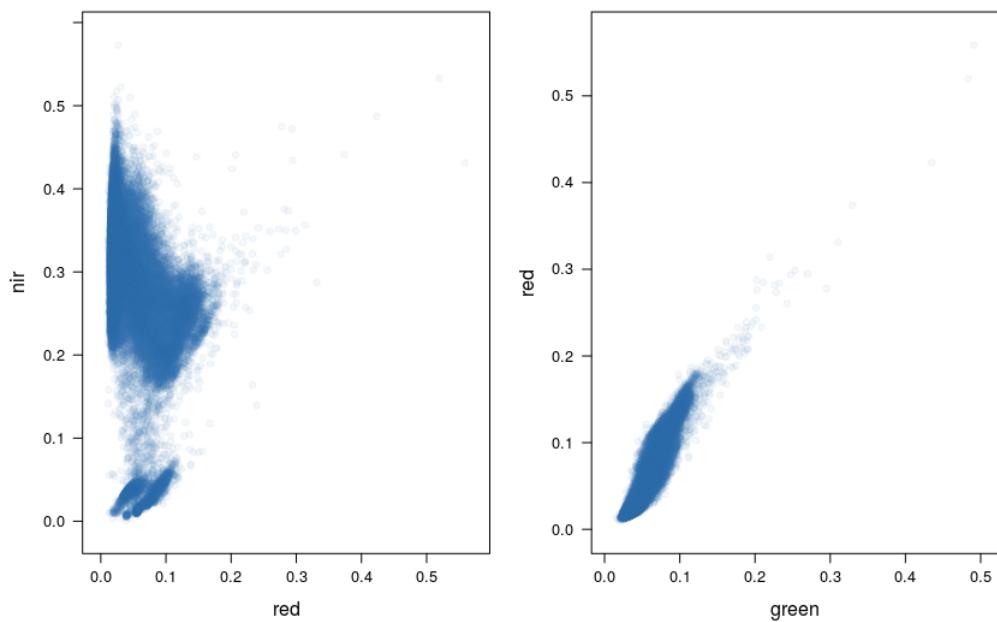


Figura 19 – Scatterplot verde-rojo y nir-red.

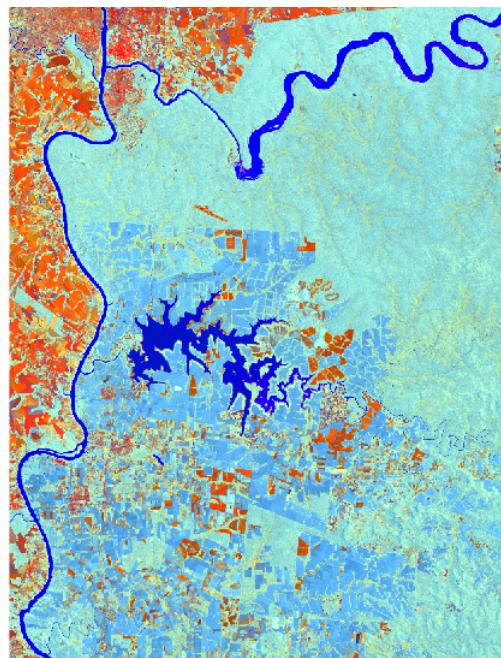


Figura 20 – Imagen de la transformada tasseled cap.

```
1   plotRGB( tsc.2016 ,r=1,g=2,b=3, stretch="lin" )
```

Vemos en este caso que distintas zonas de la selva presentan distinto color en cyan debido al cambio en contenido de humedad, que las zonas sin vegetación se ven en gradientes de rojo igual

que las ciudades y las zonas cubiertas de agua se ven en azul inenso.

Actividad 4.1. Calcule la transformada tasseled cap para la imagen landsat 7 del año 2000.

4.2. Calculo de la transformada por componentes principales

Veamos ahora como calcular la transformada por componentes principales. Para esto utilizaremos la herramienta `rasterPCA` del paquete `RStoolbox`. Realizaremos ademas un analisis previo sobre las mismas para ayudar a comprender como funciona dicha transformada.

Ejemplo 4.2. Comencemos analizando la transformada por componentes principales de la imagen de 2016. ¿Que podemos predecir? Miremos primero los scatterplots `pairs`(ref. 2016)

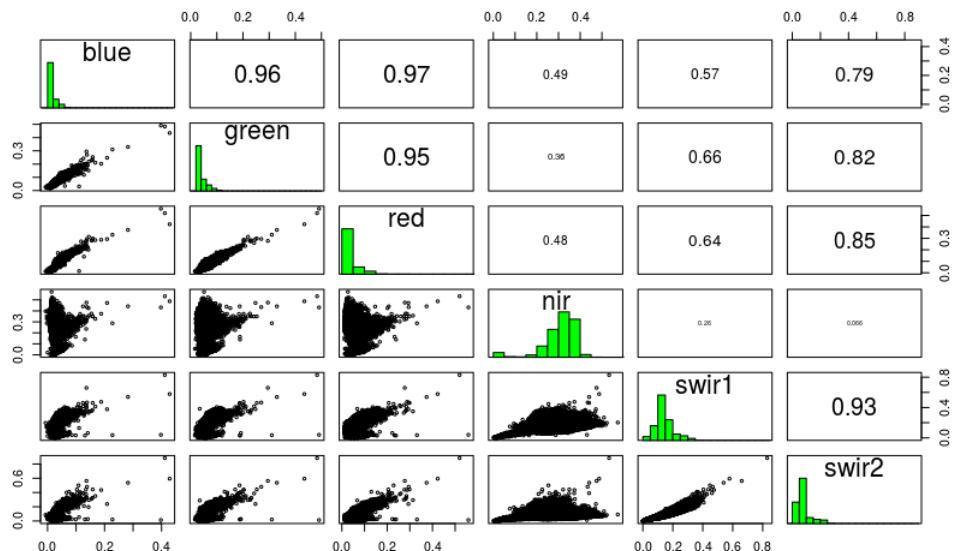


Figura 21

Mirando el resumen de la imagen vemos que hay varias bandas muy correlacionadas entre si. Por ejemplo las del visible, mientras que otras lo estan poco, por el ejeplo el nir y el swir. 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 mas bajo.

Apliquemos entonces la transformada por componentes principales y veamos que sucede

```

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

```

Veamos el sumario del modelo obtenido,

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 analizar las varianzas, vemos que las 3 primeras explican mas 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 alcanzan para explicar casi todo el comportamiento. Analisemos la primera, usamos el comando `loadings(pca.2016$model)` para ver como son las componentes.

```

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

```

la primer componente pesa, segun la banda pero siempre con el mismo signo, a todas las demas componentes. Podemos interpretarla por lo tanto como un brillo y esperar que sea mas alta cuando miramos zonas de la imagen que tengan mas reflectancia en promedio

Si mostramos las tres primeras componentes en combinacion RGB vemos que distintos tipos de coberturas se distinguen de distinta manera,

```
1 plotRGB(pca.2016$map, r=1,g=2,b=3, stretch="lin")
```

Vale la pena aclarar en este caso que debemos referenciar el mapa dentro del objeto pca y no directamente el objeto para graficarlo.

Actividad 4.2. Analice la segunda componente de la transformada por componentes principales para la imagen Landsat 8.

Actividad 4.3. Calcule y analice la transformada por PCA de la imagen Landsat 7 del año 2000.

4.3. Algunas ideas sobre series temporales

Empecemos esta seccion con una actividad para que realicen

Actividad 4.4. Aplique la transformada por componentes principales al stack de bandas del año 2000 y 2016.

Veamos que pasa al trabajar con series temporales de indices.

Ejemplo 4.3. Otra aplicacion de la transformada por componentes principales por componentes principales es el analisis 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 llenados los espacios donde no habia datos podemos abrir la imagen en qgis y realizar consultas de pixel en distintas zonas de la misma.

Utilizando la herramienta de identificar objetos espaciales podemos consultar como es el comportamiento de la serie temporal para cada pixel. vemos que distintas zonas tienen distintos comportamiento intra e interanual.

Podemos analizar el promedio y el desvio standar para cada pixel 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)

```



Figura 22

Actividad 4.5. Grafique las primeras 4 componentes por de la transformada por componentes principales de la imagen del stack de NDVI. Que zonas puede identificar en la primera? que zonas se distinguen en la segunda? que comportamiento encuentra en la tercera y cuarta.

Actividad 4.6. Grafique en el scatter-plot la imagen completa y marque en el mismo zonas con vegetacion, agua y suelo sin cobertura vegetal. Vea como cambian estas zonas frente a las transformadas por componentes principales y tasseled cap.

5. Clasificacion supervisada de imagenes

En esta practica seguiremos trabajando con la clasificacion supervisada de imagenes satelitales. Utilizaremos mas paquetes en este caso.

```
1 library(RStoolbox)
2 library(rgdal)
3 library(raster)
4 library(rasterVis)
```

ademas de los paquetes que incorporan los distintos metodos de clasificacion

```
1 library(caret)
2 library(randomForest)
3 library(e1071)
4 library(kernlab)
```

comenzamos abriendo la imagen del año 2016 para las bandas reflectivas como en la clase anterior. Abrimos tambien el vector de entrenamiento.

Empecemos con la clasificacion por el metodo de maxima verosimilitud

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

y realizar el scatterplot de dichas variables como.

```
1 ref.2016 <- sup.2016
2 xyplot(nir~red, groups=mlc, data=ref.mlc)
```

Cambiando el algoritmo de clasificacion en el parametro `model` podemos calcular distintas clasificaciones supervisadas. Algunas de las vistas en clase son `rf`, `svmRadial`, `kNN`. Cada una de ellas usa alguna libreria adicional de las cargadas antes.

Actividad 5.1. Realice clasificaciones por los distintos metodos y comparelas visualmente.

Para poder comparar en que zonas los clasificadores presentan mas o menos dispersion podemos calcular la entropia de las distintas clasificaciones en cada pixel. Para esto utilizaremos la funcion `rasterEntropy`. Para esto comenzamos corriendo la clasificacion para distintos modelos, los apilados y despues calculamos la entropia de los mismos

```
1 modelos <- c("rf", "mlc", "svmRadial", "svmLinear", "kNN")
2 ensemble <- lapply(modelos, function(mod){
3   set.seed(5)
4   sc <- superClass(ref.2016, trainData = vector,
5                     responseCol = "MC_ID", model=mod)
6   return(sc$map)
7 })
8 prediction_stack <- stack(ensemble)
9 names(ensemble) <- modelos
10
11 model_entropy <- rasterEntropy(prediction_stack)
12
13 plot(model_entropy)
```

6. Clasificacion no supervisada de imagenes

En esta clase vamos a trabajar con clasificaciones no supervisadas de imagenes satelitales. Vamos a usar los paquetes

```
1 library(raster)
2 library(RStoolbox)
```

Cargaremos primero la imagen landsat 8 y habilitaremos la opcion para escribir el header de ENVI.

```
1 rasterOptions(addheader = "ENVI")
2 set.seed(6)
3 kmeans.2016 <- unsuperClass(ref.2016, nClasses = 5, nStarts = 100,
4                               nSamples = 100)
5 writeRaster(kmeans.2016, "raster_data/processed/kmeans2016",
6             datatype="INT1U")
```

Podemos ahora graficar por separado cada una de las clases

```
1 classes.2016 <- layerize(kmeans.2016)
2 plot(classes.2016)
```

Abriremos la imagen ahora en el qgis e identificaremos cada una de las clases realiendo interpretacion visual de la imagen.

Para realizar la identificacion primero vamos al menu *propiedades de la imagen* → *Estilo* → *Tipo de renderizacion* → *Unibanda pseudocolor*. Elegimos de modo Intervalo Igual y en numero de clases ponemos con el minimo en 1 y el maximo en 100. En estilo de color elegimos colores aleatorios. Iremos luego cambiando los colores uno a uno por un color brillante e identificado a que cobertura pertenece dicha clase espectral.

Construiremos con ella una tabla como la siguiente

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

que guardaremos en un archivo de texto. El mismo lo utilizaremos para realizar la fusion de clases.

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

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

Actividad 6.1. Clasifique por el metodo de kmeans la imagen en reflectancia con una cantidad de clases espetrales lo suficientemente altas para separar todas las clases espetrales.

Actividad 6.2. Vuelva a repetir la clasificacion utilizando la imagen obtenida de la transformada por componentes principales descartando las bandas que aporten menos informacion.

Podemos ahora utilizar la clasificacion para separar zonas de la imagen en el espacio espectral

```
1 ref.2016$kmeans <- reclas.2016  
2 xyplot(nir~red, groups=kmeans, data=ref.2016)
```

Actividad 6.3. Grafique en los cortes del espacio espectral la imagen sin fusionar. Compare la diferencia entre clases espectrales y clases de informacion.

7. Tecnicas pos-clasificacion

Veamos ahora algunas tecnicas de que permiten mejorar las clasificaciones y nos ayudaran a validar y extraer datos de las imagenes clasificadas.

Comenzamos viendo como aplicar un filtro a una imagen. Comenzamos cargando las librerias utilizar.

```
1 library(RStoolbox)
2 library(rgdal)
3 library(raster)
4 library(rasterVis)
```

Para aplicar un filtro a una imagen monobanda, debemos primero definir cual es la ventana en la que trabajaremos y luego cual es la operacion que desamos realizar en dicha ventana.

```
1 window <- matrix(1, nrow=3, ncol=3)
2 classification.3x3<-focal(classification.2016, w=window, fun=modal)
```

En el caso de un filtro por moda, estaremos dejando el mayor que mas veces aparezca entre los que rodean al pixel.

Actividad 7.1. Aplique filtros de 5x5 y 7x7 para filtrar la imagen. Que problemas desaparecen? que dificultad introducen.

Actividad 7.2. Aplique el filtro de 3x3 a la imagen correspondiente al año 2000.

Actividad 7.3. Utilice la funcion raster sieve de qgis para realizar un filtrado espacial. Que diferencias encuentra.

Una vez filtrada la imagen podemos obtener de la misma la matriz de confusion. Para esto debemos cargar el poligono de validacion y la calculamos con la funcion validateMap.

```
1 valid.2016 <- readOGR(dsn="vector_data/", layer="validacion")
2 val.unsup.2016 <- validateMap(sup.2016$map, valData = valid,
3                               responseCol = "MC_ID")
```

Actividad 7.4. Construya la matriz de confusion y obtenga la presicion global para todas las clasificaciones. Que algoritmo funciona mejor con la imagen?

Otra forma de construir incorporar contexto espacial a las clasificaciones es contruir una capa de textura. Veamos como construir una banda de textura utilizando la banda pancromatica de Landsat degradada a 30m.

Ejemplo 7.1. Comenzamos cargando la imagen pancromatica

```
1 pan.2016 <- raster("raster_data/LE72240782000188EDC00/LE72240782000188EDC00
2 _B8.TIF")
```

Una vez cargada podemos visualizarla como

```
1 plot(pan.2016)
```

calculamos ahora el estimador mas sencillo de la textura mediante el calculo del desvio standar en una ventana de 3x3

```
1 windows <- matrix(1,nrow=3,ncol=3)
2 sd.2016 <- focal(pan.2016,window,fun=sd)
```

desvio que pondemos graficar como

```
1 plot(sd.2016)
```

finalmente, degradamos el mapa obtenido a 30m para poder utilizarlo con la imagen multiespectral.

```
1 sd.aggregate.2016 <- aggregate(sd.2016,fact=2,fun=mean)
```

finalmente usamos la funcion **stact** para juntar todas las bandas y proceder a la clasificacion.

```
1 context.2016 <- stack(ref.2016,sd.aggregate.2016)
2 class.2016 <- supClas{}
```