Remote Sensing Lab 5: Topographic correction in R

Victor Gutierrez ([victorhugo@temple.edu](mailto:victorhugo@temple.edu))

## Lab due

October 7, 2021

## Goals

1. To learn how to align and mask images with different geographic projections and spatial resolutions.
2. To master methods for topographic adjustment to spectral satellite images
3. To understand the purpose of and methods for aligning raster files (projecting, cropping and masking).
4. To learn how to mitigate illumination effects in the spectral characteristics of satellite imagery from mountainous areas.

## Total score

The lab counts for up to 4 points towards the final grade of the course.

## Lab instructions

This lab assumes that some of the procedures learned in previous labs for downloading and manipulating images in R are already mastered and therefore are not explained in detail here. Refer to previous labs if there is any problem with those procedures.

1. Go to earthexplorer.usgs.gov. Enter “Hanzhong” as the Feature Name and “China” as the country in the search criteria. In datasets select: • Landsat > Landsat Collection-2 Level-2 > Landsat 8 OLI/TIRS C2 L2 • Digital Elevation > SRTM > SRTM 1 Arc-Second Global.
2. In results, select Landsat as the dataset and download image corresponding to path 128, row 37 from 2020/06/03. Download the product bundle.
3. Select SRTM as the dataset and download the only image available. It should correspond to the coordinates 33, 107. Download the GEOTIFF product.
4. Open R Studio, open a new R script and change the route to your working directory in the script using the function setwd().
5. Copy and paste each chunk of code in your new R script and run it trying to understand the purpose, logic and syntaxis of each line. Make sure the code runs with no errors before moving to the next one.
6. If the code runs, adapt the script to apply it to the “Rocky Mountains”. The Landsat image corresponds to path 38 row 29 from 10-AUG-18. The SRTM image correspond to the coordinates 44, -110.
7. Answer the questions in the answer sheet and submit it along with the required pdf file to canvas.

Have fun!

1. Load required libraries.

library(raster)  
library(rgdal)  
library(RStoolbox)

1. Set working and temporary directories: Change the path below by copying and pasting the route to any selected folder in your workstation. If you are using a windows machine, make sure you use forward slash instead of backward

wd="/Users/tug61163/Documents/Courses/IntroRemoteSensing/2021Fall/Class6/Lab6Materials"  
setwd(wd)  
dir.create('tempfiledir')  
tempdir=paste(getwd(),'tempfiledir', sep="/")  
rasterOptions(tmpdir=tempdir)

1. Decompress, open and stack bands 2 through 7 from the Landsat image

TIFS=list.files(wd, pattern=".TIF")  
TIFS  
# Make sure the numbers corresponding to the names of bands 2 to 7 in the tifs object  
L8=stack(TIFS[c(4:9)])   
  
plotRGB(L8, r=4, g=3, b=2, stretch="lin")

1. Open and project SRTM image. Then plot it.

# This produces a different object because R is case sensitive.  
tifs=list.files(wd, pattern=".tif")  
tifs  
srtmName=tifs[1]  
  
srtm=raster(srtmName)   
srtm  
plot(srtm)

1. Resample the SRTM image so that it has the same projection system. Notice that the two images have different coordinate systems:

# Check that the coordinate system and pixel size of Landsat and SRTM are very different  
crs(L8)  
crs(srtm)  
res(L8)  
crs(srtm)

We can use the projectRaster() function to project srtm to the geographic parameters of the Landsat image. You notice that the parameters are the same. However when you plot it, you see that the srtmPrj image covers the extent of the Landsat image but it only contains valid information in the geographic extent of the original srtm image.

srtmPrj=projectRaster(srtm, L8)  
crs(L8)  
crs(srtmPrj)  
res(L8)  
crs(srtmPrj)  
plot(srtmPrj)

1. Resize images to a common extent: The project raster function has the “alignOnly” argument that ensures that the image has a size equivalent to the extent of the original image. However there seems to be a bug in the current version of the raster package that results in an image with the desired extent but with invalid pixels in it. This derives into an error when trying to plot it:

srtmExt=projectRaster(srtm, L8, alignOnly=TRUE)  
plot(srtmExt)  
  
"Error in .plotraster2(x, col = col, maxpixels = maxpixels, add = add, :   
 no values associated with this RasterLayer"

However, srtmExt has the right extent (at least much smaller than srtmPrj)

extent(srtmExt)  
extent(srtmPrj)

Let’s use the function crop to reduce the area of the srtmPrj object to the extent of the original image. In previous labs we used the drawExtent() function.

Remember that previously we obtained the extent by using the drawExtent() function. You can see below that the crop() function can read the extent of another image and crop the input object to that size.

We will also crop the Landsat image so that we only use the overlapping area between the two images

srtmPrj=crop(srtmPrj, srtmExt)  
L8rsz=crop(L8, srtmExt)  
plot(srtmPrj)  
plotRGB(L8rsz, r=4, g=3, b=2, stretch="lin")

1. Mask images to a common valid domain

m=c(-Inf,0,NA, 0,Inf,1)  
reclassmatrix=matrix(m, ncol=3, byrow=TRUE)  
L8msk=reclassify(L8rsz, reclassmatrix)  
L8msk=Reduce(f="\*", as.list(L8msk))  
srtmsk=reclassify(srtmPrj, reclassmatrix)  
  
mask=L8msk\*srtmsk  
plot(mask)  
  
L8mskd=mask(L8rsz,mask)  
srtmskd=mask(srtmPrj, mask)  
  
plot(L8mskd[[5]])  
plot(srtmskd)  
  
# Remove elements to release memory  
#rm(mask, L8msk, L8msk,srtmsk, L8Rsz, srtmRsz)

1. Calculate terrain variables from the DEM.

slope=terrain(srtmskd, opt='slope')  
aspect=terrain(srtmskd, opt='aspect')  
terrainvar=stack(slope,aspect)  
names(terrainvar)=c('slope', 'aspect')  
rm(slope,aspect)

1. Calculate illumination. This code is to simulate the shading of the sun according to the illumination geometry. For that, you will need to retrieve the sum azimuth and elevation data from the metafile that comes with the Landsat image

illuGeom=c(114.68025538,(90-68.42711855))  
shade=topCor(L8mskd, dem=terrainvar, solarAngles=illuGeom, method="illu")  
plot(shade)

1. Apply topographic correction

L8cor=topCor(L8mskd, dem=terrainvar, solarAngles=illuGeom, method="C")  
  
pdf("VGutierrez\_Lab6.pdf")  
 plotRGB(L8mskd, r=5, g=4, b=3, axes=FALSE, stretch="lin")  
 plotRGB(L8cor, r=5, g=4, b=3, axes=FALSE, stretch="lin")  
dev.off()

## Lab 6 deliverables

1. Upload two RGB map composites representing the pixel values for bands 543 from Landsat before and after topographic correction for the Rocky mountains. Change the name of the pdf file for your Last name First initial and the name of the lab assignment (“GutierrezV\_Lab1.pdf”“) before submission (2 pts).
2. Read the help documentation for the function topCor() and then answer the questions below briefly (0.5 pts each):

In the application of the topCor function to produce an illumination model: a. What are the two variables that it calculates from the DEM raster?

1. What are the two variables that it obtains from the Landsat metadata file? What is the value of those two variables in the Landsat scene used for the Rocky mountains?
2. What does the application of the “illu” method in the function topCor() produce?
3. What does the application of the method “C” in topCor() produce (check the images in the BeforeAndAfter.pdf file obtained from running the algorithm)