1. Introduction

In today's lab, you will be familiarizing yourself with the programming language "R" and the popular user interface "RStudio", as they relate to the basic processing of satellite imagery. The increasing quality and availability of remotely sensed images is accompanied by a rise in demand for the computer-based skillsets that enable researchers to perform the necessary image processing. The R language and software provide one means of doing so, as well as supplying a host of statistical tools for further analysis. To begin, we will cover some basics about how the R language works and what the RStudio interface provides.

2. What is R? What is R Studio?

R is an open-source (free) computer language that can be used as a powerful tool for data analysis, statistical computing, and the production of graphics. R allows users to import and analyze large quantities of spatial (and non-spatial) data in many formats, and to perform a large variety of spatial processing tasks on both vector and raster data. It contains many libraries with highly specialized tools for statistical modelling of spatial data. Finally, it has capabilities for visualizing intermediate results and producing quality maps and output graphics. Rstudio is an interface through which we can communicate with R to both provide instructions and receive messages. If we consider R to be a car's engine, RStudio is the dashboard. This software is also free and open-source, and is available to download for all major operating systems. RStudio provides users with dedicated space to write, edit, test and debug code, as well as tools to display graphics, consult working history, and access help documentation. We will be working with both of these components, hand-in-hand.

3. The RStudio Interface

Before diving in to working with the R language, it will be helpful to orient ourselves to the RStudio interface, which is divided into four main windows. The window on the top left is the "Source", where you will write and store your working code. You can see here that there is a script open and lines of code showing. The window on the bottom left is the "Console", a place you can type commands and see non-graphical results. You can see that some commands have been written in here (blue text) and results returned (black text). The window on the top right is the "Environment", a place that will keep a handy list of the objects you have created. You can see that there are currently five objects in R's working memory. The window on the bottom right will display the "Plots" (graphic outputs) you create, and also return "Help" documentation if you call for it. You can see that right now it is displaying a true color composite that has been created in the script.

4. Getting Started with the R Language

Some of the most basic fundamentals of R include assignment, variables, and functions. An "assignment" is an operation in which you assign some value to a variable. You can think of a variable like a container, and assigning a value to the variable like putting something in the container. Then, we can simply call on the container to get its contents. We assign a value to a variable by using the

assignment operator you see here, which looks like a left facing arrow. Whatever value is on the right of the arrow, will be assigned to the variable on the left of the arrow. In the green example, we have assigned the value “Hello world” to a variable called greeting. You decide what the variable will be called, though it is case-sensitive, must start with a letter and must only include alphanumeric characters, a dot, or an underscore (no spaces!). Depending on the value you assign, the variable will inherit a type. The most common types are character (a word, letter, description) and numeric (a number), and more complex combinations of these values in the form of vectors, lists or matrixes. Functions are like small tools that perform a variety of different tasks and return specific results. Here we see the print function, where “print” is the name of the function and the parentheses that follow it contain the arguments the function requires (in this instance, the object to be printed). By running this function, we would get the contents of the greeting variable (“Hello world”) printed out in the Console window. In the lab exercise you will gain experience working with a number of different functions, both simple and complex.

5. Basics of Working with Images in R

R comes with a basic set of libraries, containing many functions that apply to simple statistics or operations. However, to work with images you must install and load image-specific libraries onto your computer and then into the R session. In the lab exercise, you will make use of the raster library, which is capable of interpreting, manipulating, and analyzing any sort of raster data, including the satellite images we are interested in processing. Once this library is installed, each desired band of data from a satellite image can be assigned to its own variable, using the assignment operator we just learned about. Then, the bands are combined into what is called a “raster stack”, which can be manipulated in several ways. It can be plotted in several forms of color composites, one of which is seen here on the right. You can also perform band math such as the NDVI calculation, you can retrieve or visualize information about single bands or the whole stack, and you can supply this raster stack object to a more complex function in order to perform processing tasks. You will get a chance to perform all of these operations yourself during the lab exercise.

6. Saving Your Work in R

Finally, it is crucial to save all of your hard work! Not only will you want to save the final results, but you may also want to save the code that you wrote to do the processing itself, the plots you generated, or the workspace you had set up. To save any output rasters you may have created, such as a classified image, you can use the writeRaster function supplied with the appropriate input data and arguments. This will generate a TIF file in the specified directory, which can be viewed in another raster-supported software such as ArcMap or TerrSet. To save a plot or a graphical output such as the one we saw on the last slide, you can simply click the Export button at the top of the Plot window when the desired plot is displayed, choosing whether you would like to save your plot as a PDF file or an image file, and specifying the necessary details such as directory, filename, and image size. To save an entire script, you can click the Save button at the top of the Source window and specify a directory and filename. Finally, you will also be given the option to save a workspace when you choose to save your script, which keeps a record of all the objects you have created and will have a .RData file extension.

9. Lab Scenario

For today's lab, you will be exploring all of the details we just discussed by getting your hands dirty and writing some code yourself! First, you will download and install both R and RStudio (if they are not already installed on your machine). You will then orient yourself to the interface and try out some basic coding before diving into two actual image processing tasks. These tasks will include performing an NDVI calculation and an unsupervised classification upon satellite imagery from western New Mexico. After completing this lab, you should be able to achieve the objectives listed here, and will have gained a degree of familiarity with a new coding language! The amount of time this lab will take you will depend on whether or not you already have familiarity with coding, and the pace you set for yourself. To get the most out of this exercise I recommended taking your time (regardless of the speed your neighbor may be going!) and get a full grasp of each step before moving on to the next. Also, typing the lines of code out yourself rather than copying and pasting is a great way to cement your knowledge and will ensure that you don't run into any formatting errors.