Directories and File Organization

Marguerite A. Butler

January 21, 2021

Anytime that you want to read in a file from your hard drive or save output to your hard drive, you need to know where your working directory is. By default, when you start R it is wherever you started the R program. If you start R by double clicking on a script, it will generally set the working directory there. If you start it from your Applications or Program menu, It will be your home user directory by default. You don't want all your work going there. So let's take a moment to set up some nice directories.

0.1 Directories and File organization

In order for R to interact with the files on your computer (i.e., for INPUT/OUTPUT), R needs to know the path to your working directory. This is where R is ”parked” on your computer, and will look here for external files, or will write output files to here.

Mac the default working directory (on your computer) is your User directory. For example: ”`/Users/marguerite`” or ”`/Users/marguerite/Documents`”. Or where you opened your .R file (more on this later).

PC default is ”C:/Program Files/R/R-4.0.3” (your installed R version number).

Linux/UNIX or running R in a terminal default is where you started R.

0.1.1 Course Directory Organization

So let's create a working directory. For the purposes of this course, at the top level of your user directory or in Documents, please create a folder called `Rclass`. You will have to do this outside of R. Either create the `Rclass` folder through the Finder or open a terminal, change to your Documents directory if you're not there via ”`cd  /Documents`” then ”`mkdir Rclass`.” As you accumulate files, you may want to make additional folders inside ”`Rclass`” for example called ”`Homeworks`” or what have you. You should get in the habit of making a folder for each R project and in general keeping your work organized. If you have a lot of junk files, at least consolidate them into one folder. Personally with my own data, I keep a ”`RawData`” folder that contains raw data that never gets written over. I archive a copy of pristine raw data in case I ever need to go back to it. If I want to edit it, I edit a copy in a working data folder.

`Rclass` the main project file for the course. It will contain all source code (scripts) and direct output. If this folder gets too big, we can make subfolders.

`Data` to store our raw data input files (spreadsheets and text files). It should be within `Rclass`.

`Rdata` if you have a lot of binary Rdata files for example if you are doing simulations, or you have processed data for your dissertation, etc., you may want to set up a separate folder for writing processed R data files. After setting this up, future analyses or scripts can access these files directly, rather than working from the raw data files.

When you are done with the course, you can move it to an appropriate place in your file heirarchy. As an example, this is the way I organize my personal computer (Fig. 1): Organization will vary by need, but for large project with multiple parts, I like to have all of my Data in a subfolder called ”Data”. Within my working directory, I like to save all of my R code and analyses. I might have separate folders for each subproject. This working directory is called ”Rclass”.

0.1.2 Moving through the directories

You can ”get” your current working directory from within R. You can also set your working directory (after creating the directories first). This is the filepath from the root directory of my computer:

```
> getwd()
> setwd("/Users/marguerite/Documents/Rclass")
```

And this is the filepath from my ”home” directory which is ”marguerite”:

```
> setwd("~/Documents/Rclass")
```

As you may have guessed, the ”filepath” is the path to your files. In the Unix file system, the ”root” is signified by starting the filepath with ”/”. You can't go up any more folders from the ”root”. Anything to the right are names of the folders within the root, and within that folder, etc. The first example above is called an *absolute filepath*. You can also use *relative filepaths*, which navigate relative to where you are. In this case, start with a name rather than ”∵ Some useful special characters are:

`~` a special character for ”my user directory”

`..` which means to go up one level

`.` which means the current directory (here)

`/` separator between folders or levels. If you begin your filepath with `/` with nothing preceeding it, this indicates an absolute file path starting from the root.

For example, if you wanted to back up to your user directory and change to a project called ”`MyFirstAnalysis`”, you would have to go up one directory and then specify the folder name, so filepath would be ”`../MyFirstAnalysis`”. To go up two directories and then into a new directory, use ” `../../MyFirstAnalysis`.”

Now try rerunning the code (you can get the lines you typed or cut and pasted by hitting the up arrow, or by clicking on the history icon (the blue and yellow striped box), and voila! You will see the pdf appear in your Rclass folder.

```
> pdf(file="myplot.pdf")      # turn on pdf device for output
> x <- rnorm(10)
> y <- rnorm(10)
> plot(x,y)             # plot 10 random points
> dev.off()                # turn off pdf device to return output to the default
> save(x,y, file="xy.rda")  # save  in Rdata format
```

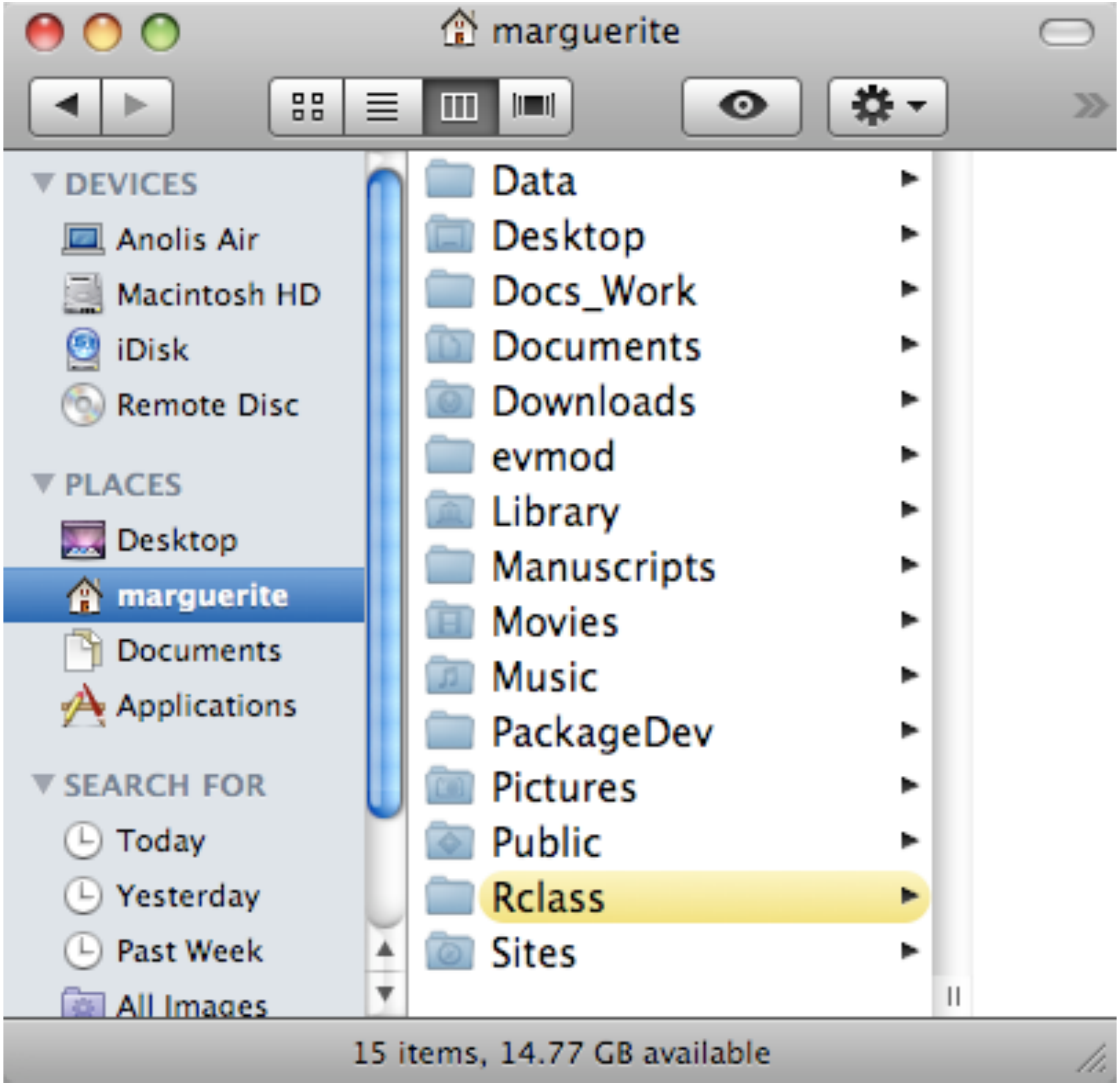

Figure 1: An example of directory organization to keep your R programming projects organized. When you are actively working on a project, it may be more convenient to have it at the top level of your user directory (here ”marguerite”). When you‘re done you can move it to a permanent location.