# Barcode Analysis Application User Guide

This guide was created to provide an understanding of how to use the BAF.DistributedScript.R. The script requires a setup file to run and for the sake of avoiding to make any changes within the analysis script, the setup file is stored as a separate document. It is recommended to run the script in the IDE R Studio, as it provides a variable explorer that can be useful for quickly accessing the internal data structures within the barcode analysis function call. You are free to use whatever IDE you choose, but this user guide will be written with RStudio in mind.

Within the accompanying setup file, you will need to provide a path to the reference list (if you are using a reference), the path to the primer list and the path to the folder containing your run info form and the FastQ sequences. The app is fairly particular about the structure of these documents, so please be sure to the follow the examples provided. First, let's start with a brief introduction to RStudio.

R Studio Basics:

Let's cover some general usage guidelines for running the script in RStudio.
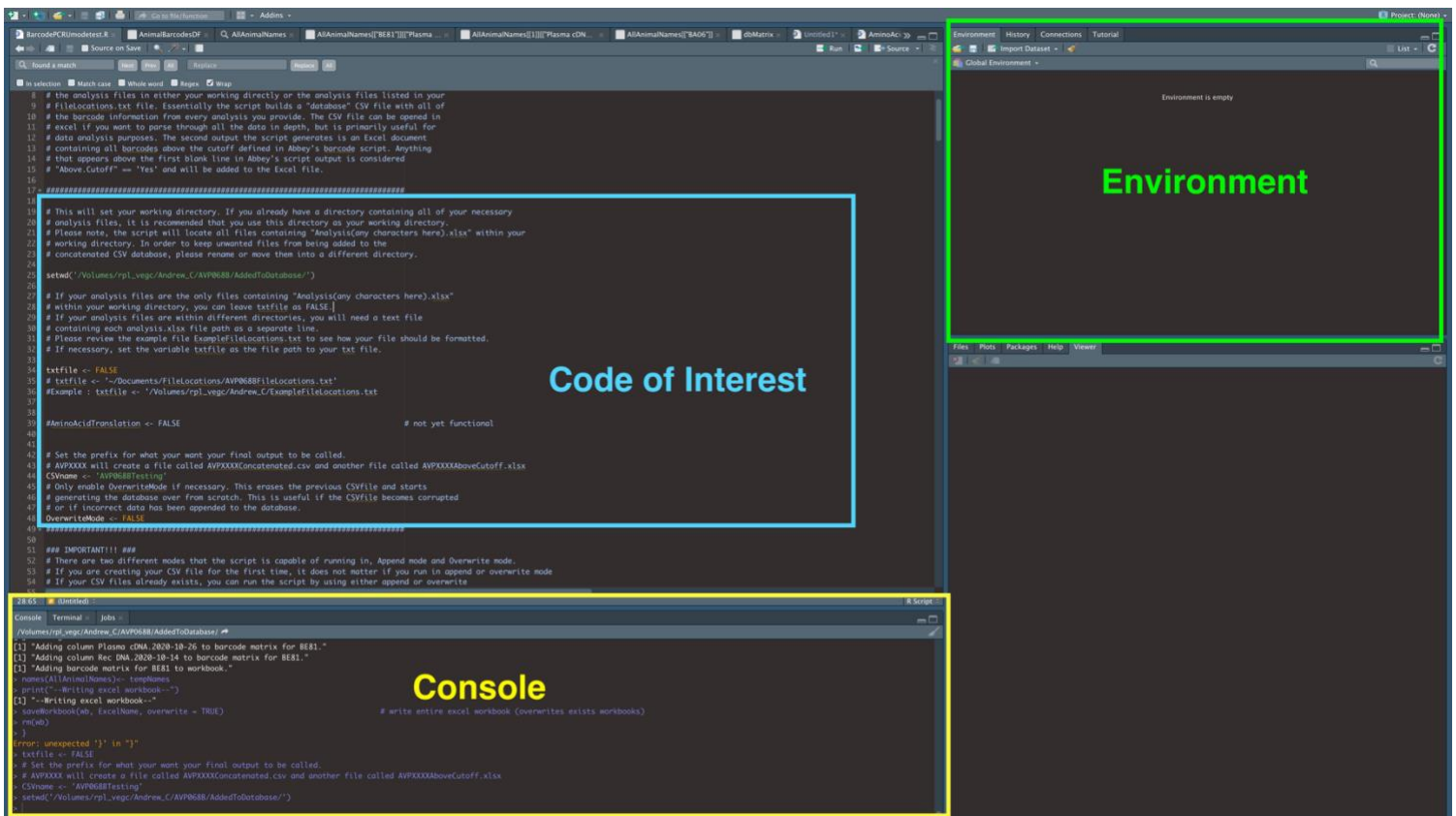
Within RStudio, you have your script/coding area, your console (which will update you about the status of the code/script that is currently running) and your environment (where variables are stored). For the sake of running this script you don't need to worry about the variables that pop up within the environment. However, you will want to ensure that the environment is clear before initializing the script every time. This can be done by clicking on the broom at the top of the environment window.

Setting up the Analysis Script:

Primer List:

Now that we're discussed some of the RStudio basics, let's talk about the input files that will be required for the analysis script to run, starting with the primer list. The primer list file is fairly straightforward. It simply includes all the possible the primer names and the primer sequences that could be present within your sample. In this case, we have provided a VPX primer list which includes 106 different primers. Please note that you should leave the first row of the primer list blank for the data to be read in properly.

For the sake of this tutorial, I will gloss over the composition of the primer indexes. Essentially, we want to isolate the 8bp primer index (shown below in purple) that distinguishes it from the other primers within the list. The isolation of the primer index is performed on lines 81-87 within the BAF script. **Please note, if you are using the VPX barcodes and VPX primer list, you do not need to change anything in lines 81-87 of the BAF.DistributedScript.R and you can continue on to the Fasta file section.**

```
81    primerseqs <- suppressMessages(read_excel(primerseqs, col_names = FALSE))
82    primerseqs <- as.data.frame(primerseqs)
83    primerseqs[,2] <- gsub("AATGATACGGCGACCACCGAGATCTACACTCTTTCCCTACACGACGCTCTTCCGATCTNNNN","",
84                      unlist(primerseqs[,2]))
85    primerseqs[,2] <- gsub("CCAGAACCTCCACTACCCATTCATC","", unlist(primerseqs[,2]))
86    primerseqs[,2] <- toupper(unlist(primerseqs[,2]))
87    primerseqs <- primerseqs[order(as.numeric(gsub("VPX.P5.", "", unlist(primerseqs[,1])))),]
```

On line 126, there is a gsub function call that will remove the region of the sequence that is upstream of the unique primer index (shown in red). The entire sequence is replaced with "", meaning that those characters are removed. This step occurs again on line 128 of the script, but the linker portions of the primer are removed (shown in blue). Only line 139, another gsub call occurs which removes the portion of the primer name. In our example, we temporarily remove a portion of the primer name so that we can order the primer list by the primer number. In this example, we temporarily substitute out the "VPX.P5." portion of the primer name so we can order them as 1, 2, 3, 4, etc. while reading them in. These 3 lines of code will need to be changed if you are using a different set of primers other than what we use with SIVmac239M, SIVmac239M2 or SIVmac239NIRM.

Fasta File:

Continuing on, we can look at the reference file provided, which should be in a basic FASTA format. Shown below is a sample of the FASTA formatted reference, with a description line and a sequence line. The description line contains the barcode name, while the sequence line contains the barcode sequence. The SIVmac239M FASTA file was generated from a barcode stock characterization as previously described in the following paper. (https://pubmed.ncbi.nlm.nih.gov/31597757/)

```
>SIVmac239M.10203
AAAAGCCTCCCCCATTTGAGCCAACTGTGGGCGT
>SIVmac239M.9766
AAACAGAGTGTTTCCAAACCGCTGAATGAAGCGT
```

Run Info Form
The final file that is passed to the setup R script is the run info form. The run info form contains the following identifying information for each sample:

- the animal (or a unique name for each cell line if sequencing on cell culture)
- the sample type (tissue/plasma, cDNA or DNA, etc)
- the sample collection date
- the primer ("Barcodes") used for each sample
- the total amount of input per barcode
- the total amount of input per well

The run info form also contains a bit of information about the sequencing run number, the run name, the run date and the type of sequencing being performed (paired-end, vs single-read)

There are two extremely important things to consider when setting up the run info form. If you want to avoid any errors.

1) **The FastQ file should have the same base name as the Run Name provided, with an underscore following the name. (i.e. if your RunName is ExampleSetup, then your FastQ file should be something named like ExampleSetup_Test.fastq or ExampleSetup_.fastq.gz. Anything after the underscore is insignificant as it is not considered by the script when matching the FastQ files to the run info form.**
2) **It is imperative that you ensure that the primers are properly ordered and sorted by numerical order.** If you accidentally switch the order of the barcodes within the runinfo sheet, you may encounter an error within R.

Run Number	Animal	Sample	Date	Barcodes	Input TOTAL PER BARCODE	Input TOTAL PER WELL		(F Barcode)	(cDNA)
MS3111601-300V2	Animal1	LN DNA	10/14/20	VPX.P5.1	10,000.00	10,000.00		VPX.P71F	SL8R
	Animal1	LN cDNA	10/14/20	VPX.P5.2	52.80	26.40		VPX.P71F	SL8R
	Animal1	Rec DNA	10/14/20	VPX.P5.3	504,477.65	126,119.41		VPX.P71F	SL8R
	Animal2	Rec cDNA	10/14/20	VPX.P5.4	392.24	196.12		VPX.P71F	SL8R
	Animal2	Plasma cDNA	10/14/20	VPX.P5.5	500,000.00	500,000.00		VPX.P71F	SL8R
	Animal2	Plasma cDNA	10/22/20	VPX.P5.6	500,000.00	500,000.00		VPX.P71F	SL8R
	Animal2	Plasma cDNA	10/26/20	VPX.P5.7	96,964.26	48,482.13		VPX.P71F	SL8R

Run Name
ExampleSetup

Date
10/27/20

Regular Taq
300-V2 Reg Kit
VPX

One-way

As long as your run info form follows these two guidelines, you should at least get the script to run. Let's look at the information we need to provide to the setup file:

```r
#### Inputs that we need to pass to the function (change frequently) ####
# Path to folder containing fastq file and runinfo excel file
folder <- "/Path_To_Folder_with_FastQ_And_Run_Info"

# Path to the reference FASTA containing known barcode list
# There is a different reference file for each barcoded virus
# reffilepath <- NA # For uncharacterized/new stocks
reffilepath <- "/Path_To_239M reference.fasta"

#### Inputs that rarely change unless doing a complicated analysis ####

# Adapter sequence (ours always has NNNN)
adapter = "NNNN"

# Specify whether reads are forward or reverse complemented
fwdrev <- "Reversed" # For 239M and 239M2
#fwdrev <- "Forward" # For X-virus/INT

# Reference sequence reads will be aligned to
ref = "ATGGAAGAAAGACCTCCAGAAAATGAAG" # For 239M and 239M2
offset <- 0
# Maximum number of mismatches to reference that are allowed when aligning
mismatches <- 2

# Specify whether to extract region upstream or downstream of reference
direction <- "Upstream" # For 239M and 239M2

# How many bases to cut
numbases <- 34 # For 239M and 239M2

# Whether to use 1/input cutoff
cutoff <- "Yes"

# Whether to include uniques in full analysis or not
full <- FALSE # For normal runs

# Hamming distance to flag
mindist <- 1

# How many reads to analyze at a time (more is faster but uses more memory)
parse <- 5*10^5

# Path to excel file containing Miseq primer sequences
primerseqs <- "/Volumes/rpl_vegc/Abbey/BaseSpace/106 primer list.xlsx"
```

The first item that needs to be provided is the "folder" or the path of the directory containing the run info and the FastQ file(s). Please note that when searching for run info forms, the script will find all ".xlsx" files within the provided directory that do not contain the word "Analysis". Please be sure to not store your reference files or primer list within the directory you provide to the folder variable. Additionally, the script will search through all the nested directories contained within the path you have provided for fastq files and .xlsx documents. The two examples provided below are both valid setups for running the script.
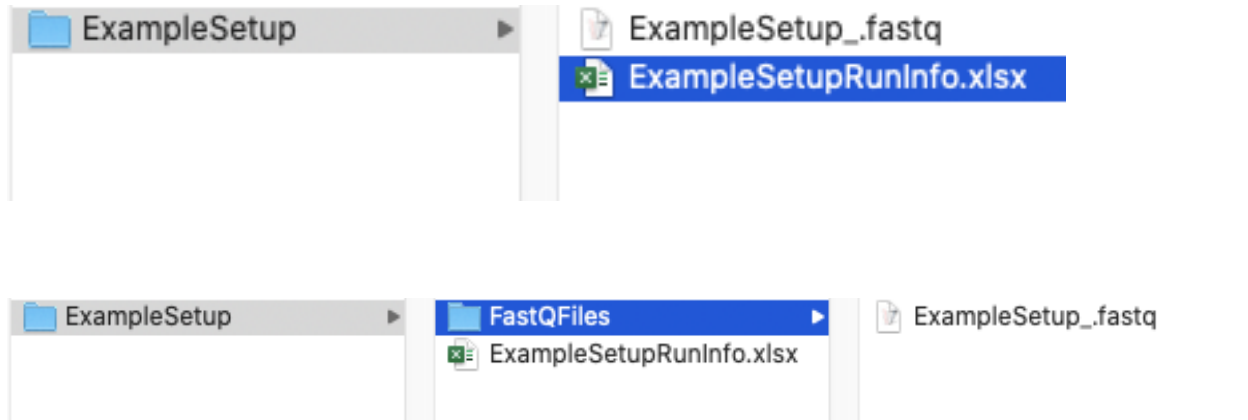
On a Mac, there is any easy way to determine the file path. Simply right click the document of interest and then select "Get Info". This will have a pop-up window with the information regarding the file. Highlight the text to the right of there "Where:" identifier as shown below. This information contains the file path up to the directory that the file is held in. The example shown below shows the file path for the example run info form. You can copy the information that is highlighted in blue and paste it directly into the quotation marks provided for the folder variable. In this example, we used the "where:" information of the Run Info file. Since its parent directory also contains both the fastq file and the runinfo file.
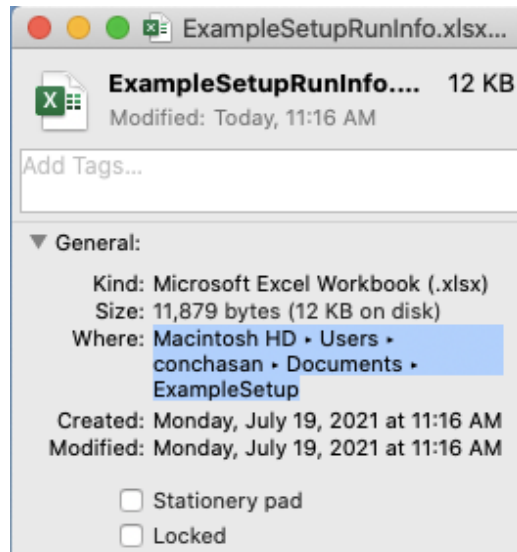
Repeat this process to provide the file path for reference and the primer list. However, when providing the reference file and the primer list file paths, we need to provide the specific path to the file itself, not just to its parent directory. There are two methods of doing this. The first follows the same steps as before, but this time be sure to also add the document name and extension (highlighted below) to what you include inside of the quotation marks.

You can paste the path to the parent directory, manually type "/" and then paste the name/extension of the file.
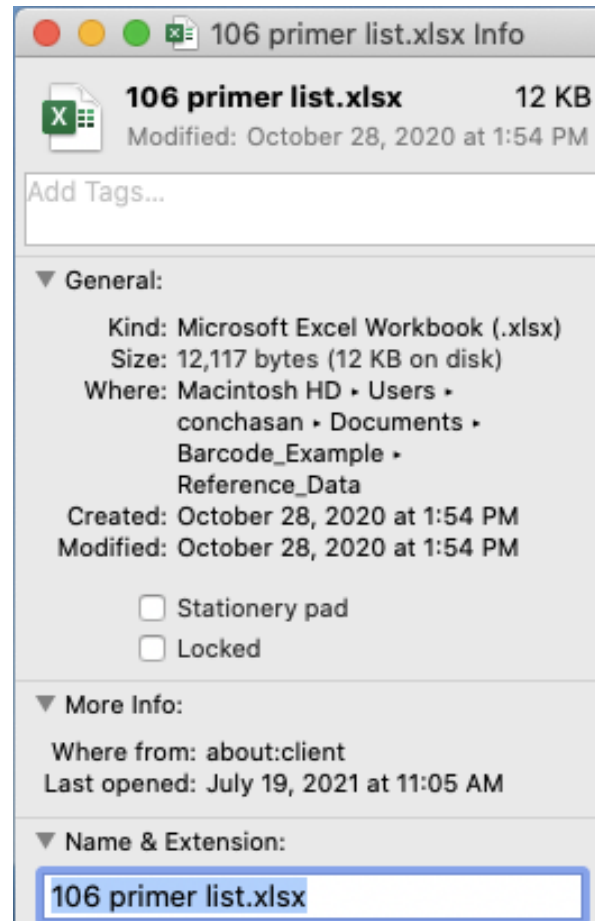
The alternative is to simply paste the path to the parent directory, press tab and scroll through the list of files with your arrow keys until your document is highlighted and then press tab again to have it autocomplete the text. In the example provided below, the parent directory path was pasted into RStudio and then tab completion brought up the files contained in the directory. For the reffilepath, we would use the arrow keys to select "239M reference 071619.fasta" and then press tab again.

If you are using the VPX barcodes, the default settings should be sufficient for the remainder of the variables in the setup file. These runs are performed with the following critera:

- reversed mode (reference sequence is reverse complemented and found in the 3' to 5' direction)
- a reference of "ATGGAAGAAAGACCTCCAGAAAATGAAG"
- an offset of 0
- capturing the 34bp region "upstream" of the reference sequence (easier to conceptualize if you think of the 5' direction as upstream, 3' direction as downstream)
- 2 number of mismatches of reference sequence allowed
- with a 1/input cutoff applied
- hamming distance match of 1
- full uniques set to FALSE
- parsing 500,000 fastq sequences at a time

If you need help running a non-standard (not VPX) analysis, feel free to reach out to Andrew Conchas (andrew.conchas@nih.gov) for additional help in setting up these parameters.
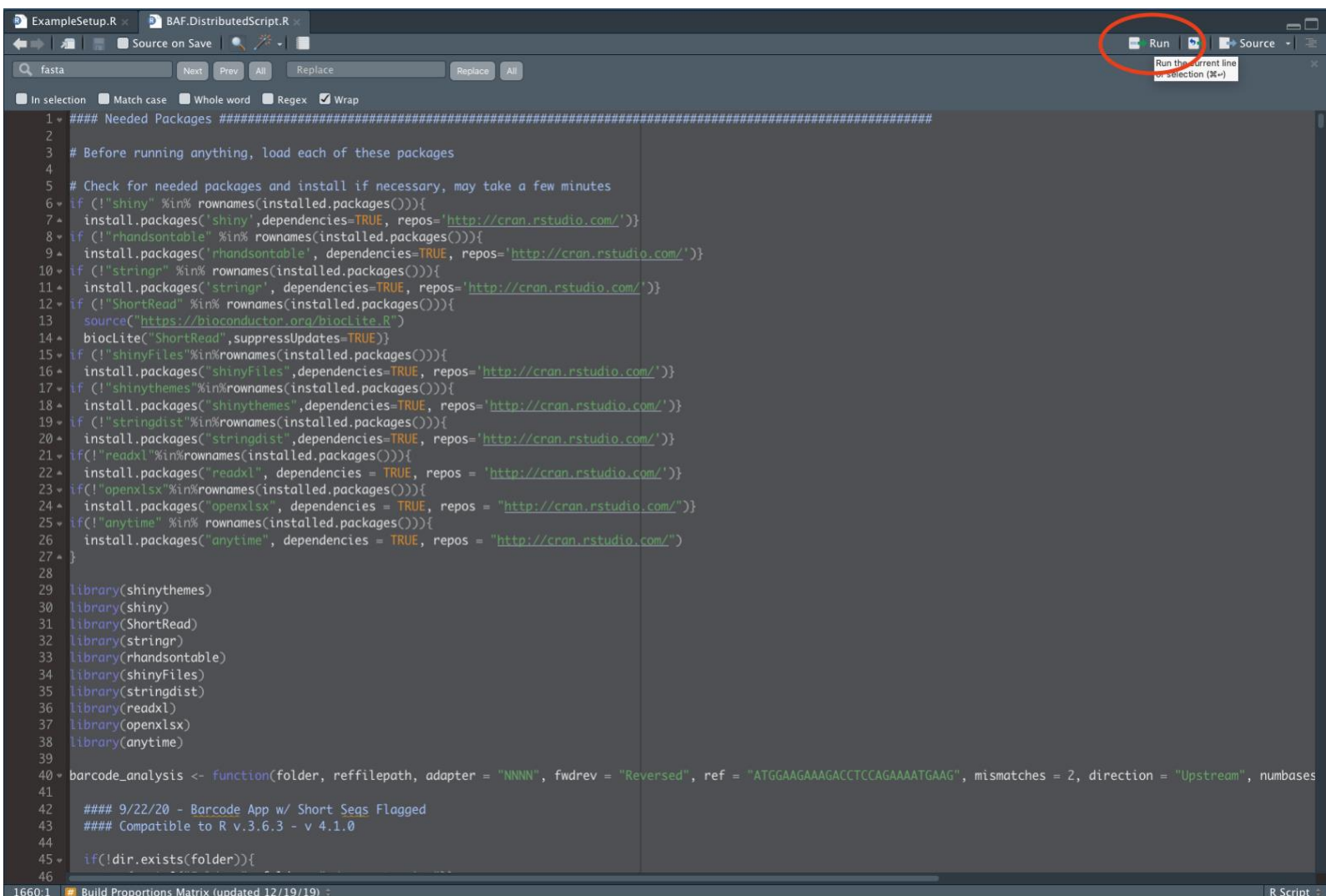
At this point, you are ready to begin running the barcode app.

Running the Script:

There are two methods for running the script, the basic method and the advanced method (which is useful for troubleshooting).

We'll start with the basic method. To start the script, navigate to the BAF.DistributedScript.R file and ensure that lines 84-90 are correctly implemented to handle the primer list you provided. Once you have verified that those lines are correct, press Command + A on Mac or CTRL + A on Windows within the coding area (upper left window). This will highlight all of the code, which you can then run by clicking the run button in the top right, or by pressing Command/CTRL + Enter.

This will load all of the code from the BAF.DistributedScript.R into a function. If this is your first time running the script, then it will also install the necessary R packages. Your environment window will look like the following:
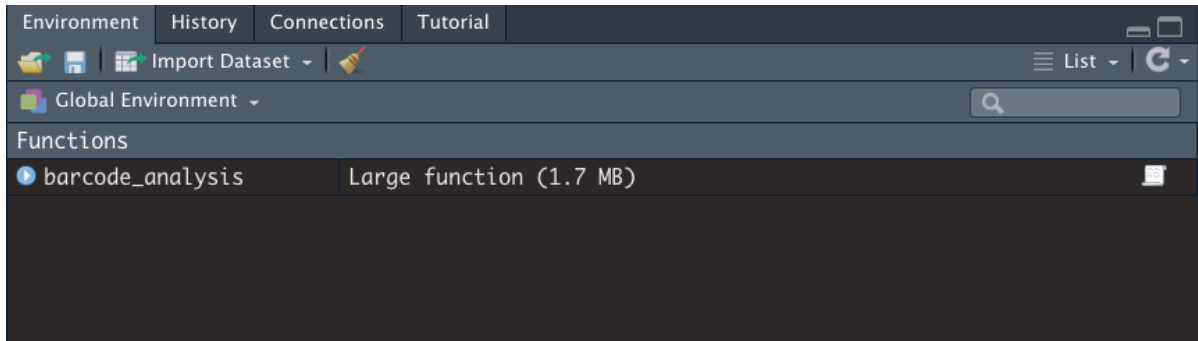
Next, open the setup file within RStudio. Your environment window (upper right hand window) should still have the barcode analysis function loaded within it. Now please ensure that all of the information provided to the setup file is correct, and then run the script by pressing Command/CTRL + A and then Command/CTRL + Enter (or by clicking the green run button once all of the code is highlighted). This will call the barcode_analysis function using the parameters in the setup file. If everything is set up properly, the script should run and provided periodical updates in the lower left window (the console).

If you are more experienced with R, then you may want to run the script in a manner that will allow you to view the internal data structures. To do this, load the setup file first instead of the analysis file. When running the setup file, you should encounter an error saying the barcode_analysis function is not found. You should see all of the variables in the setup R file populate your environment variable within the top right hand corner. Once the variables are loaded, proceed to the BAF.DistributedScript.R script and load the necessary packages by highlighting lines 1-38 and running them. Afterwards, highlight the lines from line 42-1661 (excluding the final line that has an extra "}" bracket relating to the barcode_analysis function) and run them. This will essentially run the barcode_analysis code, but will maintain all of the temporary data variables within your variable explorer in the top right. This could be useful if you are attempting to diagnose an error within the code that prevented it from creating the analysis file. For helps on troubleshooting, feel free to reach out to Andrew Conchas (andrew.conchas@nih.gov).

The analysis file generated by the script presents the outputs in Excel format and provides different tabs for each animal included in the run info sheet. An example analysis form has been included within ExampleSetup directory for you to reference.