How-To Guide for Instructors

This module is intended to demonstrate how R can be used to understand taxonomy and nomenclature, retrieve taxonomic information from online databases, parse species names, and retrieve taxonomic identifiers linked to the Open Tree of Life. This is the first in a series of modules that demonstrate how to use phylogenies to understand trait evolution or can be a stand-alone module on Taxonomy.

To run this module, students and the instructor should have downloaded *RStudio* or have access to *R Studio Cloud*.

The module is available for instructors as a RStudio Project (*Instructor_Phylo_Module_1*). For more information on projects, refer to the RStudio support site. All scripts, input, output and instruction files are contained within this project. A separate Project (*Student_Phylo_Module_1*) is available for students (excluding the answer keys and other instructor-specific materials).

Within the main Project folder, there are four folders that will be used in the module: *Data*, which contains the input data, *Output*, which contains saved output from various stages, *Scripts*, which contains the files of code for running the module, and *Instructions*, which contains the instruction files for both students and the instructor. The *Admin* file contains files that are used in the background, files documenting references and source material, and the answer keys to assessments. The *packrat* folder can be ignored - it contains the packages used in this module.

Instruction files are contained within the *Instructions* folder. Both PDF and .RMD files are available for each section. For those confident in R, feel free to use and edit the .RMD files. For those new to R, the PDF files will be more accessible. In this module, there are two sections: Introduction to R, which provides a brief overview of some of the commands and actions that will be used in the module, and Taxonomy, which contains the core of the lesson. Students have access to both the **Introduction to R** and **Taxonomy** instruction handouts. These handouts walk the students through the exercise.

Within the *Scripts* folder, the script files are labeled by section title. For example, for the Taxonomy section, the script is labeled **Taxonomy.R**. To open scripts, simply click on the title and it will open in the script quadrant of RStudio. Explanations for how to run lines of code are included in the **Introduction to R** handout.

Input and output files are labeled with descriptive titles that should indicate what the files contain. Intermediate output files (for cases where internet is unavailable or in case students run into challenges with an earlier step) are saved in the *Intermediate_output* folder within the *Output* folder in both the Student and the Instructor versions of the module. Students should avoid using those files unless necessary but they are available for students to access so assessments should not be limited to reproducing those files. These objects have been saved as .rds or R objects. To read an intermediate output R object, use the following line of code (edited to the appropriate file name):

```
#This is just an example: replace object name and file path with the appropriate names
row1 <- readRDS("../Output/Intermediate_output/row1.rds")
```

To start, open the appropriate instruction file and script, and proceed through the module. For example, for an introduction to some commands and how to run R, open the **Introduction_to_R.pdf** handout in the *Instructions* folder and the **Introduction_to_R.R** script in the *Scripts* folder. Otherwise, if you are ready to get started with the core of the module, open the **Taxonomy.pdf** handout in the *Instructions* folder and the **Taxonomy.R** script in the *Scripts* folder.

Common Errors

Warnings can be ignored in this module

- Error in tnrs_match_names(birds) : could not find function "tnrs_match_names"

Make sure you load the necessary packages before running the script (lines at the top of the script). If the packages are not loaded, the functions can often not be found.

- ants_df <- read.delim("../Data/Ant_names.txt", header = FALSE, stringsAsFactors = FALSE /+ /+

If the line isn't "complete" the console won't be able to finish the line and will wait for more input with the symbol +. Check to make sure the ending of the line wasn't omitted or deleted (here the final) was deleted). To get rid of the +, enter several **)))**. You will get an error and can rerun the **complete** line.

- Error in str.default() : argument "object" is missing, with no default

The function was called without the appropriate arguments within the parentheses (eg *str()* rather than *str(object)*). Make sure that the correct arguments (variables that the function will work on) are provided for the function.

- Error: unexpected string constant in "conifers <- c("Abies" "Thuja""
- Error: unexpected string constant in "conifers <- c("Abies", a"Thuja""

Unexpected symbols or strings often refer to missing commas or other puncutation. In the first example, a comma is missing between two strings. To correct this, insert a comma between "Abies" and "Thuja" to get c("Abies"**,** "Thuja"…..). If this happens when running the original script, revert to the original version of the document (there shouldn't be issues with this in the original version). Otherwise, check for missing punctuation, extra letters or symbols accidentally typed into the code (as seen in the second example).

- Error: object 'ant' not found

There can be a few explanations for an object not being found including 1) the line of code that would generate the object hasn't been run yet or 2) the object name is misspelled but exists with different spelling. Here, the correct object name is "ants" rather than "ant."

- Error in .tnrs_match_names(names = names, context_name = context_name, : You must supply a 'names' argument

This is another example of where the correct argument (such as an object or word) must be supplied. If supplying arguments in the same order as indicated by the function, the specifier can be omitted; this means that tnrs_match_names(names = names_object) is equivalent to tnrs_match_names(names_object). To ensure that arguments aren't missing, it can be helpful to include the specifier to verify that each field is provided.

- Error in which_rank(downto):NROW(taxize_ds$rank_ref) : argument of length 0

In this case, the rank in the function was misspelled as **speices** (sapindaceae_species <- downstream("Sapindaceae", downto = "speices", db = "itis")) so the function was not able to match the rank. In these cases, *Argument of length 0* often indicates that input was either misspelled or misspecified.

- cannot open file '../Outputs/ant_species_names.txt': No such file or directoryError in file(file, ifelse(append, "a", "w")) : cannot open the connection

If there is "no such file or directory," make sure that the folder exists and that you have specified the correct path. Typos can cause problems here, as can misspecifying how far back in the directory (../../) you are pointing to. To fix this, check the directory listings in the Files corner of R Studio.

- conifer_subspecies $conifer_species_list [1] parentname parenttsn rankname taxonname tsn <0 rows> (or 0-length row.names) attr(,"class") [1] "children" attr(,"db") [1] "itis"

If a query does not return the expected results (eg, the object is empty), make sure that the query is input correctly. Common mistakes are related to punctuation, such as adding quotation marks around an object name (turning it into a verbatim search) when quotation marks should not be used. Punctuation is very important when coding so ensure that all commas, quotation marks, and parentheses match the examples given.