Emma Graves
April 25, 2023
OEAS 805: Env. Data Science

Taxonomy Clean-Up Pipeline Capstone Report

Abstract:

A common issue within metabarcoding analysis is that taxonomic databases do not always assign the same amount of classification or taxonomic ranks to all sequences, resulting in taxonomy files with mismatched taxonomic ranks in each column. The purpose of this capstone project was to create a python script that would clean up taxonomy files, getting rid of extra taxonomic ranks and putting each rank in the correct spot in each line. The script created successfully does that, with an 86.9% success rate on an unseen dataset of eukaryotic phytoplankton sequences and associated taxonomy. The code goes line by line and uses regular expression searching from the re python package to find common taxonomic rank suffixes and assign them to specific taxonomy variables, flagging those that are not found. The result is two files, one for successfully reorganized taxonomy and one for flagged taxonomy that needs additional attention. This pipeline is specifically designed for eukaryotic phytoplankton, but is easily editable for any additional organism group, and thus, is able to be used for a wide variety of ecological data.

Introduction:

Metabarcoding is a common method for studying ecosystem biodiversity, and typically involves using high-throughput sequencing of short DNA fragments matched to species taxonomy to describe communities (Nørgaard et al., 2021). Most often, specific regions of 18S or 16S rRNA, for eukaryotes and prokaryotes respectively, are used, since these DNA sequences are present in all living things and are highly variable (Alberti et al., 2018). The highly variable nature of 18S and 16S rRNA means that each species, and sometimes groups within species, will have a different DNA sequence for the fragment, allowing species level taxonomy to be assigned based on the DNA sequence.

After the sequencing data goes through several preprocessing programs to filter, trim, and pair the sequences, amplicon sequence variants (ASVs), or all the unique sequences found in any sample, are exported to be assigned taxonomy. Hypothetically, all Linnaean taxonomic ranks should be assigned, including Kingdom, Phylum, Class, Order, Family, Genus, and Species, however, this is not always the case. Multiple databases have been created to match taxonomy to rRNA sequences, including the NCBI Blast Database and the SILVA database (Altschul et al., 1990; Quast et al., 2012). These databases are usually open source, and scientists can upload their own sequences with associated taxonomy.

This introduces the main issue with taxonomy assignments. Taxonomy is ever changing, both in specific species names, as well as in larger rank reorganization (Mallet & Willmott, 2003). Some organisms have sub-ranks and supra-ranks, whereas others are missing normal taxonomic ranks altogether. There are no current official global standards for taxonomy, meaning that scientists can upload to these rRNA

databases with whatever format they choose. This means that, after matching sequences to taxonomy using these databases, the resulting dataset is unorganized and has rows with various lengths. One cannot pull out a specific column or line object number and get the same taxonomic rank for each line. Since downstream analysis of metabarcoding data relies on specific taxonomic ranks to be used for comparison, the taxonomy datasets need to be reorganized and cleaned. An example of downstream analysis and data presentation has been included in Figure 1 to demonstrate the importance of being able to pull a specific column and have all assignments in that column be the same rank. Furthermore, Figure 1 also provides an example of the way that taxonomy data is used: to understand the composition of a community at a certain taxonomic level.

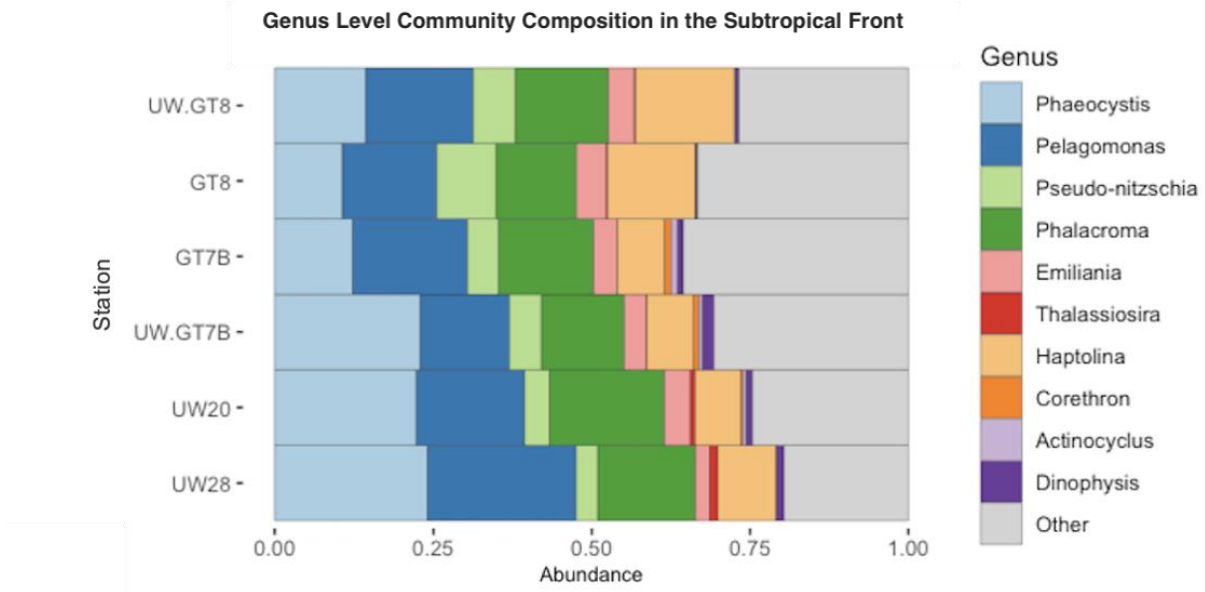

Figure 1: Example graph of organism abundance at the genus level in the Subtropical Front (STF) of the Southern Ocean. Abundance is in % of the population, and the y-axis denotes different sample stations.

The goal of this capstone project was to create a script that could be run on a taxonomy file to reorganize the rank assignments in each row and place them in the correct column or object number in each row. This was done by writing a .py script that can be run in the terminal using python, taking in the taxonomy file, cleaning it up, and returning two output files: one with good, cleaned up sequences and taxonomy, and one with flagged sequences and taxonomy that was unable to be cleaned up. A workflow script was also created in Jupyter Notebook to test each part of the code, and will be mentioned, but is not the final output of this project.

Data:

The data used for the formation and initial testing of this project was a subset of Southern Ocean phytoplankton community composition data. Specifically, a random set of 30 sequences with associated taxonomy were extracted from the much larger file of full community composition. These sequences were obtained by PCR amplifying the

highly variable V4 region of 18S rRNA, using PCR primers designed to best amplify phytoplankton species (Balzano et al., 2015). After filtering and cleaning the sequencing data using R-package DADA2, the sequences were then matched to their taxonomy using the SILVA 18S database (Callahan et al., 2016; Quast et al., 2012). This pipeline was designed to work on eukaryotic phytoplankton taxonomy but could be edited to work on a wider range of organism groups by changing the search terms for each taxonomic rank to fit those of that organism group.

Methods:

The main packages used in this script were re: Regular Expressions Operations and sys: System-specific Parameters and Functions. Both packages are included in python and do not require any additional installation. The python package pandas: Powerful Python Data Analysis Toolkit was used to read in the data in the testing workflow Jupyter Notebook script but was not used in the final .py taxonomy cleanup script. The re package uses special symbols to encompass more characters. Specifically in this pipeline, the character "\w+" means any combination of letters and parentheses around any word or character allows access to that word in the regular expression search results, meaning that word can be assigned to a new variable.

The pipeline starts by using the sys package to check for how many input variables were listed when the script was initiated in the terminal. Since the command needs to include the input file to be cleaned up, as well as "py" or "python" to denote how to run the script, there should be 3 input variables when the script is initiated. If the input into the command line has less than 3 variables, the usage statement will print in the terminal. The usage statement describes what the script does and includes instructions on how to write the command in the terminal for the script to run. If there are 3 input variables in the command, the script will then open the input file and create the two output files, one for good, successfully cleaned up taxa, and another for the flagged, unsuccessfully cleaned up taxa. The script then manually writes the header line in each output file, with the labels for each column that is needed in the final, cleaned up lines: Kingdom, Clade, Phylum, Class, Subclass, Order, Family, Genus, and Species.

Following this, a For loop is opened to read through each line of the input file, not including the first line, which is the header line. The line is split at each comma, the common delineator in most input taxonomy files, and the first object in the line, which is always the DNA sequence, is assigned to the "seq" variable. Objects or words are assigned to a variable when they are necessary for further analysis and thus, need to be included in the final output files.

Then, the script begins to search for each taxonomic rank within the line using the regular expressions package. Since 18S data was used to form this script, the only kingdom in the data is Eukaryota. Thus, the search term used for kingdom is "(\w+ota)". This searches for any word ending in -ota. Then, if the search resulted in a word found, that word is assigned to the "king" variable. Since kingdoms are not used in most further research analysis, if the kingdom is not found, an NA is assigned to the "king" variable in its place.

Clade is not an actual taxonomic rank, but it is information that can be helpful to know or look at. There are no common suffixes for clade, so this pipeline specifically searches for 6 common clades within eukaryotic phytoplankton taxonomy: Haptista, Sar,

Viridoplantae, Ochrophyta, Opisthokonta, and Metazoa. Similarly to kingdom, if any of these words appear in the line, they are assigned to the “clad” variable, but otherwise, an NA is assigned to the variable.

Phyla typically have the common ending of -phyta, but there are various other endings that show up in typical 18S data as well, including -myxa, -atea, -mycota, and -phora. This script also searches for Arthropoda and Mollusca, as those are common metazoan taxonomies found in this data. The script searches for all these phylum options, either by “(\w+suffix)” or by searching for the whole phylum directly. If found, the phylum was assigned to the “phy” variable. Unique to the SILVA database, which was used to obtain the data used in the creation of this pipeline, all alveolates are assigned Alveolata as their phylum. The actual phylum for alveolates is Myzozoa, so the pipeline searches for any instance of “Alveolata” and replaces it with Myzozoa. If none of the previous phylum suffixes or phyla were found in the line, an NA was assigned to the “phy” variable.

Moving on to class, this is where taxonomic ranks become important for downstream analysis, so this is where flags are introduced. All classes within eukaryotic phytoplankton taxonomy have the common ending -phyceae, so the regular expression search term is “(\w+phyceae)”. If the class is found, it is assigned to the “cla” variable. If not, a flag is assigned to the “cla” variable instead, indicating where the taxonomy failed. At the class level, the flag looks like “Flag: Failed at Class”.

Subclass is another term that is not technically an official taxonomic rank. Thus, if the subclass is not found, an NA is assigned instead of a flag. All subclasses in this data have the common ending -idae, so the regular expression searches for any words that end in -idae and assigns them to the “subc” variable. Otherwise, as previously mentioned, an NA is assigned to the “subc” variable.

Order and family are again both official taxonomic ranks that are used in downstream data analysis, so flagging is used for these ranks as well. All orders in this data end in -ales and all families end in -aceae. Both terms are searched for and any words that end in the order or family suffixes are assigned to the “ord” and “fam” variables respectively. If those suffixes are not found at all in the line, a flag, either “Flag: Failed at Order” or “Flag: Failed at Family” is assigned to the “ord” or “fam” variables.

Lastly, the format for species within the SILVA output is “Genus species”. Therefore, the script searches for any instance of two words with a space between them. Since some genera include dashes or periods, the full search term includes those instances and is “([\w.\-\w]+) (\w+)”. There are two separate sets of parentheses so that genus and species can be accessed from the search results separately. Within the data, some sequences are only identified up to genus, and are classified as “Unclassified genus” at the species level. To account for this, a search is done within the results from the first search for the word “unclassified”. If that is present, the second variable from the first search is assigned to the “gen” variable. If not, the first variable from the first search is assigned to the “gen” variable. This ensures that the gen variable is always the genus. Both the first and second variable from the first search are assigned to the “spec” variable for the species. If no two-word combinations are found, flags are assigned to both the “gen” and “spec” variables.

At this point, the full taxonomy has been searched for and assigned to variables. For each line, a new, cleaned up line is formed by combining the seq, king, clad, phy,

cla, subc, ord, fam, gen, and spec variables. Each line is searched for the word “Flag” to separate the flagged taxonomy from the successful taxonomy. If a flag is found, that new line is written to the flag_taxa.csv file with the original line appended to the end of it, so that any associated data can be examined when the flags are manually edited. If there are no flags in the new line, it is written to the good_taxa.csv file. The output of running this script is the two output .csv files, good_taxa.csv and flag_taxa.csv, which will be created in the current working directory.

Results and Discussion:

When running this pipeline on the small, random subset of 30 used to design the script, it correctly cleans up the taxonomy for all 30 of the sequences. The pipeline was then run on unseen data, specifically the full 4228 sequence taxonomy dataset of Southern Ocean phytoplankton community composition. Of the 4228 lines to clean up, 3673 were successful and were written out into the good_taxa.csv file. This is an 86.9% success rate on this specific unseen dataset. In the 555 flagged lines, only 93 different species were present, meaning that only 93 manual corrections would need to be made. This is far better than the thousands of manual corrections that would have been made on the data without this pipeline and will make future analysis much quicker and easier.

In its current format, this pipeline most likely only works on eukaryotic phytoplankton, since that was the type of taxonomy data used to write the script. However, the script is highly editable to include other groups. The search terms can be changed for each taxonomic rank to include the common suffix for a different group of organisms, or additional search terms can be added within the if/elif/else loops where the found words are assigned to the taxonomic rank variables.

Conclusion:

Overall, this pipeline works as expected and performs well. Moving forward, improvements could come from fixing the lines found in the flag_taxa.csv file and finding ways to incorporate the taxonomy found in those lines into the pipeline. Similar to alveolates, there may be some taxonomy that uses common names instead of the actual taxonomic rank name, which could be included in the search terms for that rank. Some of the flagged taxonomy was due to all labels being “incertae sedis”, which is a taxonomic term used for unknown or undetermined (Merriam-Webster, n.d.). The pipeline could be edited to search for any use of this term and allow that term to be assigned to any taxonomic variable. In the future, this pipeline will also be used on eukaryotic phytoplankton from other areas of the ocean than the Southern Ocean, and edits will be made to include more groups and allow for use on a wider range of phytoplankton taxonomy datasets. In conclusion, the pipeline accomplished the goal that was set out to clean taxonomy files, and future work will surely allow for a more encompassing application.

References:

Alberti, A., Poulain, J., Engelen, S., Labadie, K., Romac, S., Ferrera, I., Albini, G., Aury, J.-M., Belser, C., Bertrand, A., Cruaud, C., Da Silva, C., Dossat, C., Gavory, F., Gas, S., Guy, J., Haquelle, M., Jacoby, E., Jaillon, O., … Wincker, P. (2018). *18S*

and 16S rRNA genes amplicon generation for eukaryotic and prokaryotic metabarcoding v1 [Preprint]. https://doi.org/10.17504/protocols.io.qwhdxb6

Altschul, S. F., Gish, W., Miller, W., Myers, E. W., & Lipman, D. J. (1990). Basic local alignment search tool. *Journal of Molecular Biology*, *215*(3), 403–410. https://doi.org/10.1016/S0022-2836(05)80360-2

Balzano, S., Abs, E., & Leterme, S. (2015). Protist diversity along a salinity gradient in a coastal lagoon. *Aquatic Microbial Ecology*, *74*(3), 263–277. https://doi.org/10.3354/ame01740

Callahan, B. J., McMurdie, P. J., Rosen, M. J., Han, A. W., Johnson, A. J. A., & Holmes, S. P. (2016). DADA2: High-resolution sample inference from Illumina amplicon data. *Nature Methods*, *13*(7), 581–583. https://doi.org/10.1038/nmeth.3869

Mallet, J., & Willmott, K. (2003). Taxonomy: Renaissance or Tower of Babel? *Trends in Ecology & Evolution*, *18*(2), 57–59. https://doi.org/10.1016/S0169-5347(02)00061-7

Merriam-Webster. (n.d.). Incertae sedis. In *Merriam-Webster.com medical dictionary*. https://www.merriam-webster.com/medical/incertae%20sedis

Nørgaard, L., Olesen, C. R., Trøjelsgaard, K., Pertoldi, C., Nielsen, J. L., Taberlet, P., Ruiz-González, A., De Barba, M., & Iacolina, L. (2021). EDNA metabarcoding for biodiversity assessment, generalist predators as sampling assistants. *Scientific Reports*, *11*(1), 6820. https://doi.org/10.1038/s41598-021-85488-9

Quast, C., Pruesse, E., Yilmaz, P., Gerken, J., Schweer, T., Yarza, P., Peplies, J., & Glöckner, F. O. (2012). The SILVA ribosomal RNA gene database project: Improved data processing and web-based tools. *Nucleic Acids Research*, *41*(D1), D590–D596. https://doi.org/10.1093/nar/gks1219