**Description of function final\_metbar: organising and cleaning metabarcoding data**

Authors: Diogo Ferreira ([ferreiradfa@gmail.com](mailto:ferreiradfa@gmail.com)), Crinan Jarrett ([crinan.jarrett@vogelwarte.ch](mailto:crinan.jarrett@vogelwarte.ch)) and Lara Gross ([lara.gross@vogelwarte.ch](mailto:lara.gross@vogelwarte.ch))

There is not really an established protocol for filtering metabarcoding data; instead, different people use different methods. Depending on your research question, you may want to choose different options. Below I describe each of the filtering options implemented in the function ‘final\_metbar’.

1. Input: the function needs two datasets, the first one ‘data’ is the diet data file as described in ‘MetadataBarcodingData.docx’. In the function there is an option called ‘prefix\_control’ allows you to set the label given to extraction controls in the dataset. Make sure the prefix of column names of controls matches this setting. The second dataset ‘sample\_list’ is a file that will allow the function to match the sample number to other sample info (species, site, etc.). The sample list must contain at least 2 columns: ‘lab.nbr’ (sample lab number that matches number in diet data file) and ‘species’. Other columns e.g., site, habitat, landscape, etc are optional.
2. Filtering steps:
   1. Remove\_samples (TRUE or FALSE): if TRUE, this step removes all samples that have less reads than the controls. The logic here is that if controls have 100 reads (all due to contamination), then a sample with 90 reads will be entirely contamination and no true diet signal. However, this step is not recommended because there are several dilution steps during the lab work and the amount of DNA in samples is hard to estimate, so using an arbitrary cut-off can be too strict.
   2. Remove\_control\_asvs (TRUE or FALSE): this is a more lenient alternative to the step above. If TRUE, instead of removing entire samples, it removes ASVs that appear at high frequency (>1% of full run) in control. So, if a control sample has 100 reads of Prey1, and that constitutes 1.5% of all reads from sequencing run, that ASVs is removed from all samples. The logic is the same: if the control has a high frequency of reads of a certain ASVs due to contamination, then perhaps this ASV should be disregarded from all samples.
   3. Keep\_class (vector of class names to keep): the diet data often contains groups such as Fungi that may not be of interest. This step allows you to select which taxonomic classes you are interested in (e.g., keep\_class=c(“Arachnida”,”Insecta”)).
   4. Hits\_clean: functions like asvs\_clean, but it removes taxon hits (not ASVs) that occur at e.g., <1% of total sample reads. Example: instead of looking at ASV1, it looks at the hit "Lasius niger" or "Insecta".
   5. Asvs\_clean (any number <5, default = 1): this step removes all ASVs with less than X% of the total number of reads this sample. This is used also to try to remove ASVs that occur only due to contamination. Of course, it is possible, especially if we set a high cut-off (e.g., 5%), that we may be removing ‘true’ diet items, but in any case they are likely not the main diet items.
   6. Control\_asvs\_clean: option to clean all ASVs that occur in the control at >1% of the total reads for this ASV. (In comparison: "remove\_control\_asvs=T" removes all ASVs that appear in the control at >% of the full run, i.e. all reads irrespective of the ASV).
   7. Remove\_NAorders (TRUE or FALSE): if TRUE, all ASVs that are not identified to order level (i.e., have NA in the order column) are discarded. This option may be useful if you are interested in the composition of diet, because unidentified items are essentially useless for this type of analyses. On the other hand, if you are interested in comparing the diversity of diet between individuals, it may not matter that items are unidentified, and you can simply consider number of ASVs as an indication of dietary diversity.
   8. Remove\_NAfamily (TRUE or FALSE): if TRUE, removes all ASVs that are not identified to family level (i.e., have NA in the family column).
   9. Desired\_species (NULL, or vector of species): this refers to the predator species you are interested in. For instance if you have a dataset with many different predator species but you only want to consider the blue tits, then you could enter desired\_species=”blue tit”.
3. Output: the output of the function consists of a dataset with the following columns:
   1. Predator: the sample number
   2. Prey: the unique identifier of the diet ASV
   3. Weight: the number of reads corresponding to that ASV in the predator sample
   4. Otu\_phylum: taxonomic assignment of ASV
   5. Otu\_class: taxonomic assignment of ASV
   6. Otu\_order: taxonomic assignment of ASV
   7. Otu\_family: taxonomic assignment of ASV
   8. Otu\_genus: taxonomic assignment of ASV
   9. Otu\_species: taxonomic assignment of ASV
   10. Species: the species of the predator
   11. Site: site at which predator was caught
   12. Total reads: the total number of reads from that sample (i.e., summed across all ASVs) after cleaning.
   13. Total ASVs: the number of distinct ASVs found in that sample after cleaning.
   14. Proportion: the proportion of total reads for each sample made up by each ASV after cleaning.
   15. Reads\_hit: number reads for this taxon hit in a sample after cleaning.
   16. Reads\_hit\_proportion: proportion of number reads for this taxon hit to all reads in a sample after cleaning.