ReadMe for lipidwrap script – a wrapper for LipidR (<https://www.lipidr.org/>)

1. Set up
   1. Generate a new R script. Set working directory to folder containing the project
   2. Source the lipidwrap script using source(“/path/to/lipidwrap/script.R”)
   3. Organize your data. There should be 2 files for each lipidomics experiment
      1. Data file: a csv file with first column lipid names. Each subsequent column represents one sample with sample name in row 1 and values for lipid species below
      2. Samples file: a keyfile with the following columns:
         1. sample: sample names should match row 1 of data file
         2. Group: main grouping variable. For FCG experiments, “Group” column should have the following categories: XX\_ov, XX\_test, XY\_ov, XY\_test
         3. Any other grouping columns you’re interested in. FCG experiments should have columns “chromSex” and “gonSex”, which have values of XX and XY and O and T, respectively.
2. Preprocess lipidomics
   1. Purpose: creates a LipidomicsExperiment object from data file and samples file.
   2. Call the function using preprocess\_lipidomics with the following arguments:
      1. String experiment name – this will be used to name all of your files/plots and save them and MUST be unique for every experiment, or files will be overwritten
      2. String raw data file name – ending in .csv
      3. String sample file name – ending in .csv
      4. Optional: method. This is the method for transforming the data. Default is log fold change, but if the original data has a lot of zeros, you’ll want to specify method = “boxcox” which applies a boxcox transformation
   3. preprocess\_lipidomics will return a LipidomicsExperiment object (referred to as d in this readme)
   4. Function will by default will eliminate any lipid species for which more than 1/3 of the samples are missing a value. It will also use the 10 nearest neighbors for k nearest neighbors imputation of missing values. These parameters can be set using the percna and knn parameters, respectively.
3. Optional intermediate step: subsetting
   1. To access the lipid species information from the LipidomicsExperiment object, use rowData(d)
   2. To access the sample information, use colData(d)
   3. To subset either only some samples or some lipid species, standard R subsetting operations can be used. For example, the following will produce pc\_lipids, a LipidomicsExperiment object with only lipids from the classes PC, PCO, and PCP.

pc\_lipids = [rowData](https://rdrr.io/pkg/SummarizedExperiment/man/SummarizedExperiment-class.html)(d)$Class [%in%](https://rdrr.io/r/base/match.html) [c](https://rdrr.io/r/base/c.html)("PC", "PCO", "PCP")

d\_pc = d[pc\_lipids,]

1. Analyze lipidomics experiment
   1. There are currently pre-written functions for analyzing common structures of data (like FCG) that explicitly do ALL analysis steps using pre-defined variables, like XX, XY, O and T for FCG.
   2. get\_mva will do a PCA on d using d and an experiment name, saving a PNG of the PCA, and will return a data table with the raw data from the PCA
   3. get\_sup\_mva will do OPLS-DA on d using d, an experiment name, and a “grouping” variable, which represents the variable upon which the OPLS-DA will be generated. For FCG, this is either “chromSex” or “gonSex”. This will save a PNG of the OPLS-DA plot and return a data table with raw data from the analysis.
   4. de\_analysis is performed for all pairwise comparisons of interest. It is called using the LipidomicsExperiment d plus whatever comparison you’re interested in. For example, to assay the gonadal effect in XX mice, use the following command: de\_analysis(d, XX\_ov - XX\_test). You can also do de\_analysis by grouping two or more primary groups from the Group column by calling de\_analysis with the optional variable group\_col and specifying it to be whatever group you’d like to compare. For example, use de\_analysis(d, group\_col="chromSex", XX - XY) to collapse O and T and instead just consider XX and XY genotypes, agnostic to gonadal sex.
   5. lipidset function will use a de\_analysis return object (a dataframe) and the experiment name and will perform lipid set enrichment. A plot will be created showing the comparison and the difference in lipid class, chain length, and chain unsaturation.
   6. After all lipidset comparisons are complete, their results are aggregated together and saved as a csv ending with “\_lsea\_results.csv”
   7. composite\_volcano produces a volcano plot with a variable number of de\_analysis volcano plots and can be called by passing in a list of de\_analysis results along with the experiment name and an optional additional identifier that will be included in the title and filename of the PNG containing the plot.
   8. Next, all de\_analysis results should be concatenated together and saved as a single csv.
   9. Finally, the analyze\_ functions will create a lipidrResults object: a list of ALL of the raw data for a given analysis accessible by result type. This object is returned by the analyze\_ functions and its components can later be used independently from the analyze\_ functions.
2. Notes:
   1. The user can create new analyze\_ functions based on their needs. Copy the format of the analyze\_ functions at the top of the lipidwrap script. It is best to use descriptive variable names as these will be used to name the files produced by the script. For example, the analyze\_fcg function uses variable names like chr\_ov, which represents the chromosomal effect with ovaries background.
   2. For analysis with only two groups instead of 4, refer to the structure of the analyze\_kdm5c function. Basically, there are many fewer comparisons that can be made and the function is much simpler, but the outputs are all the same.
   3. All of the functions that are called in the analyze\_ functions can be used individually and can have more specific parameters assigned when they’re called individually instead of as part of the analyze\_ functions. For example, if you’re doing lipidset enrichment and want multiple plots representing different comparisons to have the same Y axis limits, you can specify the fold\_change parameter to set the min and max Y axis values. For example, fold\_change=4 will set the limits to -4 and 4. The default is NA and will be set automatically based on the min and max FC for that particular comparison.