MaAsLin User Guide v2.0

October 15, 2012

Timothy Tickle and Curtis Huttenhower

**Table of Contents**

A. Introduction to MaAsLin

B. Related Projects

C. MaAsLin Inputs

D. Process Flow Overview

E. Process Flow Detail

F. Expected Output Files

G. Description of Output Files

**A. Introduction to MaAsLin**

MaAsLin is a multivariate statistical framework that finds associations between clinical metadata and some data. In our context we use it to assess associations between clinical metadata and microbial community relative abundance or function. Although, this context is the reason MaAsLin was developed it is not required for it's use. Essentially MaAsLin performs boosted additive general linear models between one group of data (metadata/the predictors) and another group (in our case relative taxonomic abundances/the response).

Given that metagenomic data is sparse, the boosting is used to select metadata that show some potential to be useful in a linear model between the metadata and abundances. This is similar to any model selection or regularization technique. In the context of metadata and community abundance, a sample's metadata is boosted for one otu. The metadata data that is selected for use by boosting is then used in a general linear model using metadata as predictors and otu abundance as the response. This occurs for every otu and sample. Given we work with proportional data, the Yi (abundances) are arcsin(sqrt(Yi)) transformed. A final formula for the association is as follows arcsin(sqrt(Yi)) = bo sum BiXi + ei . If you are not working with proportional data or have a different transformation you prefer, this can be updated in the .R file described later in section “C. MaAsLin inputs”.

**B. Related Projects**

Two other projects exist at www.bitbucket.com which may help in your analysis are MetaCheck and QiimeToMaAsLin.

**MetaCheck** is a utility project, still underdevelopment, which is targeted at diagnosis and visualizing the metadata in a study. The intent is to create visualizations to comment on the balance, correlation structure, and other properties of your metadata to better use MaAsLin.

**QiimeToMaAsLin** is a project that reformats abundance files from Qiime for MaAsLin. Several formats of Qiime consensus lineages are supported for this project.

**C. MaAsLin Inputs**

There are 3 required files for each project, the "\*.read.config" file, the "\*.pcl" file, and the "\*.R" script. Although the "\*" in the file names can be anything, it needs to be identical for all three files. All three files need to be in the ../sfle/input/maasalin/input/ folder. Details of each file follow:

1. "\*.pcl"

A PCL file is the file that contains all the data and metadata. This file is formatted so that metadata/otu are rows and samples are columns. All metadata rows should come first before any abundance data. The file should be a tab delimited text file with the extension ".pcl" .

2. "\*.read.config"

A read config file allows one to indicate what data is read from a PCL file without having to change the pcl file or change code. This means one can have a pcl file which is a superset of metadata and abundances which includes data MaAsLin you are not interested in for the run. This file is a text file with ".read.config" as an extension. This file is later described in detail in section “D. Process Flow Overview” subsection “4. Create your read.config file”.

3. "\*.R"

The R script file is using a call back programming pattern which allows one to add/modify specific code to customize analysis without touching the main MaAsLin engine. A generic R script is provided “maaslin\_demo.R” and can be renamed and used for any study. The R script can be modified to add quality control or formatting of data, add ecological measurements, affect the MFA visualization, or other changes.

**D. Process Flow Overview**

1. Obtain your abundance table.

2. Obtain your metadata.

3. Format and combine your abundance table and metadata as a pcl file for MaAsLin.

4. Create your read.config file.

5. Create your R script or use the default.

6. Place .pcl, .read.config, .R files in ../sfle/input/maasalin/input/

7. Run.

8. Discover amazing associations in your results!

**E. Process Flow Detail**

1. Obtain your abundance table.

Abundance tables are normally derived from sequence data using Mothur or Qiime. Please refer to their documentation for further details.

2. Obtain your metadata.

Metadata would be information about the samples in the study. For instance, one may analyze a case / control study. In this study, you may have a disease and healthy group (disease state), the sex of the patents (patient demographics), medication use (chemical treatment), smoking (patient lifestyle) or other types of data. All aforementioned data would be study metadata. This section can have any type of data (factor, ordered factor, continuous, integer, or logical variables). If a particular data is missing for a sample for a metadata you may leave it blank or write NA. It is preferable to write NA so that, when looking at the data, it is understood the metadata is missing and it's absence is intentional and not a mistake

If you are not manually adding metadata to your abundance table, you may be interested in an associated tool to help combine your abundance table and metadata to create your pcl file. If so, please collect your metadata and place it in a file as follows:

i. Row 1 is expected to be the ID for the metadata

ii. Column 1 is expected to be the sample IDs which should match your abundance table.

iii. Each row represents the metadata associated with 1 sample

iv. The file is tab delimited.

v. An example is found in this project at maaslin/input/for\_merge\_metadata/maaslin\_demo\_metadata.tsv

3. Format and combine your abundance table and metadata as a pcl file for MaAsLin.

Please note two tools have been developed to help you! If you are working from a Qiime output and have a metadata text file try using QiimeToMaaslin found at bitbucket. If you have a tab delimited file which matches the below .pcl description (for instance MetaPhlAn output) use the merge\_metadata.py script provided in this project (maaslin/src/merge\_metadata.py) and documented in maaslin/doc/Merge\_Metadata\_Read\_Me.txt .

PCL format description:

i. Row 1 is expected to be #ID\_indicator and then sample ids in each following column separated by tabs.

ii. Rows of metadata. Each row is one metadata, the first column entry being the name of the metadata and each following column being the metadata value for that each sample.

iii. Row of taxa/otu abundance. Each row is one taxa/otu, the first column entry being the name of the taxa/otu followed by abundances of the taxa/otu per sample.

iv. Abundances should be normalized by dividing each abundance by the sum of the column (sample) abundances.

v. Here is an example of the contents of an extremely small pcl file; another example can be found in this project at maaslin/input/maaslin\_demo.pcl .

#SampleID Sample1 Sample2 Sample3 Sample4

metadata1 True True False False

metadata2 1.23 2.34 3.22 3.44

metadata3 Male Female Male Female

taxa1 0.022 0.014 0.333 0.125

taxa2 0.406 0.029 0.166 0.300

taxa3 0.571 0.955 0.500 0.575

4. Create your read.config file.

A \*.read.config file is a structured text file used to indicate which data in a \*.pcl file should be read into MaAsLin and used for analysis. Allows one to keep their \*.pcl file intact while varying analysis. Hopefully, this avoids errors which may occur while manipulating the pcl files.

Here is an example of the contents of a \*.read.config file.

Matrix: Metadata

Delimiter: TAB

Read\_PCL\_Columns: 2-15

Read\_PCL\_Rows: 2-4,6,8,10-12

DT\_Numeric: 6

DT\_Factor: 2-4

DT\_Logical: 11

DT\_Integer: 8,10,12

Matrix: Abundance

Delimiter: TAB

Read\_PCL\_Columns: 2-15

Read\_PCL\_Rows: 18-100

DT\_Numeric: 18-100

The minimal requirement for a MaAsLin .read.config file is the Matrix: should be specified. Metadata needs to be named "Matadata" for the metadata section and "Abundance" for the abundance section. “Read\_PCL\_Rows:” is used to indicate which rows are data or metadata. Rows can be identified as ids or their row number (starting with 1, which should be an ID and not used here). If a beginning or ending name/number is missing, the rows are read from the beginning or to the end respectively.

A minimal example is shown here:

Matrix: Metadata

Read\_PCL\_Rows: -Weight

Matrix: Abundance

Read\_PCL\_Rows: Bacteria-

With this minimal example, the delimiter of the file is assumed to be a tab, all columns are read and the data types of each metadata and abundance entry are inferred by the system. If you would like more explicit control on the interpretation of the type of data you enter, use the “DT\_Factor:”, “DT\_Integer:”, “DT\_Logical:”, “DT\_Numeric:”, and “DT\_Ordered\_Factor:” Commands to force data typing. “Read\_PCL\_Columns:” can be used to select columns/samples in the same way rows were in the example.

5. Create your R script or use the default.

The R script is used to set default thresholds, to add code which manipulates your data before analysis, and for manipulating the multifactoral analysis figure. A default “\*.R” script is available with the default MaAsLin project at maaslin/input/maaslin\_demo.R .

6. Place .pcl, .read.config, .R files in ../sfle/input/maasalin/input/

All studies should have these 3 input files in the maaslin/input folder.

7. Run.

Go to ../sfle and type the following: scons output/the\_Name\_of\_your\_pcl\_file\_without\_the\_extension

8. Discover amazing associations in your results!

**F. Expected Output Files**

The following files will be generated per MaAsLin run. In the following listing the term *projectname* refers to what you named your “\*.pcl” file without the extension.

**Troubleshooting:**

*projectname*.txt

Contains the detail for the statistical engine. Is useful for detailed troubleshooting.

data.tsv

The data matrix that was read in (transposed). Useful for making sure the correct data was read in.

data.read.config

Can be used to read in the data.tsv .

metadata.tsv

The metadata that was read in (transposed). Useful for making sure the correct metadata was read in.

metadata.read.config

Can be used to read in the data.tsv .

read\_merged.tsv

The data and metadata merged (transposed). Useful for making sure the merging occurred correctly.

read\_merged.read.config

Can be used to read in the read\_merged.tsv .

read\_cleaned.tsv

The data read in, merged, and then cleaned. After this process the data is written to this file for reference if needed.

read\_cleaned.read.config

Can be used to read in read\_cleaned.tsv .

ProcessQC.txt

Contains quality control for the MaAsLin analysis. This includes information on the magnitude of outlier removal.

**Analysis:**

*projectname*-metadata.txt

Each metadata will have a file of associations. Any associations indicated to be performed after initial boosting is recorded here. Included are the information from the final general linear model (performed after the boosting) and the FDR corrected p-value (q-value). Can be opened as a text file or spreadsheet.

*projectname*-metadata.pdf

Any association that had a q-value less than or equal to the threshold given in the “\*.R” file (default is 0.25) will be plotted here. If this file does not exist, the projectname-metadata.txt should not have an entry that is less than or equal to the threshold. Factor data is plotted as knotched box plots; continuous data is plotted as a scatter plot with a line of best fit.

*projectname*.pdf

Contains the multifactoral analysis visualization. This visualization is presented as a build and can be affected by modifications in the R.script

*projectname* \_Summary.txt

Any entry in the projectname-metadata.pdf are collected together here. Can be opened as a text file or spreadsheet.

**G. Description of Output Files**

**H. Troubleshooting**

1. When using the command "scons output/maaslin/..." to run my projects I get the message:

ImportError: No module named sfle:

File "/home/user/sfle/SConstruct", line 2:

import sfle

Solution: You need to update your path. On a linux terminal in the sfle directory type the following.

export PATH=/usr/local/bin:`pwd`/src:$PATH

export PYTHONPATH=$PATH

2. When trying to run a script I am told I do not have permission even though file permissions have been set for myself.

Solution: Most likely, you have not set the script you wrote as an executable.

If you right click on your file, you should be able to change the properties of the file to be executable.