**README**

Thank you for checking out the **foxy\_qtl\_pipeline**. This is a series of computer scripts that performs quantitative trait loci (QTL) analysis of biparental genetic mapping populations. Essentially, it is a series of **R/qtl** [1] scripts called by a python program on a Linux computer server.

It can also be run on a MacBook or other Linux desktop but the analysis of many traits in parallel (server mode) is not recommended.

Currently it automates several types of analysis

1) A genome scan with a single QTL model (significance: p-value < 0.05, based upon 1000 permutations).

2) A two-dimensional genome scan using a 2-QTL model (significance: p-value < 0.05, based upon 200 permutations).

3) A stepwise model selection (significance based upon penalized LOD score, estimated in either the first 2 analysis methods).

4) Single QTL model genome scan and stepwise model analysis of function valued traits (temporal or spatial) [2].

How to launch an analysis

1) Download all relevant files from GitHub:

<https://github.com/maxjfeldman>

You’ll want to run this program on a large computer server. Our current implementation requires 1-2 GB of RAM per trait at time of analysis. This can be an issue if you want to analyze all traits in parallel on a small computer.

The efficiency could be improved rather easily but I’ve never had to do more than 50-100 traits in parallel.

2) Format your input data.

Current format this is a .csv file that contains a few fields that aren't necessary for the analysis. Please see file named: “qtl\_data\_format\_example.csv”.

Column headers:

Obs – This is just an number entry number sometimes recorded by an instrument but can be as simple as row number

experiment – This is usually an identifier that specifies a 2 character description of experiment and the year.

year – What year is the data from?

treatment – Is there a contrast being performed (wet v. dry | dense v. sparse)? If no treatment just specify the same string for each entry (“none” for example).

plot – Setaria grow outs can be summarized by overall plot. If you experiment does not contain multiple plot you can just specify the plot using the same string for each entry (“none” for example)

subplot\_id – This is a string that associates the individual plant location within the plot (can also specify “none”).

id – This is the name of the RIL. The nomenclature you use must match the names in the genetic map. We generally use the format “RIL\_001”.

sampling – This is a subcategory that can be used to distinguish between identical plants sampled at different time points (usually, this is imputed as “none”).

The remaining columns contain phenotypic values (height\_25, height\_46, height\_67).  
  
\*\* The most important detail is that trait names cannot contain spaces or periods. Periods are currently used as a field delimiter to distinguish between the same trait measured in different treatments.

3) Call the program.

Change into the directory where you have downloaded the script and use the following function call:

[user@computer~]$ python foxy\_qtl\_pipeline.py –i input\_file.csv -o name\_of\_output\_directory -c [y|n] -m name\_of\_genetic\_map -s [y|n] -t riself

There are several arguments that need to be specified:

i – This is the name of the input file (see qtl\_data\_format\_example.csv)

o – This the name of the directory where you will store the results

c – Are you doing a comparison between treatments (wet v. dry) or not? “y” indicates that the treatment field in the input file contains 2 different levels

m – File name of your genetic map formatted for R/qtl csvs input.

s – Are you running this analysis on a server? If you select “y” traits will be analyzed in parallel. If you select “n” they will be run consecutively.

t – Type of mapping population. Currently I am working with a F7 RIL mapping population so I use ‘riself’.

Other arguments that will be added in the future:

d – distribution/model type (is this a normal, binary, or 2-part or non-parametric)

q – Method to use for QTL analysis (currently all is done using Haley-Knott regression)

f – Is this a functional trait (time-series)? [y|n] \*\*Currently we have a separate set of scripts to do this type of analysis. See below.

Example: