### Part of Hasiba Asma’s dissertation: Chapter 2 (Section 2.2.5)

### 2.2.5.Orthology mapping

The main goal of my research is to trace the evolutionary history of transcriptional enhancers. For this purpose, a set of orthologous enhancers from more than one species is required. Having power of cross species prediction using SCRMshaw gave us extensive predicted enhancers from many species of our interest (within Holometabola). After running SCRMshaw, we chose to map the associated genes from each species to their *Drosophila melanogaster* orthologs since we train our SCRMshaw against known *melanogaster* enhancers and it is also known to have the most well characterized gene annotation. To accomplish this, I built an orthology mapping pipeline which maps all genes that are associated with each SCRMshaw prediction of respective species to their fly orthologs.

#### 2.2.5.1. Detailed Protocol

One of the required files for this pipeline to run is a FASTA-formatted file of all annotated proteins, which (when available) can be downloaded from NCBI along with the genome fasta and GFFv3 annotation files. The protein fasta files of *D. melanogaster* and of the current species of interest are used as an input to run *Orthologer*, ortholog-determining software from the Zdobnov lab [ref] (which is the software behind orthoDB [ref]). Orthologer uses blat [cite] for protein sequence alignment and returns the best reciprocal hits for each protein. To run *Orthologer*, I followed the steps provided on the [links fed into an orthology mapping (OM) script that is written in python to post process the data from *Orthologer* and convert internal ids to the known fly gene symbols [figure] and map fly ortholog to each of SCRMshaw prediction.

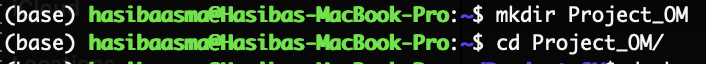
The input files required for this script are as follows:

* The output file from *Orthologer* (og\_map file) [file 1]
* Mapping files (file that will have Protein ids (used for *Orthologer*) mapped to associated gene ids (present in SCRMshaw predictions)) [file 2]
* SCRMshaw’s output list of predicted enhancers [file 3]

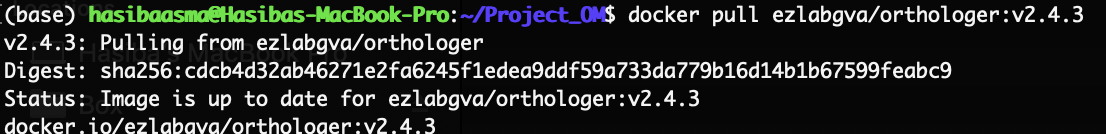
##### Step 1: Map orthologs between Dmel and species X using Orthologer

To run *Orthologer* [to get file 1], a protein fasta file is required for the species of interest, for which we want to map fly orthologs. Once we have that, we can run *Orthologer* [ref].

* Initialize docker
* Created a project directory and cd to that directory



* Following steps are copied from: <https://hub.docker.com/r/ezlabgva/orthologer>
  + Pull docker
    - *docker pull ezlabgva/orthologer:v2.4.3*



* + cd orthologer\_container
    - *docker build -t orthologer*
  + Set a working environment in directory $(pwd) as user $(id -u)
    - *docker run -u $(id -u) -v $(pwd):/odbwork ezlabgva/orthologer:v2.4.3 setup\_odb.sh*

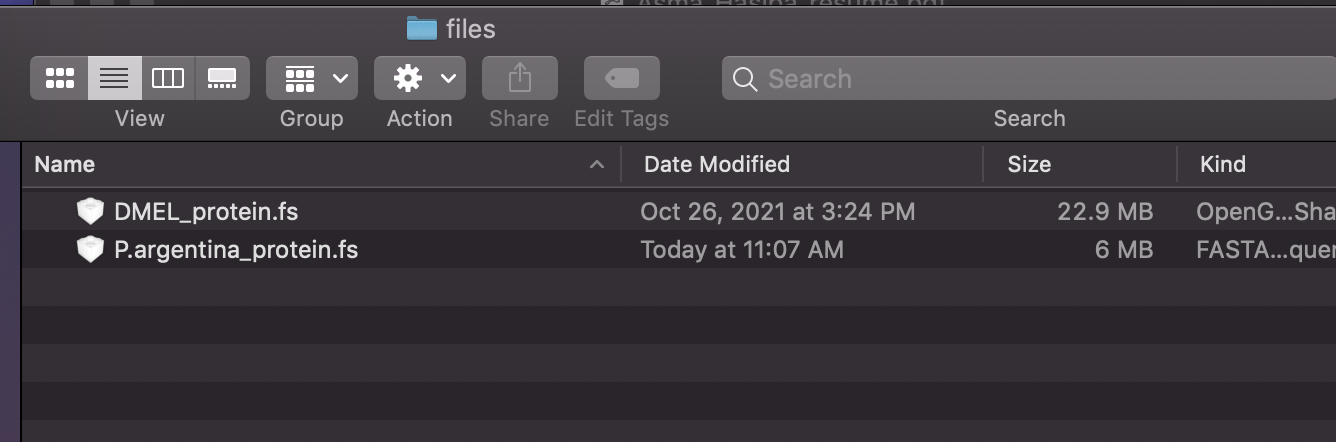
Text

Description automatically generated

* + This will generate the following files in our empty directory

A screenshot of a computer

Description automatically generated with medium confidence

* + Create a folder “files” and put input protein FASTA file of *Drosophila melanogaster* (base species) and species X (*P.argentina* in the example below)  with the extension .fs  
  + Then execute the following command and it will create a text file specifying which protein FASTA files to be imported.
    - *for x in $(ls files/\*.fs); do echo "+$(basename $x .fs) $x"; done > mydata.txt*

A screenshot of a computer

Description automatically generated with medium confidence

* + Import the proteomes specified in the mydata.txt file by following command
    - *docker run -u $(id -u) -v $(pwd):/odbwork ezlabgva/orthologer:v2.4.3  ./orthologer.sh  manage -f mydata.txt*

Text

Description automatically generated

* + Run the container
    - *docker run -u $(id -u) -v $(pwd):/odbwork ezlabgva/orthologer:v2.4.3  ./orthologer.sh -t todo/mydata.todo -r all*

Graphical user interface, text

Description automatically generated

* Once orthologer finishes running it creates output files in the Results, Cluster and Rawdata directories. Create a subfolder in the project directory “orthologer\_output” and copy the following files of interest from orthologer’s output. Discard all the rest of its output files (for memory’s sake)
  1. Cluster/mydata.og\_map : this file have all the list of internal mapped ID (ortho/paralogs) between *Drosophila melanogaster* and speciesX (*P.argentina*)
  2. Rawdata/DMEL\_protein.fs.maptxt & Rawdata/Xspecies\_protein.fs.maptxt (P.argentina\_protein.fs.maptxt e.g): these are the mapping files of internal gene\_IDs to IDs in original protein FASTA files

Text

Description automatically generated

A screenshot of a computer screen

Description automatically generated with medium confidence

##### Step 2: Map internal protein ids to commonly known gene ids for species X:

Now we have protein IDs mapped [from step 1], but the problem is that our SCRMshaw predictions have associated gene-ids and not protein-ids. We therefore need another intermediate mapping file[file 2] which would have each gene ID –protein ID that goes with it. To get that ‘mapping file’ I followed these steps:

* This is extracted from GFF with following command,
  1. *perl -F'\t' -ane 'next if $\_ =~ /#/; if ($F[2] =~ /CDS/){$F[8] =~ /gene=(.\*?);.\*protein\_id=(.\*)/; print "$1\t$2\n";}'  speciesX.gff > speciesX\_A.gff*
* But first copy GFF to your project directory and then execute the above command

Text

Description automatically generated

* Open speciesX\_A.gff, and made sure column1 matches with the gene-IDs that are present in speciesX\_SCRMs\_file (col # 6) and column 2 has the ids that matches with protein-ID that are present in mydata.og\_map file. If there are un-necessary things, get rid of them, and if there is something that needs to be added, add them. E.g this is how P.argentina\_A.gff looks like without editing.

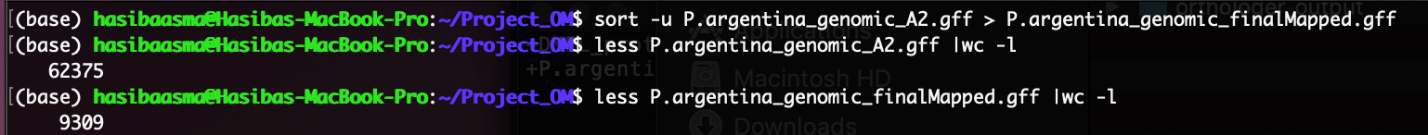
Text

Description automatically generated

* Looking at this file we can see that protein-ids (2nd) column matches with the format of *Orthologer* output but gene-ids 1st column looks slightly differently formatted than SCRMshaw output as those have ‘gene-’ in the beginning. We therefore edit the file so that it looks like this:

Graphical user interface, text

Description automatically generated

* Finally, to make sure there’s no repetition of gene mapping lines, we saved only unique values by following this command. We can see there were lots of duplicates now eliminated in our final file. species\_X\_finalMapped.gff(P.argentina\_genomic\_finalMapped.gff)
* We keep file 2(*P.argentina*\_genomic\_finalMapped.gff) and discard original gff and intermediate files (*P.argentina*\_genomic.gff & *P.argentina*\_genomic\_A and A2.gff)

Text

Description automatically generated

Finally, we need to copy file 3 i.e., SCRMshaw’s output file of species X (scrmshawOutput\_peaksCalled\_allSets\_allMethods\_Parg.bed) to our project directory as follows:

*$cp ~/scrmshawOutput\_peaksCalled\_allSets\_allMethods\_Parg.bed .*

Graphical user interface, text, application

Description automatically generated

##### Step 3: Map DMEL orthologs to each of species X SCRMshaw prediction

Now that we have all the required files[1,2,3] in our project directory, the next step is to clone and run the scripts that would actually map fly ortholgs to each of our SCRMshaw prediction in the SCRMshaw output file.

* First copy the script to our project directory
  1. *git clone* [*https://github.com/HalfonLab/Mapping-D.mel-Orthologs.git*](https://github.com/HalfonLab/Mapping-D.mel-Orthologs.git)
* Change to the Mapping-D.mel-Orthologs directory

Text

Description automatically generated

* Copy all the files from other main(Project\_OM) and subdirectory(orthologer\_output) to this folder

Graphical user interface, text

Description automatically generated

* Execute the script with the following command

*$python OM\_mappingFlyOrthologsToSCRMshawPredictions.py -ft GCF\_000001215.4\_Release\_6\_plus\_ISO1\_MT\_feature\_table.txt -mD DMEL\_PROTEIN.fs.maptxt -mX P.ARGENTINA\_PROTEIN.fs.maptxt -og mydata.og\_map -sp1id P.argentina\_genomic\_finalMapped.gff -so scrmshawOutput\_peaksCalled\_allSets\_allMethods\_Parg.bed*

* The output file will have same file name as SCRMshaw’s output file with an additional “SO\_” prefix (SO\_scrmshawOutput\_peaksCalled\_allSets\_allMethods\_Parg.bed) and the same format but with addition of *Dmel* orthologs. It is in the form of a 18-column tab delimited file as follows:
  1. Chromosome
  2. *start*
  3. *end*
  4. *Peak amplitude*
  5. *SCRMshaw score*
  6. *flanking gene*
  7. *Dmel Ortholog of flanking gene*
  8. *distance of hit from flanking gene (basepairs)*
  9. *location of hit relative to flanking gene*
  10. *local rank*
  11. *next closest flanking gene*
  12. *Dmel Ortholog of next flanking gene*
  13. *distance of hit from flanking gene (basepairs)*
  14. *location of hit relative to flanking gene*
  15. *local rank*
  16. *training set*
  17. *method (hexmcd, imm, pac)*
  18. *rank*

This pipeline has been a very useful addition to our toolset because now when we run SCRMshaw on a given species, running the output through this pipeline provides easy-to-recognize fly ortholog names rather than just gene IDs for each prediction. For example, if SCRMsahw has a prediction in the *Anopheles gambiae* genome associated with gene-ID AAEG001412, it would be difficult to to recognize the role or function of the gene. However, our pipeline now provides a fly ortholog—in this case, “wg”—which provides instantly recognizable insight into the gene’s identity and function. This has tremendously helped us in choosing enhancers for in vivo validation [more details to follow in chapter 3].