2019 Genomics Workshop – Practical on Phylogenomics

Jacob L. Steenwyk & Antonis Rokas

Programs that you will use: Bash, Python (modules: sys), Mafft, trimAl, awk, IQ-Tree, RAxML, FigTree

In this workshop practical, we will reenact the following research scenario. The Brewmaster of a major local brewery (and, as some of you know by now, there are many in the Czech Republic!) has sent you a yeast species isolated from one of the oak barrels that she uses for beer storage. To determine the identity of the species and set the stage for further analyses, you have sequenced and assembled the yeast isolate's genome and now wish to create a phylogeny to understand what yeasts it is most closely related to. Thus, we will build a species phylogeny using multiple loci.

To do so, we will identify single copy orthologous genes in a set of yeast taxa and subsequently align, trim, and concatenate the single copy orthologous genes. Although we will be working with a smaller set of genes, this methodology can in principle be applied to any number of loci (the computational time required for the analysis of your data is going to be the key constraint).

This methodology has been used to infer and scrutinize species tree topologies in numerous recent publications using thousands of genes and greater than 300 taxa including Shen *et al.* (2018), *Cell* and Steenwyk *et al.* (2018), *bioRxiv*.

Note, steps within objectives that have fill-in-the-blank prompts are indicated as such using blue color font. Please fill in these prompts.

Protocol

1) Examine the set of taxa that will be used to infer a species tree from Shen *et al.* 2018, Cell (https://www.cell.com/cell/fulltext/S0092-8674(18)31332-1).

• Taxa will be a subset of 20 yeast species used in Shen *et al.* 2018.

• Genome fasta files are in directory *FILES_genome_files*

• Brewery_genome.fna is the newly sequenced genome

Objectives:

i) Examine the contents of the directory with the *ls* command

ii) Confirm that there are 20 genome fasta files and that Brewery_genome.fna is present

2) Identify BUSCO genes across the 20 yeast taxa using the Ascomycota database of universally single copy orthologous genes provided by OrthoDB

• To identify orthologous genes across the taxa, we will infer universally single copy orthologous genes (or BUSCO) genes in each individual genome using the Ascomycota database of BUSCO genes

• This analysis takes too long to be part of the phylogenomics practical

• Results from BUSCO analysis have been provided for you in directory *FILES_busco.* These results were obtained by running the following command:

python BUSCO.py -i <input_fasta_file> -o <output_file_name> -l <orthoDB_database> -m genome

where <*input_fasta_file*> was one of the genome fasta files, <*output_file_name*> is the name of the output directory that will be made by BUSCO, and <*orthoDB_database*> is a directory of BUSCO genes which can be downloaded from the BUSCO website (https://busco.ezlab.org/) under the section datasets, and *-m genome* lets BUSCO know you will be running the analysis on a genome assembly (as opposed to a proteome or transcriptome one). In this case, the database of BUSCO genes used was the Ascomycota database.

Objectives:

i) Examine the contents of the *FILES_busco* directory with the *ls* command

ii) Go into the output directory of the newly sequenced genome using the *cd* command

iii) Examine the contents of the directory with the *ls* command

iv) Read the contents of the *short_summary_Brewery_genome.fna.txt* file. How many BUSCO genes were used to determine genome completeness and how many are considered present in the genome as complete (i.e., single copy and duplicated), fragmented, or missing?

• Number of BUSCO genes used:

• Number of BUSCO genes that are complete:

- Single copy complete:
- Duplicated:

• Fragmented:

• Missing:

v) Examine the contents of directory *single_copy_busco_sequences* using the *ls* command. This directory is populated with amino acid (denoted by *.faa*) and nucleotide (denoted by *.fna*) fasta files of single copy BUSCO genes (denoted by *EOG092D**) that were identified in the genome fasta file *Brewery_genome.fna*.

3) Align and trim single copy BUSCO genes

• To infer the evolutionary history among these taxa, we will need to create an aligned super-matrix (or concatenation matrix) of all BUSCO genes. But before we do so, we must first align and trim each individual gene

• In this step, we will align and trim all single copy BUSCO genes

Objectives:

i) Examine the contents of the directory *FILES_usco_fas*

ii) How many multi-fasta BUSCO genes are there with 100% taxon occupancy (i.e., with a gene sequence from every species that we analyzed)? Hint: list how many fasta files there are and then count how many there are using *wc -l*.

Number of BUSCO genes with 100% taxon occupancy:

iii) Align sequences for every multi-fasta BUSCO gene. To do so, execute the following command from directory *FILES_usco_fas*

for i in $(ls EOG092D.fa); do mafft --inputorder --bl 62 $i > $i.mafft ; done*

This command will loop through each multi-fasta file using a *for loop* and align the sequences using the BLOSUM62 matrix of substitutions (*--bl 62*) and maintain the order of sequences in the output file (*--inputorder*). Lastly, the command will direct all output to a new file with the same name as the input file but with '*.mafft*' appended to the end (*$i.mafft*)

iv) Type the *ls* command and examine the contents of the directory. At this stage, the number of files is becoming unwieldy and it would be good practice to organize your files. To do so, create a new directory to store unaligned files using the command '*mkdir FILES_unaligned*' and move all unaligned files into this directory by executing the following command:

*mv *.fa FILES_unaligned*

Type the *ls* command to confirm files have been successfully moved (you may also want to check the contents of the new directory to ensure that the moved files are actually now there).

v) Trim sequences for every aligned multi-fasta BUSCO gene. To do so, execute the following command from the directory *FILES_usco_fas.*

*for i in $(ls EOG092D*mafft); do trimal -in $i -out $i.trimal -automated1 ; done*

This command will loop through each aligned multi-fasta file using a *for loop* and trim spurious sequences or poorly aligned regions using the *-automated1* set of parameters and placing the trimmed alignment into a specified output file (*$i.trimal*)

vi) Type the *ls* command and examine the contents of the directory. As was done in step *iii*, clean up the directory by creating separate directories for the aligned and trimmed and aligned files named *FILES_aligned* and *FILES_trimmed*. Move aligned and trimmed multi-fasta files to their appropriate directories.

4) Concatenate single copy BUSCO genes

• To create a concatenation matrix of all sequences in the directory *FILES_trimmed*, we will use a custom script:

Objectives:

i) To use the custom script, *concatenate_seqs_phyloPractical.py*, we will need to create an input file which contains a list of fasta files whose sequences will be concatenated. To do so, execute the following command in *FILES_trimmed*:

*ls *trimal > fastaFiles.list*

This command will list all files that end with the string *"trimal"* (which are the ones containing our trimmed individual gene alignments) and redirect the output to the *fastaFiles.list* file. To check that your file contains the right information, you can examine the content of the *fastaFiles.list* using the *head* command.

ii) To concatenate the sequences in *fastaFiles.list* into one multi-fasta file, execute the following command (all on one line):

python ~/workshop_materials/phylogenomics/FILES_scripts_and_matrices/concatenate_seqs_phyloPractical.py fastaFiles.list

Executing this command created two files, *concatenation.fasta* and *partition.file,* which are the concatenated matrix of BUSCO genes and a file that describes where one gene ends and another begins, respectively. Examine the contents of these files using the *head* command.

5) Infer putative species tree

• To infer a species tree using the concatenation matrix, we will use the IQ-tree software (http://www.iqtree.org/). To determine the support for bipartitions, we will use ultrafast bootstrap (UFBoot; https://academic.oup.com/mbe/article/35/2/518/4565479) replicates, which is a fast alternative to the classic bootstrapping approach, and internode certainty, which is a tree-based measure of internal branch conflict.

• Tree inference takes too long to be part of the phylogenomics practical. However, it will be good practice to get a sense for what inferring a phylogeny is like from a single gene (in this case BUSCO gene EOG092D3RDW).

Objectives:

i) To infer the evolutionary history of a single gene, execute the following command:

iqtree -s EOG092D3RDW.fa.mafft.trimal -seed 76813459876 -pre EOG092D3RDW -st AA -m LG+G4

where *EOG092D3RDW.fa.mafft.trimal* is the trimmed BUSCO gene alignment, where *-seed* specifies the random seed (rerunning the analysis with the same seed will return identical results and allows others to reproduce your results), *-pre* is the specified file output prefix, *-st AA* specifies that the input matrix is comprised of amino acids, and *-m LG+G4* is the model of amino acid substitutions (the same model was used in the original study by Shen *et al.* 2018). Note, for **.treefile* to be viewed in FigTree, the file must end in *.tree* or *.tre,* which requires you to rename the **.treefile* file.

ii) Examine the topology of the tree in FigTree. What is Brewery_genome most closely related to according to this single gene?

Brewery_genome closest relative:

iii) Now back to conducting multi-locus analyses…results from IQ-Tree analysis of the concatenated matrix has been provided for you in directory *FILES_results_inferring_species_tree.* These results were obtained by executing the following command:

iqtree -s concatenation.fasta -bb 1000 -seed 97163456 -pre concatenation -st AA -m LG+G4 -wbtl

where *concatenation.fasta* is the concatenation matrix of BUSCO genes, where *-bb 1000* will conduct bipartition support using 1000 UFBoot) replicates, where *-seed* specifies the random seed, *-pre* is the specified file output prefix, *-st AA* specifies that the input matrix is comprised of amino acids, *-m LG+G4* is the model of amino acid substitutions (the same model was used in the original study by Shen *et al.* 2018), and *-wbtl* specifies that an additional output file of the 1000 bootstrap replicate trees will be generated (again, if you want to view this file in FigTree, please make sure you rename it such that it ends in *.tree* or *.tre*.

iv) Examine the contents of *FILES_results_inferring_species_tree*, which is in directory *FILES_results*, using the *ls* command. Examine the *concatenation.log* file to gain a sense of what IQ-Tree had done. The most important result files will be the *concatenation.treefile,* which is a newick-formatted tree file (for more on the newick

format, see http://evolution.genetics.washington.edu/phylip/newicktree.html) that contains support for each bipartition and can be viewed in the phylogenetic tree viewing software *FigTree*, and *concatenation.ufboot*, which contains 1000 newick trees where each newick tree represents one UFBoot replicate.

6) What is Brewery_genome?

• Now that you have inferred the phylogeny, it is time to examine the phylogeny for what your newly sequenced genome is.

Objectives:

i) To do so, compare the known yeast phylogeny shown in Figure 2 of the Shen *et al.* 2018 manuscript (you can view the manuscript as a PDF from here: https://jlsteenwyk.github.io/publication_pdfs/Shen_et_al_2018_Cell.pdf) with the newly inferred phylogeny (*concatenation.treefile*) so that you can find what genome matches Brewery_genome. Hint: The easiest way to find taxa on the phylogeny you created and the Shen *et al.* 2018 phylogeny is to use *control+f* to search for taxon names in Figure 2. Be sure to root the phylogeny by clicking on the stem branch of the clade containing *Starmerella* apicola, *Starmerella bombicola,* and *Wickerhamiella versatilis* and clicking "Reroot" from the top panel of options.

ii) What species is the Brewery_genome from? After you have done that, find out some information about this organism by searching with the species' name on the internet. Does the presence of this species in a brewery make sense?
Brewery_genome is:
Does finding Brewery_genome in a brewery make sense, why?

7) Examine bipartition support

• A major issue in phylogenomics is that individual gene trees often disagree with each other on the resolution of specific internal branches (bipartitions). In this step, we will examine bipartition support using UFBoot, which we calculated in the previous step, and internode certainty (https://www.nature.com/articles/nature12130?page=7 and https://academic.oup.com/mbe/article/33/6/1606/2579777). Internode certainty is a method

developed to quantify the degree of support for a given internal branch among a set of phylogenies (i.e., bootstrap replicates or single gene trees). Values above 0.4-0.5 suggest that most individual genes recover the internal branch in question (i.e., there is little or no conflict), while values near 0 suggest that there is substantial conflict (i.e., strong support for one or more conflicting internal branches).

Objectives:

i) Examine the *concatenation.treefile* and bootstrap support. To view bootstrap values in FigTree, select *Node Labels* and then from the Display drop down menu, select *label*. How many internal branches have UFBoot support values lower than 100%? According to UFBoot, how well supported is the placement of Brewery_genome?

ii) To determine internode certainty, we will use RAxML to execute the following command:

raxmlHPC -f i -t concatenation.treefile -z concatenation.ufboot -m GTRCAT -n concatenation

where *-f i* will calculate and display *internode certainty* and *internode certainty all* values onto the reference tree (specified with the *-t* parameter) based on calculations using a set of trees (specified with the *-z* parameter). To view internode certainty values, use FigTree to open the file *RAxML_IC_Score_BranchLabels.concatenation.ic.tree* in directory *FILES_results_internode_certainty* (note that internode certainty values, much like bootstrap values, are shown for each individual branch). View this file using FigTree and determine how many internodes have internode certainty values lower than 1.0? According to internode certainty, how well supported is the placement of Brewery_genome?

7) Conduct single-gene phylogenetic signal analysis between two topologies

• Consider an alternative hypothesis that Brewery_genome is most closely related to *Ogataea polymorpha* rather than to a *Brettanomyces* species. To determine the single-gene phylogenetic signal in support of the two alternative hypotheses, we will calculate the difference in gene-wise

log-likelihood scores (ΔGLS) per gene as originally described in Shen *et al.* 2017, *Nature Ecology and Evolution* (https://www.nature.com/articles/s41559-017-0126). To do so, we will need three sets of files: newick files with the two different input topologies, which can be found in *FILES_GLS_files*, the concatenation matrix, and the partition file. NOTE, be sure to provide the full path to input files.

Objectives:

i) Calculate the gene-wise log-likelihood scores (GLS) using the following commands (each command takes ~8 minutes):

Command 1:

iqtree -s concatenation.fasta -seed 78913467814 -st AA -pre GLS_reference_topology_res -spp partition.file -te FILES_GLS_files/concatenation.topology -wpl

Command 2:

iqtree -s concatenation.fasta -seed 13469781343 -st AA -pre GLS_alternative_topology_res -spp partition.file -te FILES_GLS_files/concatenation.alt.topology -wpl

Unlike previous commands where arguments were explained, it will be good practice to navigate software manuals. Determine what parameters *-seed, -spp, -te,* and *-wpl* do. Fortunately, IQ-Tree is nicely documented and this information can be found here http://www.iqtree.org/doc/Command-Reference

- *-seed*:
- *-spp*:
- *-te*:
- *-wpl*:

ii) Next, create a results summary file that will also calculate ΔGLS. To do so, execute the following command:

paste <(cat partition.file | awk '{print $2}') <(cat GLS_reference_topology_res.partlh | tr " " "\n" | grep "-") <(cat GLS_alternative_topology_res.partlh | tr " " "\n" | grep "-") | awk -v OFS='\t' '{print $0, $2-$3}' | awk -v OFS='\t' '{if ($NF<0) print $0, "t2"; else print $0, "t1"}'

where *paste* is a command that prints the output of a set of commands specified using "<*(*" and "*)*" such that the first set of commands will print the second column of the *partition.file* and the second and third commands will print out the log-likelihood scores for the reference and alternative topologies on a per gene basis. The resulting output from the *paste* command will have three columns with gene names, reference topology log-likelihood score, and the alternative topology log-likelihood score. This output is then redirected to calculate the difference between log-likelihood values and print "*ref*" or "*alt*" to represent greater single gene support for the reference or alternative topology, respectively.

iii) Currently, the command described in step *iii* does not save the output to a file. Modify the code above to redirect the output to a file with a clear name describing what the contents of the file are.

iv) Determine how many genes support the reference topology and alternative topology

Number of genes supporting the reference topology:

Number of genes supporting the alternative topology: