Homology and Clustering exercises

Download the files that will be used

Browse to and download http://jbpease.github.io/crete2016/exercise_2.tar.gz

```
tar -xzf exercise_2.tar.gz
```

This should create a directory called exercise_2. You should be able to do everything in there

Phylogeny and BLAST

First we will look at cary.matK.fasta and see what happens when we blast a sequence into this set and compare it to a tree we have already built. We want to build a database and then blast a sequence from that database into itself. We are going to search with Suessenguthiella_scleranthoides_FN825756. Look at this in the extracted portion of the tree

```
figtree cary.matK.rr.sub.tre
```

Now we will do a blast of this sequence against these sequences and many others.

```
makeblastdb -in cary.matK.fasta -dbtype nucl
```

```
blastn -query cary.test -db cary.matK.fasta -num_alignments 0 -num_descriptions 10
```

How many need to be listed (change the num_descriptions to be higher) before we get Coelanthum_parviflorum_FN825759 in the list? What does this mean for analyses that interpret the order of the hit?

Pulling out sequences for alignment with BLAST (Baited BLAST)

We can use a file that contains all the sequences from GenBank for the plant group Caryophyllalles for this example called cary.renamed.fasta. We will use another file (some bait) as the database to pull out all of the rbcL sequences from cary.renamed.fasta (maybe backwards from what you would think).

```
makeblastdb -in cary.atpB.fasta -dbtype nucl
```

```
blastn -db cary.atpB.fasta -query cary.fasta -evalue 10e-10 -num_threads 2 -max_target_seqs 2
-out cary.fasta.rawblastn -outfmt '6 qseqid qlen sseqid slen frames pident nident length mismatch
gapopen qstart qend sstart send evalue bitscore'
```

Output is `cary.fasta.rawblastn` *in case things don't finish (14 secs for my laptop). We are going to filter the results with some basic filters for length of hit using a provided script*

```
python filter_results_bait.py cary.fasta.rawblastn cary.fasta cary.fasta.table  > atp.results
```

We can now conduct a quick multiple sequence alignment and phylogeny

```
mafft --adjustdirection atp.results > atp.aln
```

```
python fasta2phylip.py atp.aln atp.phy
```

```
raxmlHPC-PTHREADS-AVX2 -T 2 -s atp.phy -n ATP -p 12345 -m GTRCAT
```

If we want to combine the bait with the results we can do

```
cat cary.atpB.fasta atp.results > cary_and_atp.results
```

```
mafft --adjustdirection cary_and_atp.results > cary_and_atp.aln
```

```
python fasta2phylip.py cary_and_atp.aln cary_and_atp.phy
```

```
raxmlHPC-PTHREADS-AVX2 -T 2 -s cary_and_atp.phy -n CARY_AND_ATP -p 12345 -m GTRCAT
```

Creating all by all alignments with BLAST and SWIPE

First, we will look at a file ITS.gbget that is one gene region (ITS) and a bunch of species. I know that these can be aligned together, but results can vary given clustering analyses.

There are a number of ways that we can construct the all-by-all comparisons that are necessary for the MCL analyses. We have already learned some about the tools SWIPE and BLAST and with the help of some scripts we can use those tools to construct the information we need for MCL

First we will conduct the all-by-all comparison

We make the database

```
makeblastdb -in ITS.gbget -parse_seqids -dbtype nucl -out  ITS.gbget.db
```

Now I will conduct the all-by-all using swipe (if you have swipe you can also do this, if not, watch or skip to the blast section below)

```
swipe -d ITS.gbget.db -i ITS.gbget -a 4 -c 50 -p blastn -o ITS.gbget.swipe -m 8 -e 1000000
```

Look at the results in a plain text editor (they have been provided in the folder in case you dont' have skype as the file ITS.gbget.swipe). The results are like

```
Query id, Subject id, % identity, alignment length, mismatches, gap openings, q. start, q. end,
s. start, s. end, e-value, bit score.
```

Now we convert the file to be read by mcl which is seq1 seq2 score

```
python swipe_to_mcl.py ITS.gbget.swipe
```

```
//ITS.gbget.swipe.filtered will be created
```

Now we are going to run all-by-all blast on the same database

```
blastn -db ITS.gbget.db -query ITS.gbget -evalue 0.00001 -num_threads 4 -max_target_seqs 100 -out
ITS.gbget.rawblastn -outfmt '6 qseqid qlen sseqid slen frames pident nident length mismatch
gapopen qstart qend sstart send evalue bitscore'
```

Check out these results in a plain text editor. How do the evalues differ for the same comparisons? Remember we are comparing Smith-Waterman and the BLAST heuristic.

Now we are going convert the file so that it can be run by MCL

```
awk -F '\t' '{if($6>0.7 && $7>200 && $7/$2>0.2) print $1,$3,$15}' ITS.gbget.rawblastn >bl.evalue
```

```
sed -i 's/  / /g' bl.evalue
```

```
awk '{if($3==0.0)print $1,$2,180;else print $1,$2,-log($3)/log(10)}' bl.evalue >bl.evalue-log
```

Remember that we want log e-values. These commands change the file into what we need with log10 e-values.

Constructing clusters with MCL

To run a basic MCL analysis, we just run

```
mcl bl.evalue-log --abc -I 2
```

The -I is the inflation value. If you open the results file in a text editor, you will notice that on each line is a cluster. Then, separated by spaces, there is the name of each sequence in the cluster.

Change the inflation value from 1.1 to 10.1 incrementing by 1. Look at the resulting number of clusters. How does increasing the value change the number of clusters? Do you notice how the runtime and iterations change?

You can extract the sequences so that alignments can be run using the command

```
python write_fasta_files_from_clusters_simple.py ITS.gbget out.bl.evalue-log.I20 .
```

In addition to adjusting the inflation value, we can, on the fly, filter with evalues. Because we did log10 evalues, we are looking at the values in reverse, so large is better. So we can include only relationships (blast results) with greater than or equal to (gq) 30 like this

```
mcl all.evalue-log --abc -I 2 -tf 'gq(30)'
```

Try different values. How does this change the results?

Doing it again with another file!

Now let's look at another file all.unaln. This includes a number of different gene regions that shouldn't go together.

Using the procedures outlined above, go ahead and repeat, at least the all-by-all BLAST and then mcl.

How do different inflation values change the clusters?

The sequence names are prepended with the gene.

Are the clusters reflecting this well?

What is the "best" inflation value, if we are going by the sequence name?

Are there potential problematic sequences?

BLAST them on webBLAST!

Running orthology and paralogy analyses

We do not have time to do more intensive analyses, in the exercises and so you will need to do them on your own. If you want to do orthology and paralogy analyses you can check out this repository and instructions

https://bitbucket.org/yangya/phylogenomic_dataset_construction

This is from this publication

Yang, Y. and S.A. Smith. 2014. Orthology inference in non-model organisms using transcriptomes and low-coverage genomes: improving accuracy and matrix occupancy for phylogenomics. Molecular Biology and Evolution. doi: 10.1093/molbev/msu245

If you want something more general, email me and I can send you more general procedures.

For baited blast analyses, you can use PHLAWD http://phlawd.net/ or send me an email and I can send you more general scripts.