Comparative Genomics 2018
Practical 5: Gene order analysis

Group 11
Fuqi Xu, Milda Valiukonyte, Shuhan Xu

Summary
In this practical we learned about how gene order analysis can help us understand the evolution of genomes. We used ortholog clusters identified in exercise 4 to create the gene order file for each genome. Pseudo-genomes were generated from the gene order files by assign a random sequence to each cluster number in the files. We then ran the Dotter program with the pseudo-genomes and produced a dot plot which enabled us to visualize gene order conservation across the genomes. Finally, we used GRIMM to calculate genomic rearrangement distances between the genomes and used Belvu to construct a phylogenetic tree. The phylogenetic tree produced has the same topology as the consensus tree constructed in the previous practical.

Exercise 1

Ortholog clusters from practical 4 is used for this practical: see attachment *cluster*

Exercise 2

1.
Gene order output for 09.fa.txt: *09_gene_order*
Gene order output for 17.fa.txt: *17_gene_order*
Gene order output for 49.fa.txt: *49_gene_order*
Gene order output for 51.fa.txt: *51_gene_order*

2.
a.
There can be ambiguities in clustering.

For genome 09.fa.txt, no genes will appear in more than one clusters since this genome is the reference genome and is used as the query to BLAST search against other genomes. Hence each cluster contain a unique 09.fa.txt gene.

For 17.fa.txt, 49.fa.txt and 51.fa.txt, there may be genes appearing in more than one cluster since each of these genomes are used as reference database in the BLAST search. For instance, two different genes from 09.fa.txt may have the same best-scoring BLAST hit in 17.fa.txt. As a result, two clusters may have different genes from 09.fa.txt but the same gene from 17.fa.txt.

If a gene appears in more than one cluster, the script will assign to it the cluster number of the first cluster in the list in which the gene appears. Hence, if the gene appears in subsequent clusters, these clusters will not have that gene after the script is run.

b.
This script does not handle forward and reverse strandedness in the gene order list. To implement this functionality in the script, the script can add a positive sign '+' in front of the cluster number if the gene

appears in the forward strand and a negative sign '-' if the gene appears in the reverse strand. This can be done in the last loop of the script:

```
for aGene in geneOrderList:
    if partOfCluster.has_key (aGene):
        if aGene.endswith('rev'):
            print '-' + str(partOfCluster [aGene]),
        else:
            print '+' + str(partOfCluster [aGene]),
```

Downstream scripts can check the sign and deal with the situation accordingly, e.g. generate reverse complement if the sign is negative.

Exercise 3

We wrote a script to perform the tasks in question 3, 4, 5 and 6. This script incorporates rndseq.py

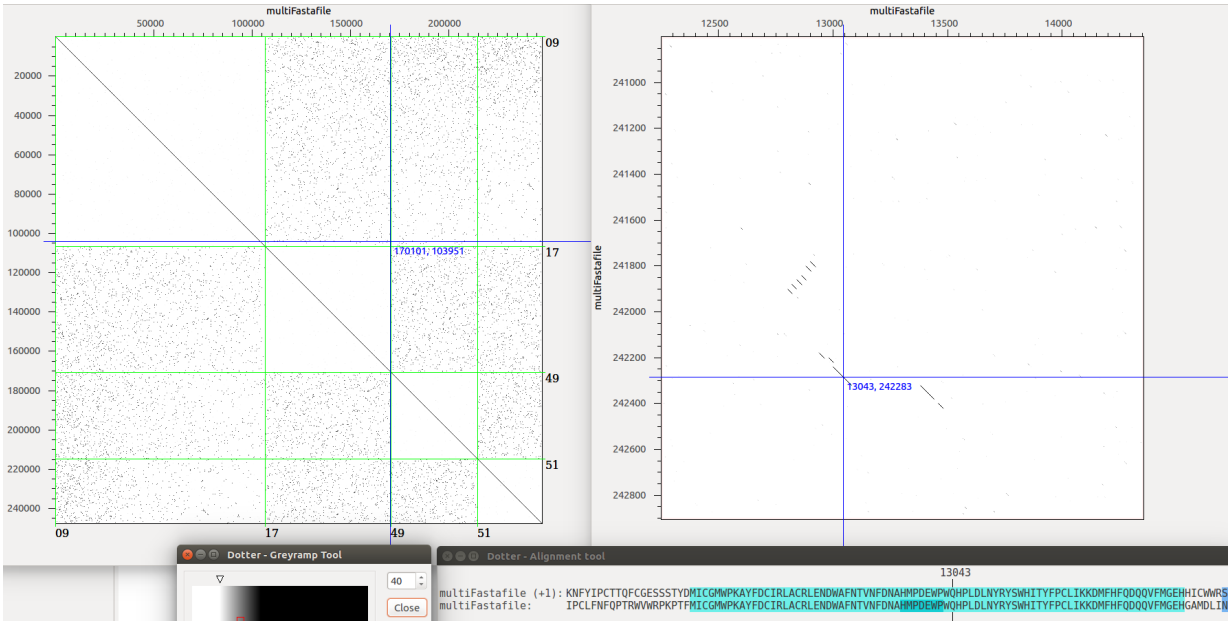
Script for generating pseudo-genomes from gene order files: see attachment *generate_pseudogenomes.py*

Usage: python3 generate_pseudogenomes.py <number of ORFs in largest proteome> <gene order file 1> <gene order file 2> <gene order file 3> …

Description: The script first generates a list of pseudo-genes (from rndseq.py code in the script) and converts it to a dictionary. Then, for every gene order file, it assigns a pseudo gene to each cluster number and concatenates the pseudo-genes into a single long sequence (pseudo-genome). Finally, it combines all the pseudo-genomes into one multi FASTA file called 'multiFastafile' in the working directory.

Multi FASTA file output from script: see attachment *multiFastafile*

6.



In the dot plot, if there is sufficient number of identical matched residues within a window centered in an aligned position, a dot will be generated at that position. If the dots produced a continuous diagonal line, this means that the two genomes have highly similar or identical sequences at that region.

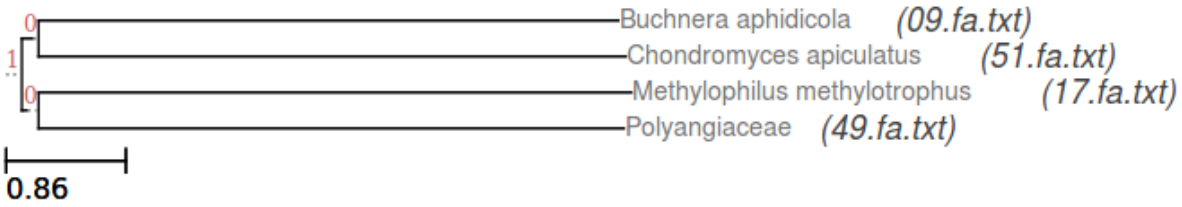
In the right window of the image above, we can see a few diagonal lines. The aligned sequences for the long diagonal line at (13043,242283) is shown in the bottom right window. We can see that there are 80 identical matched residues in this region, suggesting that the gene order of 4 genes (since each pseudo-gene is 20 residues long) are conserved across the two genomes.

Exercise 4

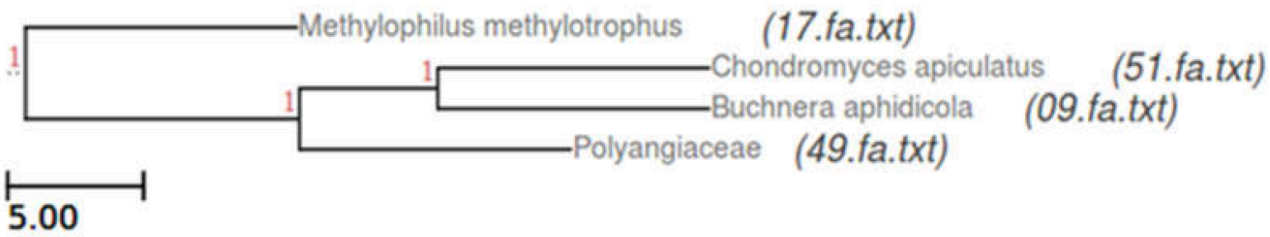
1.
Formatted gene order file for GRIMM: see attachment *grim_genomes.txt*

2.
Distance matrix from GRIMM: see attachment *distance.grim*

3.
Reconstructed Species Tree using distance matrix from GRIMM:



4.
Consensus Tree for 10 orthologs clusters from Practical 4:



Both the GRIMM tree in this practical and the consensus tree in practical 4 share the same tree topology. In both trees, 17.fa.txt and 49.fa.txt shares a node while 51.fa.txt and 09.fa.txt shares another node. Hence the same evolutionary relationship is inferred from gene order and sequence alignment.

If the results didn't agree, one of the possible explanation is that gene order is only locally preserved in closely related species. Our four genomes may be too evolutionary distant for gene order to be conserved. Hence, we cannot confidently calculate evolutionary distance from gene order.

The second explanation is our gene order does not consider strandedness of the genes, which makes the distance calculation from gene order less accurate than it should be.

The last explanation is that we only used ten genes clusters to build the consensus tree in practical 4 while we use the gene order of the whole genome to build the tree in this practical. The ten gene clusters may not be an accurate representation of the evolution of the whole genome.