**Practical 5: Gene order analysis**

Group no. 11

Fuqi Xu, Shuhan Xu, Milda Valiukonytė

In this practical we learned about how gene order analysis can help us understand the evolution of genomes. We used otholog clusters identified in exercise 4 to infer the order they appear in the genomes and analyzed the differences between the genomes by visualizing genes against each other in a dotplot using Dotter program. We later used GRIMM to determine the genomic rearrangement distance between genomes and construct a phylogenetic tree using Belvu. It showed that the biggest similarities are between *E.coli* and *S. curvatus,* and between *S.coelicolor* with *R.xylanophilus.*

**Exercise 1.**

1. Ortholog clusters were found in practical 4.

**Exercise 2.**

1. Used script getGeneOrder.py to produce output files 09\_gene\_order, 17\_gene\_order, 49\_gene\_order, 51\_gene\_order.

1. a. Yes because one gene might be homologous to multiple genes in the reference genome. The script iterates through the list of ortholog clusters and each gene name in the cluster and puts the name of the gene as key in a dictionary and assigns an ID which corresponds to the current line number. However, if the gene is already present in the dictionary, a new value is not assigned.

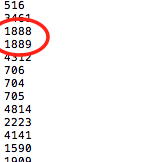
if not partOfCluster.has\_key (aWord):

partOfCluster [aWord] = id

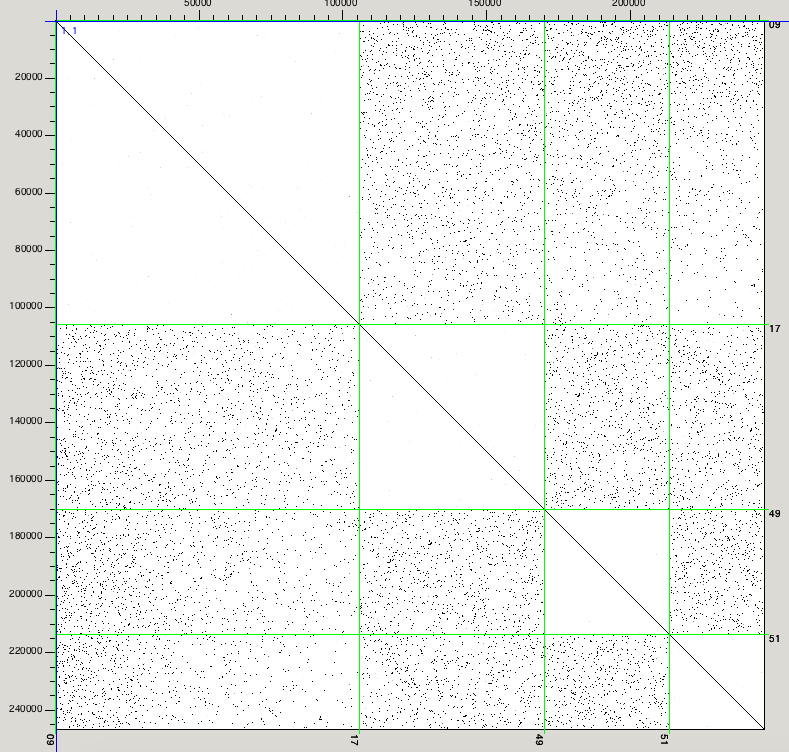
b. No, the script cannot handle forward/reverse strandedness as it assigns an ID to a gene without taking into account whether it is on the forward or reverse strand. A way to deal with this is to assign an ID with a special index, for example, instead of 1 assign 1.1. Then when the gene order is printed out one could see if the gene is in reverse direction in the genome.

**Exercise 3.**

Synteny refers to conservation of stretches of DNA on chromosomes between species. We can see that among our genomes gene order has changed however one can see small stretches of conserved gene order, for example in *S.coelicolor* (17.fa.txt) genome:

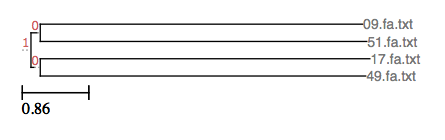


1. Modified the script rndseq.py to make it generate a dictionary with random sequences assigned to a number.
2. Script random\_seq.py assigns a random amino acid sequence to each gene and combines them into a long sequence (output: long\_sequence\_09, long\_sequence\_17, etc.)
3. –
4. Combined sequences to one file called long\_sequences\_all.
5. Dotplot between sequences:



In the dotplot we can see black dots which show similarities in genomes among the sequences. The diagonal represents the sequences aligned to themselves and the green lines represent the distinction between four genomes. From the dotplot we can see that there are many similarities between all of the genomes, mostly reserved to single orthologous genes. However, short syntenic regions of several genes can also be observed as well as short regions of gene duplications, depicted by vertical and horizontal lines.

**Exercise 4.**

1. Used getGeneOrderGrimm.py script to create gene order lists for each of our genomes (output: genomes\_grimm.txt)
2. Calculated distance matrix between genomes using Grimm (output: distance.grimm)
3. Calculated phylogenetic tree based on genomic distances: 
4. The tree constructed using genomic rearrangement distances agrees with the tree built in exercise 3 which used just the sequences. *Escherichia coli* (09.fa.txt) genome seems to be mostly related to *Spiribacter curvatus* (51.fa.txt) while *Streptomyces coelicolor* (17.fa.txt) is mostly similar to *Rubrobacter xylanophilus* (49.fa.txt).