Comparative Genomics 2018

Practical 4: Phylogenomics

Group 9
Martínez Hernández, Marina
Pérez Gómez, Fernando

Summary

In this practical we selected one of the four prokaryotic genomes as a reference genome and we looked for its ortholog proteins in the other three prokaryotic genomes. First, we created proteome databases for each of the genomes and used BLAST to find the best hits of the queries on the reference. After combining the best hits of our reference genome with the other three, we clustered together those proteins that matched the same reference ORF, so that it was possible to find their respective orthologs in the other genomes. Then, we aligned the clustered ortholog proteins and created a metagene out of the alignment. Finally, we also built the phylogenetic tree of some of these alignments and analyzed them.

Key questions to answer

1. Orthology search
2. Metagene approach
3. Consensus reconstruction

Attachments

Activity 1.5.1 and 5.2: practical4_group9_blastResultParser.py
Activity 1.5.2: parsedhits_04_16
Activity 1.6.2: practical4_group9_clustering_names.py
Activity 1.6.2: clusteredhits and clusteredhits_names
Activity 1.7.1: practical4_group9_creatingMultiFasta.py
Activity 3.2.1.3: outtree and outfile

Exercise 1 – Orthology Search

Select one of your prokaryotic genomes as the reference genome. For this prokaryotic reference genome, you will search for the best hit orthologs in the remaining prokaryotic genomes. Use BLAST for this purpose.

04.fa.txt selected as the reference genome.

1. For all of your prokaryotic genomes (including the reference genome), find multi FASTA files with proteins from one of the previous practicals. The multi FASTA file contains all protein sequences inferred from your genomes and will be called as the proteome file.

Done! File names: 04.fa.txt.pfa, 16.fa.txt.pfa, 18.fa.txt.pfa and 49.fa.txt.pfa.

2. Configure BLAST. Copy the configuration file to be able to use the BLAST commands.

Done!

Command:

```
cp /afs/pdc.kth.se/home/a/arnee/.ncbirc ~/
```

3. Create a BLAST database for your proteomes. In the previous practicals, you used makeblastdb to create a database for blastn. Repeat the same procedure for all of your selected proteomes individually (amino acid type).

Done!

Commands:

```
makeblastdb -in 04.fa.txt.pfa -dbtype prot -out db04prot
makeblastdb -in 16.fa.txt.pfa -dbtype prot -out db16prot
makeblastdb -in 18.fa.txt.pfa -dbtype prot -out db18prot
makeblastdb -in 49.fa.txt.pfa -dbtype prot -out db49prot
```

4. Perform a blastpsearch against your reference proteome. (Added -m 5 parameter returns an XML output which is easy to parse in biopython).

4.1. Repeat this process for all of your query proteomes and create the output XML files.

Done!

Commands:

```
blastp -outfmt 5 -query 04.fa.txt.pfa -db db16prot -out blastp16XML
blastp -outfmt 5 -query 04.fa.txt.pfa -db db18prot -out blastp18XML
blastp -outfmt 5 -query 04.fa.txt.pfa -db db49prot -out blastp49XML
```

5. Modify your script from Practical 3 to parse the XML output. The input will be like: <xml output> <tag for reference genome> <tag for target genome>

Done!

Command:

```
python3 blastResultParser.py blastp16XML 04.fa.txt.pfa 16.fa.txt.pfa > parsedhits_04_16
python3 blastResultParser.py blastp18XML 04.fa.txt.pfa 18.fa.txt.pfa > parsedhits_04_18
python3 blastResultParser.py blastp49XML 04.fa.txt.pfa 49.fa.txt.pfa > parsedhits_04_49
```

5.1. Reference and target genome tag parameters are used to assign the genome name for the query and target sequences

<tag for reference genome> <query protein id in reference genome>
<tag for target genome> <best hit protein id in target genome>

Done! Phyton code name: **practical4_group9_blastResultParser.py**

5.2. The output for the script should be in one line with the following four columns:

human proteinI chicken proteinXV
human proteinI mouse proteinXX
human proteinII chicken proteinIX

Done! The output can be seen in parsedhits_04_16, parsedhits_04_18 and parsedhits_04_49 files, when the different genomes are used against the reference genome selected. Example of output: **parsedhits_04_16** file (attached in this practical). In this example, 16fa.txt.pfa proteome was searched against 04fa.txt.pfa (using blastp16XML, which is the bastpsearch done during exercise 4).

Example of one of the lines in the output file:

```
04.fa.txt_orf00643:LLGEPWSRELEAELSELNDVIEAERELR
18.fa.txt_orf01805:LLGRPWANQFEAFIHAPGEVVEHATEFE
```

6. Combine the best hits into one cluster file

6.1. Use the output of your BLAST parser as the input. Each query can have the best hits in different species.

Done!

6.2. Group them in one line as in the example below, in which each line represents a cluster of orthologs:

human_proteinI chicken_proteinXV mouse_proteinXX …

Script for doing so is called **practical4_group9_clustering_names**, and the resulting clustered files are called **clusteredhits** and **clusteredhits_names.** In the first file (**clusteredhits**), there is a compilation of each hit given for an orf in the reference genome (04.fa.txt.pfa) against all genomes, including the fragments of sequences that matched, such as:

04genome_orfID 04genome_sequence 16genome_orfID 16genome_sequence
04genome_orfID 04genome_sequence 18genome_orfID 18genome_sequence
04genome_orfID 04genome_sequence 49genome_orfID 49genome_sequence

Here the reference genome ID is repeated for each of the 3 searches as the fragment of sequence that matched differs in each of them.

On the other hand, in the file **clusteredhits_names** only the orfs' IDs from the reference genome and the remaining 3 appear such as:

04genome_orfID 16genome_orfID 18genome_orfID 49genome_orfID

Here, on the other hand, as we only refer to the ID from the different orfs, there is no need to further repeat the ID from the reference genome (specific orf in 04.fa.txt.pfa) as it is the same (same orf, thus same ID name) for all of them. With this last file containing the names, we can then create a multi FASTA file containing the ID of all prokaryotic genomes with their corresponding proteomic sequences extracted from the original files (04.fa.txt.pfa, 16.fa.txt.pfa, 18.fa.txt.pfa, 49.fa.txt.pfa).

7. Select 10 different ortholog clusters such that all of your prokaryotic genomes are included and acquire the corresponding protein sequences from the multi FASTA files.

7.1. The result should be a number of multi FASTA files, one for each ortholog cluster

Done! Files named Xcluster, being X between 1 and 10. The python code for creating these multi FASTA files is called **practical4_group9_creatingMultiFasta.py**. It uses as input the clusteredhits_names output file that was obtained in exercise 6, which contains the IDs of one cluster per line.

7.2. The multi FASTA file should include all sequences appearing in the cluster of orthologs (including the query sequence).

Done!

Exercise 2 – Metagene Approach

1. Make a multiple alignment of each of your multi FASTA cluster files in order to have one alignment for each cluster of orthologous genes.

Done! With the command kalign <inputfile> <outputfile> we are able to obtain the multiple alignment of all the sequences contained in each of the 10 clusters.

Command:

```
kalign 1cluster aligned01cluster
kalign 2cluster aligned02cluster
kalign 3cluster aligned03cluster
kalign 4cluster aligned04cluster
kalign 5cluster aligned05cluster
kalign 6cluster aligned06cluster
kalign 7cluster aligned07cluster
kalign 8cluster aligned08cluster
kalign 9cluster aligned09cluster
kalign 10cluster aligned10cluster
```

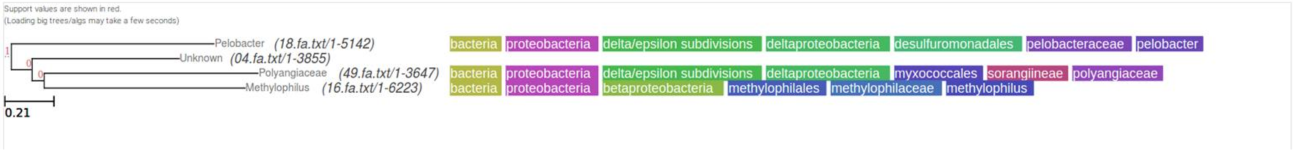
1.1. Concatenate the alignments into a single, long metagene. The end result should be a single FASTA file with a sequence for each bacterial genome consisting of the aligned genes from each genome.

Done! The metagene is built adding each of the clusters' sequences one after the other (genome 4 sequence of cluster 1 is followed by genome 4 sequence of cluster 2, etc). This file is saved as metageneclusters.

1.2. Perform the tree reconstruction using Belvu. What does the tree look like?

Tree done with default parameters (neighbor joining with Scoredist distance correction).

```
(18.fa.txt/1-5142:0.881,
(04.fa.txt/1-3855:0.640,
(49.fa.txt/1-3647:0.930,
16.fa.txt/1-6223:0.874)
0:0.048)
0:0.086);
```


Support values are shown in red.
(Loading big trees/algs may take a few seconds)
1
0
0
Pelobacter (18.fa.txt/1-5142)
Unknown (04.fa.txt/1-3855)
Polyangiaceae (49.fa.txt/1-3647)
Methylophilus (16.fa.txt/1-6223)
0.21
bacteria proteobacteria delta/epsilon subdivisions deltaproteobacteria desulfuromonadales pelobacteraceae pelobacter
bacteria proteobacteria delta/epsilon subdivisions deltaproteobacteria myxococcales sorangiineae polyangiaceae
bacteria proteobacteria betaproteobacteria methylophilales methylophilaceae methylophilus

Commnand:

```
belvu -T n -o tree metageneclusters
```

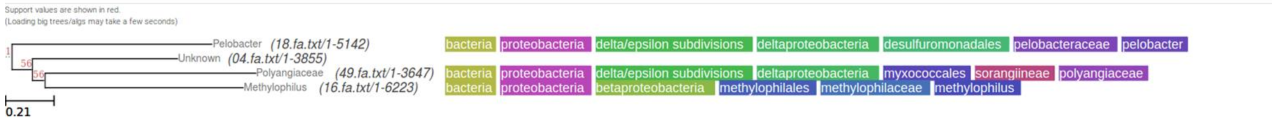
1.2. Perform sequence bootstrapping on the metagene. Evaluate the quality of the reconstruction.

Command:

```
belvu -b 100 -o tree metageneclusters
```

```
(18.fa.txt/1-5142:0.881,
(04.fa.txt/1-3855:0.640,
(49.fa.txt/1-3647:0.930,
16.fa.txt/1-6223:0.874)
56:0.048)
56:0.086);
```

Comparing this metagene tree with each of the individual trees obtained below (look at exercise 5), we can see that some of the individual trees, such as the one obtained from aligned10cluster, are very similar to the metagene tree. This might occur because this specific cluster is the one that best gathers the general features represented by the metagene and this results in a tree with similar patterns and relationships between the genomes. On the other hand, other ortholog


Support values are shown in red.
(Loading big trees/algs may take a few seconds)
1
56
56
Pelobacter (18.fa.txt/1-5142)
Unknown (04.fa.txt/1-3855)
Polyangiaceae (49.fa.txt/1-3647)
Methylophilus (16.fa.txt/1-6223)
0.21
bacteria proteobacteria delta/epsilon subdivisions deltaproteobacteria desulfuromonadales pelobacteraceae pelobacter
bacteria proteobacteria delta/epsilon subdivisions deltaproteobacteria myxococcales sorangiineae polyangiaceae
bacteria proteobacteria betaproteobacteria methylophilales methylophilaceae methylophilus

clusters' trees are not that similar to the metagene tree and reveal different tree patterns between the genomes, because of the genes that are contained in these clusters.

Exercise 3 - Consensus Reconstruction

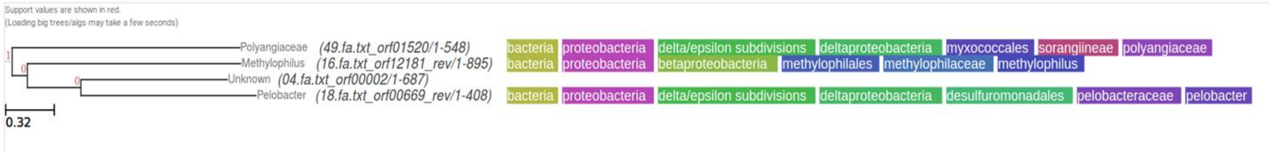
1. Perform a tree reconstruction using Belvu for each individual alignment.

Command:
```
belvu -o tree aligned01cluster > tree01newick and so on with the others…
```

1.1. Obtain 10 different trees, one for each ortholog cluster, in Newick format.

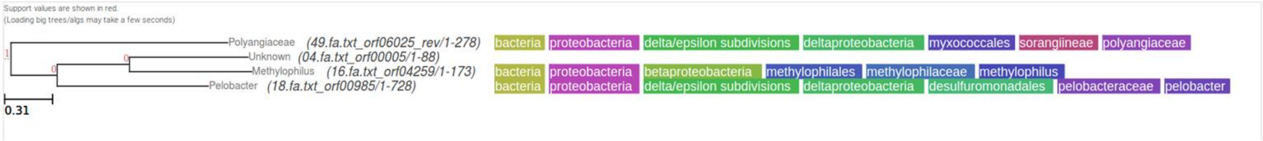
aligned01cluster:

```
(49.fa.txt_orf01520/1-548:1.500,
(16.fa.txt_orf12181_rev/1-895:1.406,
(04.fa.txt_orf00002/1-687:0.965,
18.fa.txt_orf00669_rev/1-408:1.154)
0:0.346)
0:0.094);
```


Support values are shown in red.
(Loading big trees/algs may take a few seconds)
1
0
0
Polyangiaceae (49.fa.txt_orf01520/1-548)
Methylophilus (16.fa.txt_orf12181_rev/1-895)
Unknown (04.fa.txt_orf00002/1-687)
Pelobacter (18.fa.txt_orf00669_rev/1-408)
0.32
bacteria proteobacteria delta/epsilon subdivisions deltaproteobacteria myxococcales sorangiineae polyangiaceae
bacteria proteobacteria betaproteobacteria methylophilales methylophilaceae methylophilus
bacteria proteobacteria delta/epsilon subdivisions deltaproteobacteria desulfuromonadales pelobacteraceae pelobacter

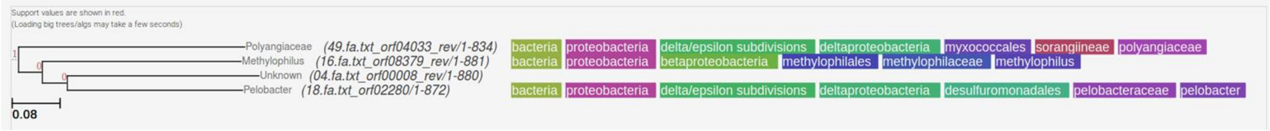
aligned02cluster:

```
(49.fa.txt_orf06025_rev/1-278:1.418,
((04.fa.txt_orf00005/1-88:0.776,
16.fa.txt_orf04259/1-173:0.800)
0:0.467,
18.fa.txt_orf00985/1-728:0.989)
0:0.296);
```


Support values are shown in red.
(Loading big trees/algs may take a few seconds)
1
0
0
Polyangiaceae (49.fa.txt_orf06025_rev/1-278)
Unknown (04.fa.txt_orf00005/1-88)
Methylophilus (16.fa.txt_orf04259/1-173)
Pelobacter (18.fa.txt_orf00985/1-728)
0.31
bacteria proteobacteria delta/epsilon subdivisions deltaproteobacteria myxococcales sorangiineae polyangiaceae
bacteria proteobacteria betaproteobacteria methylophilales methylophilaceae methylophilus
bacteria proteobacteria delta/epsilon subdivisions deltaproteobacteria desulfuromonadales pelobacteraceae pelobacter

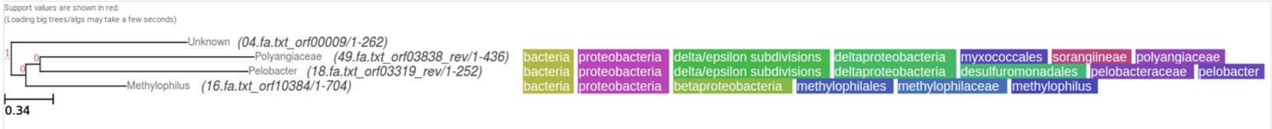
aligned03cluster:

```
(49.fa.txt_orf04033_rev/1-834:0.386,
(16.fa.txt_orf08379_rev/1-881:0.339,
(04.fa.txt_orf00008_rev/1-880:0.327,
18.fa.txt_orf02280/1-872:0.299)
0:0.041)
0:0.039);
```


Support values are shown in red.
(Loading big trees/algs may take a few seconds)
1
0
0
Polyangiaceae (49.fa.txt_orf04033_rev/1-834)
Methylophilus (16.fa.txt_orf08379_rev/1-881)
Unknown (04.fa.txt_orf00008_rev/1-880)
Pelobacter (18.fa.txt_orf02280/1-872)
0.08
bacteria proteobacteria delta/epsilon subdivisions deltaproteobacteria myxococcales sorangiineae polyangiaceae
bacteria proteobacteria betaproteobacteria methylophilales methylophilaceae methylophilus
bacteria proteobacteria delta/epsilon subdivisions deltaproteobacteria desulfuromonadales pelobacteraceae pelobacter

aligned04cluster:

```
(04.fa.txt_orf00009/1-262:1.250,
((49.fa.txt_orf03838_rev/1-436:1.522,
18.fa.txt_orf03319_rev/1-252:1.478)
0:0.094,
16.fa.txt_orf10384/1-704:0.713)
0:0.097);
```

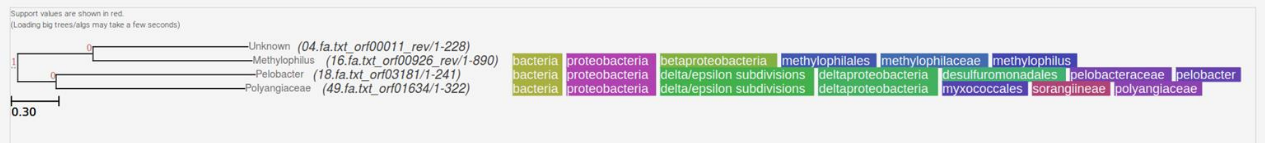


Support values are shown in red.
(Loading big trees/algs may take a few seconds)

1
0
0
Unknown (04.fa.txt_orf00009/1-262)
Polyangiaceae (49.fa.txt_orf03838_rev/1-436) bacteria proteobacteria delta/epsilon subdivisions deltaproteobacteria myxococcales sorangiineae polyangiaceae
Pelobacter (18.fa.txt_orf03319_rev/1-252) bacteria proteobacteria delta/epsilon subdivisions deltaproteobacteria desulfuromonadales pelobacteraceae pelobacter
Methylophilus (16.fa.txt_orf10384/1-704) bacteria proteobacteria betaproteobacteria methylophilales methylophilaceae methylophilus
0.34

aligned05cluster:

```
((04.fa.txt_orf00011_rev/1-228:0.987,
16.fa.txt_orf00926_rev/1-890:1.013)
0:0.474,
(18.fa.txt_orf03181/1-241:1.268,
49.fa.txt_orf01634/1-322:1.201)
0:0.239);
```

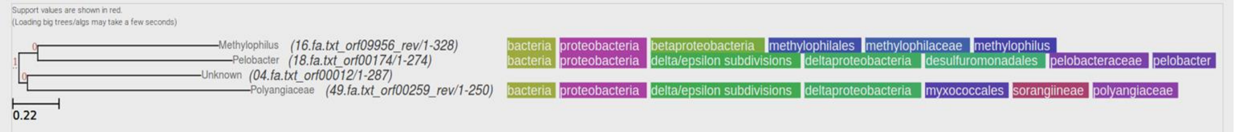


Support values are shown in red.
(Loading big trees/algs may take a few seconds)

1
0
0
Unknown (04.fa.txt_orf00011_rev/1-228)
Methylophilus (16.fa.txt_orf00926_rev/1-890) bacteria proteobacteria betaproteobacteria methylophilales methylophilaceae methylophilus
Pelobacter (18.fa.txt_orf03181/1-241) bacteria proteobacteria delta/epsilon subdivisions deltaproteobacteria desulfuromonadales pelobacteraceae pelobacter
Polyangiaceae (49.fa.txt_orf01634/1-322) bacteria proteobacteria delta/epsilon subdivisions deltaproteobacteria myxococcales sorangiineae polyangiaceae
0.30

aligned06cluster:

```
((16.fa.txt_orf09956_rev/1-328:0.877,
18.fa.txt_orf00174/1-274:0.944)
0:0.087,
(04.fa.txt_orf00012/1-287:0.841,
49.fa.txt_orf00259_rev/1-250:1.081)
0:0.037);
```

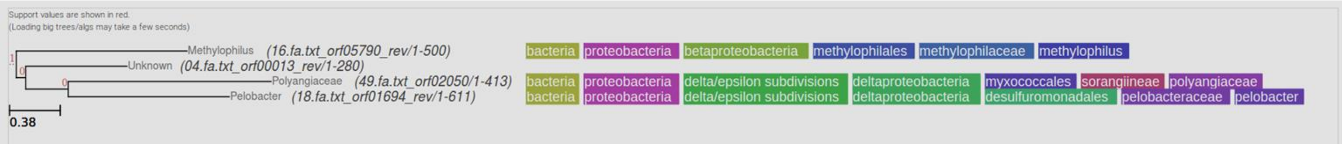


Support values are shown in red.
(Loading big trees/algs may take a few seconds)

1
0
0
Methylophilus (16.fa.txt_orf09956_rev/1-328) bacteria proteobacteria betaproteobacteria methylophilales methylophilaceae methylophilus
Pelobacter (18.fa.txt_orf00174/1-274) bacteria proteobacteria delta/epsilon subdivisions deltaproteobacteria desulfuromonadales pelobacteraceae pelobacter
Unknown (04.fa.txt_orf00012/1-287)
Polyangiaceae (49.fa.txt_orf00259_rev/1-250) bacteria proteobacteria delta/epsilon subdivisions deltaproteobacteria myxococcales sorangiineae polyangiaceae
0.22

aligned07cluster:

```
(16.fa.txt_orf05790_rev/1-500:1.272,
(04.fa.txt_orf00013_rev/1-280:0.754,
(49.fa.txt_orf02050/1-413:1.511,
18.fa.txt_orf01694_rev/1-611:1.200)
0:0.309)
0:0.063);
```



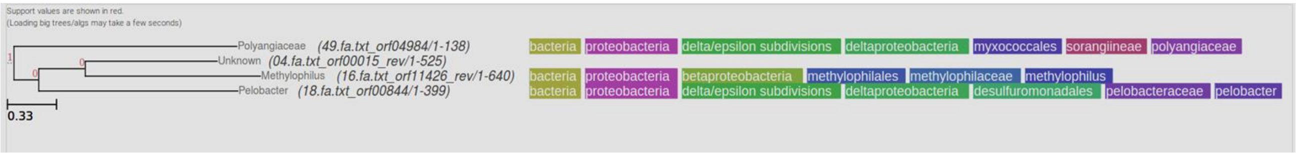
Support values are shown in red.
(Loading big trees/algs may take a few seconds)

1
0
0
Methylophilus (16.fa.txt_orf05790_rev/1-500) bacteria proteobacteria betaproteobacteria methylophilales methylophilaceae methylophilus
Unknown (04.fa.txt_orf00013_rev/1-280)
Polyangiaceae (49.fa.txt_orf02050/1-413) bacteria proteobacteria delta/epsilon subdivisions deltaproteobacteria myxococcales sorangiineae polyangiaceae
Pelobacter (18.fa.txt_orf01694_rev/1-611) bacteria proteobacteria delta/epsilon subdivisions deltaproteobacteria desulfuromonadales pelobacteraceae pelobacter
0.38

aligned08cluster:

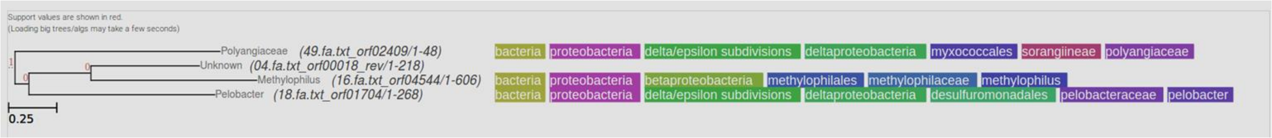
```
(49.fa.txt_orf04984/1-138:1.500,
((04.fa.txt_orf00015_rev/1-525:0.887,
16.fa.txt_orf11426_rev/1-640:1.178)
0:0.307,
18.fa.txt_orf00844/1-399:1.340)
```

0:0.160);


Support values are shown in red.
(Loading big trees/algs may take a few seconds)
1
0
0
Polyangiaceae (49.fa.txt_orf04984/1-138)
Unknown (04.fa.txt_orf00015_rev/1-525)
Methylophilus (16.fa.txt_orf11426_rev/1-640)
Pelobacter (18.fa.txt_orf00844/1-399)
0.33
bacteria proteobacteria delta/epsilon subdivisions deltaproteobacteria myxococcales sorangiineae polyangiaceae
bacteria proteobacteria betaproteobacteria methylophilales methylophilaceae methylophilus
bacteria proteobacteria delta/epsilon subdivisions deltaproteobacteria desulfuromonadales pelobacteraceae pelobacter

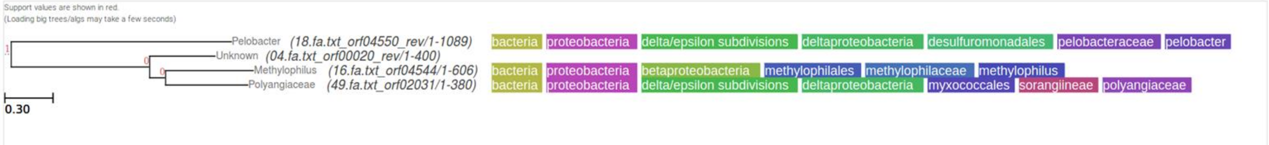
aligned09cluster:

(49.fa.txt_orf02409/1-48:1.075,
((04.fa.txt_orf00018_rev/1-218:0.569,
16.fa.txt_orf04544/1-606:0.870)
0:0.320,
18.fa.txt_orf01704/1-268:0.975)
0:0.068);


Support values are shown in red.
(Loading big trees/algs may take a few seconds)
1
0
0
Polyangiaceae (49.fa.txt_orf02409/1-48)
Unknown (04.fa.txt_orf00018_rev/1-218)
Methylophilus (16.fa.txt_orf04544/1-606)
Pelobacter (18.fa.txt_orf01704/1-268)
0.25
bacteria proteobacteria delta/epsilon subdivisions deltaproteobacteria myxococcales sorangiineae polyangiaceae
bacteria proteobacteria betaproteobacteria methylophilales methylophilaceae methylophilus
bacteria proteobacteria delta/epsilon subdivisions deltaproteobacteria desulfuromonadales pelobacteraceae pelobacter

aligned10cluster:

(18.fa.txt_orf04550_rev/1-1089:1.371,
(04.fa.txt_orf00020_rev/1-400:0.407,
(16.fa.txt_orf04544/1-606:0.546,
49.fa.txt_orf02031/1-380:0.512)
0:0.093)
0:0.856);


Support values are shown in red.
(Loading big trees/algs may take a few seconds)
1
0
0
Pelobacter (18.fa.txt_orf04550_rev/1-1089)
Unknown (04.fa.txt_orf00020_rev/1-400)
Methylophilus (16.fa.txt_orf04544/1-606)
Polyangiaceae (49.fa.txt_orf02031/1-380)
0.30
bacteria proteobacteria delta/epsilon subdivisions deltaproteobacteria desulfuromonadales pelobacteraceae pelobacter
bacteria proteobacteria betaproteobacteria methylophilales methylophilaceae methylophilus
bacteria proteobacteria delta/epsilon subdivisions deltaproteobacteria myxococcales sorangiineae polyangiaceae

1.1. How precise are those trees? Discuss.
1.2. Point a few specific genes or classes of the genes that cause disagreement.
1.3. Discuss the reasons of this disagreement.

Answer to 1.2, 1.3 and 1.4 together:
The precision depends on each of the trees. We obtain different trees depending on which cluster we are analyzing, as we can see in the ten trees above. For example, in the tree obtained from cluster 9, the genomes 04 and 16 are clustered together, whereas in the tree from cluster 10, the closest cluster happens between genomes 16 and 49. This makes sense, since in each cluster we are looking at different parts of the genome (therefore, when we compare different genes from different genomes, it is normal that the genomes will be clustered differently depending on the relationship that exists between the genes in that specific cluster), however, this also turns in some disagreement when comparing it to the results obtained with the metagenome. For example, looking at the metagene tree, genomes 18 and 49 are the most distant ones, but if we look at the tree from cluster 7 we can see that this does not apply. On the other hand, looking at the tree from cluster 10 we can see how the tree is very similar to that of the metagene, agreeing in this case.

2. Construct a consensus tree from the gene trees using Phylip example.
For combining the tree, it is important that you use the species names as protein identifier in each tree.

2.1. Use Phylip consense to construct a consensus tree;

2.1.1. Put the tree files together into a file containing a list of the trees. Phylip should be preinstalled

Done!

Command:

```
cat tree01newick tree02newick tree03newick tree04newick tree05newick tree06newick tree07newick tree08newick tree09newick tree10newick > intree
phylip consense
```

2.1.2. Name this file intree

Done!

2.1.3. Phylip will read the file named intree in the present directory, ask you a few questions and then give you the consensus tree. Describe the process.

The input is specified with the name *intree* so that the command *phylip consense* directly finds it as the input for the construction of the consensus tree. The program phylic, by means of the specified command, constructs consensus tree by the majority-rule consensus tree method. The trees are not treated as rooted and when the program finishes two files are got: **outfile**, which contains indications of progress of run, as well as some of the settings specified for the run and **outtree**, which gives us the final consensus tree obtained.