Practical-3 : Phylogenetic Reconstruction

Stefanie Friedrich and Siddharth Tomar

May 12, 2017

Summary

-

Sequences

Group 11

- 05.fa.txt
- 11.fa.txt
- 15.fa.txt
- 19.fa.txt
- 34.fa.txt

DNA tree reconstruction

Task 1

Format a blast database for your genomes , so you can search locally.

(a) What do the parameters mean?

```
makeblastdb -in <inputfile.fa> -dbtype nucl
- makeblastdb -> Make blast database for searching
- in -> Input file
- dbtype -> Type of database (nucleotide, protein)
```

Task 3

Use BLAST to query the database for the 16S rRNA file. Find the best hit in each genome as “actual” 16S rRNA and gather them as entries in a fasta file. Extract 16S sequence from BLAST results that you run against whole genome.

(a) What do the parameters mean?
Additional parameters required: max_target_seqs 1 and -max_hsps 1 (since we want to limit the number of hits to the best one. The given parser can accept only one record).

(b) Where do you find and what you need in the output ?
We find the output where we have defined the location in command line argument/parameter -out <Location/Name>. The output is in XML format, with alignment against best hit and statistical annotation.

Parsing BLAST output

Option A - Script:

```
from Bio.Blast import NCBIXML

#Define file names which are to be parsed
filename = ["05_bn","11_bn","15_bn","19_bn","34_bn"]

#Driving loop
for i in filename:
  bout = open(i)
  b_records = NCBIXML.parse(bout) #Main parsing function
  outfile = open("out_%s.fasta" % i,'w') #Each parsed sequence is stored in a new file
  for b_record in b_records:
    for alig in b_record.alignments:
      for hsp in alig.hsps:
        hitseq = hsp.sbjct.replace('-','') #Removing the gaps from the sequence
        print (">"+i+"\n"+hitseq)
        outfile.write (">"+i+"\n"+hitseq)
        outfile.close()
```

Align your sequences

Task 4

You can use KALIGN on the resulting sequence file to make a multiple alignment of the homologs identified in the previous step.

(a) What parameters for gap penalties exist, and would any of them make sense to apply?
According to KALIGN manual[1] three gap penalties are employed (affine gap penalties) :
Gap Open (-s)
The penalty for opening/closing a gap. Half the value will be subtracted from the alignment score when opening, and half when closing a gap.
Default value is: 11.0
Gap Extension (-e)
Penalty for extending a gap
Default value is: 0.85
Terminal Gap (-t)
Penalty to extend gaps from the N/C terminal of protein or 5'/3' terminal of nucleotide sequences
Default value is: 0.45

Since we align nucleotide sequences of one gene it does not make much sense to apply gap penalties. This also in respect to domains within a gene.

Perform tree reconstruction

Task 5

What are the advantages of the distance based and the parsimony method? What influenced your choice?

Distance based methods belong to phenetic methods and measure pair-wise differences among multiple aligned sequences resulting in a distance matrix. The distance can be calculated with different algorithms like Euclidean or Jaccard. Tree composing methods like Neighbour-Joining (NJ) or Minimal-Evolution (ME) build the tree from the distance matrix. (ref)
Whereas parsimony methods and maximum likelihood belong to cladistic methods and especially to character based methods. No distance will be calculated but evolution based on different models (like Jukes & Cantor, Kimura) considered. (ref)

Task 6

(a) Which distance correction method does Belvu use?
Belvu uses Scoredist distance estimator which “uses a logarithmic correction of observed divergence based on the alignment score according to the BLOSUM62 score matrix”[2].

(b) How would you request a different distance correction.
Different distance correction methods can be used by defining passing the command line argument -T.

```
-T <method> Tree options:
               b -> Use Scoredist distance correction (default)
               j -> Use Jukes-Cantor distance correction
               k -> Use Kimura distance correction
               s -> Use Storm & Sonnhammer distance correction
               r -> Use uncorrected distances
```

Task 7

The tree in newick format and newick viewer output:

```
(
(
(
11_bn/Geobacter_sulfurreducens:0.122,
19_bn/Thermodesulfovibrio_yellowstonii:0.163)
0:0.019,
15_bn/Pseudomonas_aeruginosa:0.141)
0:0.037,
05_bn/Chlamydia_trachomatis:0.188)
;
```

Refer to Figure 1 for tree.

Sequence bootstrapping

Task 8

Explain what bootstrapping is !
Bootstrapping is an analysis method to calculate confidence measurers (support values) for each node The method slices vectors out of the distance matrix and tests how often the bootstrap tree is mirrored in the tree[3].

Bootstrap values are between 0 and 100 where 100 stands for the highest confidence. Usually, only trees with bootstrap value above 75 are seen as confident.

Task 9

Construct a consensus tree with bootstrap support values from your alignment.

(a) What does N mean?
N here means number[integer] of bootstrap instances.

(b) What N do you choose and what consequences does that choice have?
N should not be smaller than 100 if possible; 1,000 is optimal but can result in high computational load.

Sanity check

Tree : Refer to figure 2 for tree.
Script:

```python
from Bio import SeqIO

#Variables
id = [] #RecordID
sequence = [] #Array to store sequence, should have used dictonary

#File Input
for record in SeqIO.parse("besthits.fa", "fasta"):
  id.append(record.id)
  sequence.append(record.seq.tostring())

#Repeat sequence by appeanding a copy
append_seq = 1

#Sequence shuffle
seq_1 = sequence[0] #define the pairs where sequence will be shuffled
seq_2 = sequence[1] #define the pairs where sequence will be shuffled
A_1, A_2 = seq_1[:len(seq_1)//2], seq_1[len(seq_1)//2:]
B_1, B_2 = seq_2[:len(seq_2)//2], seq_2[len(seq_2)//2:]
sequence[0] = B_1 + A_2
sequence[1] = A_1 + B_2

if (append_seq == 1):
  sequence[0] = sequence[0] + sequence[0]
  sequence[1] = sequence[1] + sequence[1]
  sequence[2] = sequence[2] + sequence[2]
  sequence[3] = sequence[3] + sequence[3]

#File Output
outfile = open('sanityhits.fa', 'w')
for i in range(0,4):
  outfile.write (">"+id[i]+"\n")
  outfile.write (sequence[i]+"\n")
outfile.close()
```

References

[1] T. Lassmann and E. L. Sonnhammer. Kalign–an accurate and fast multiple sequence alignment algorithm. *BMC Bioinformatics*, 6:298, Dec 2005.

[2] E. L. Sonnhammer and V. Hollich. Scoredist: a simple and robust protein sequence distance estimator. *BMC Bioinformatics*, 6:108, Apr 2005.

[3] B. Efron, E. Halloran, and S. Holmes. Bootstrap confidence levels for phylogenetic trees. *Proc. Natl. Acad. Sci. U.S.A.*, 93(23):13429–13434, Nov 1996.

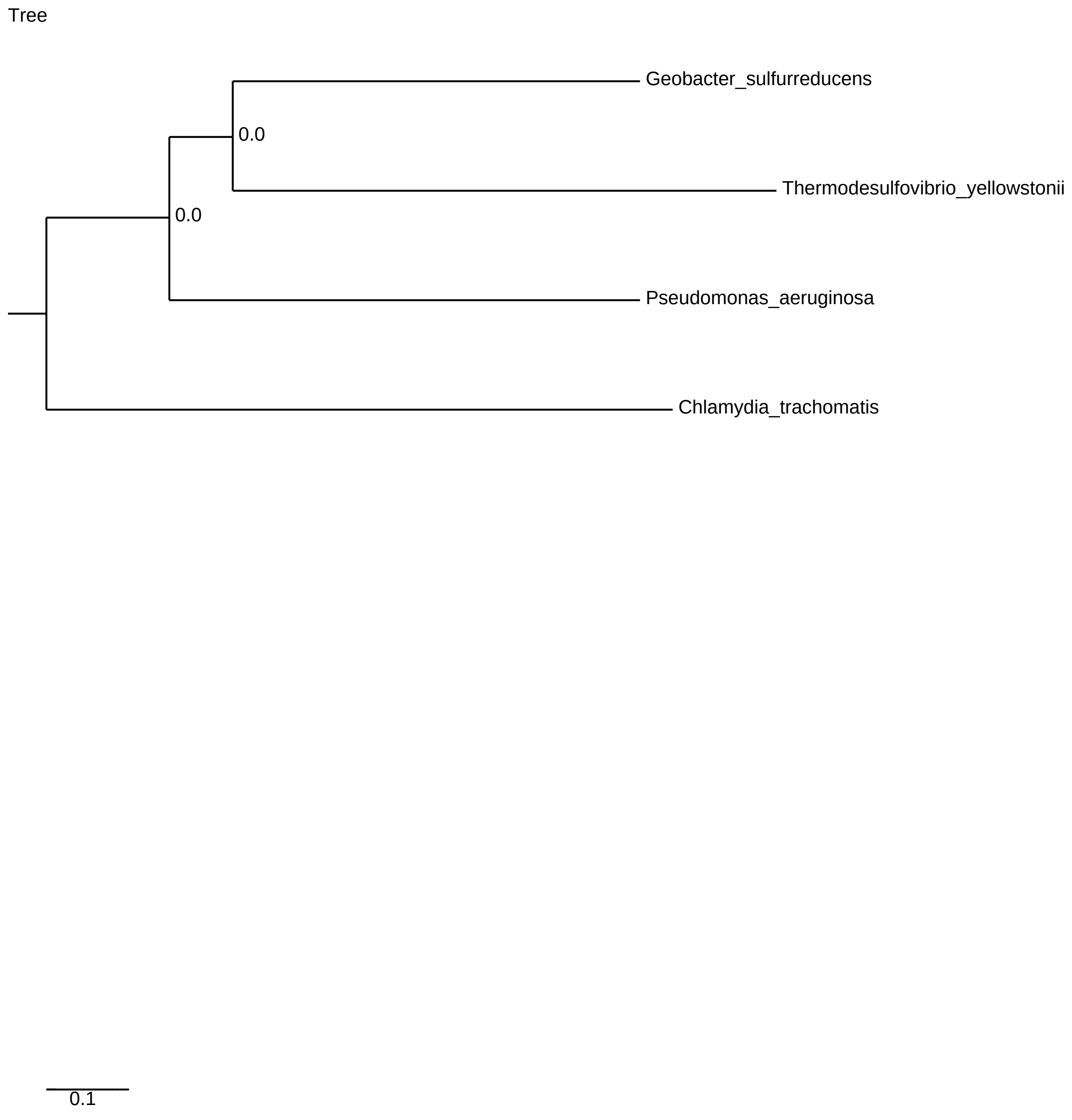

Figure 1: Tree

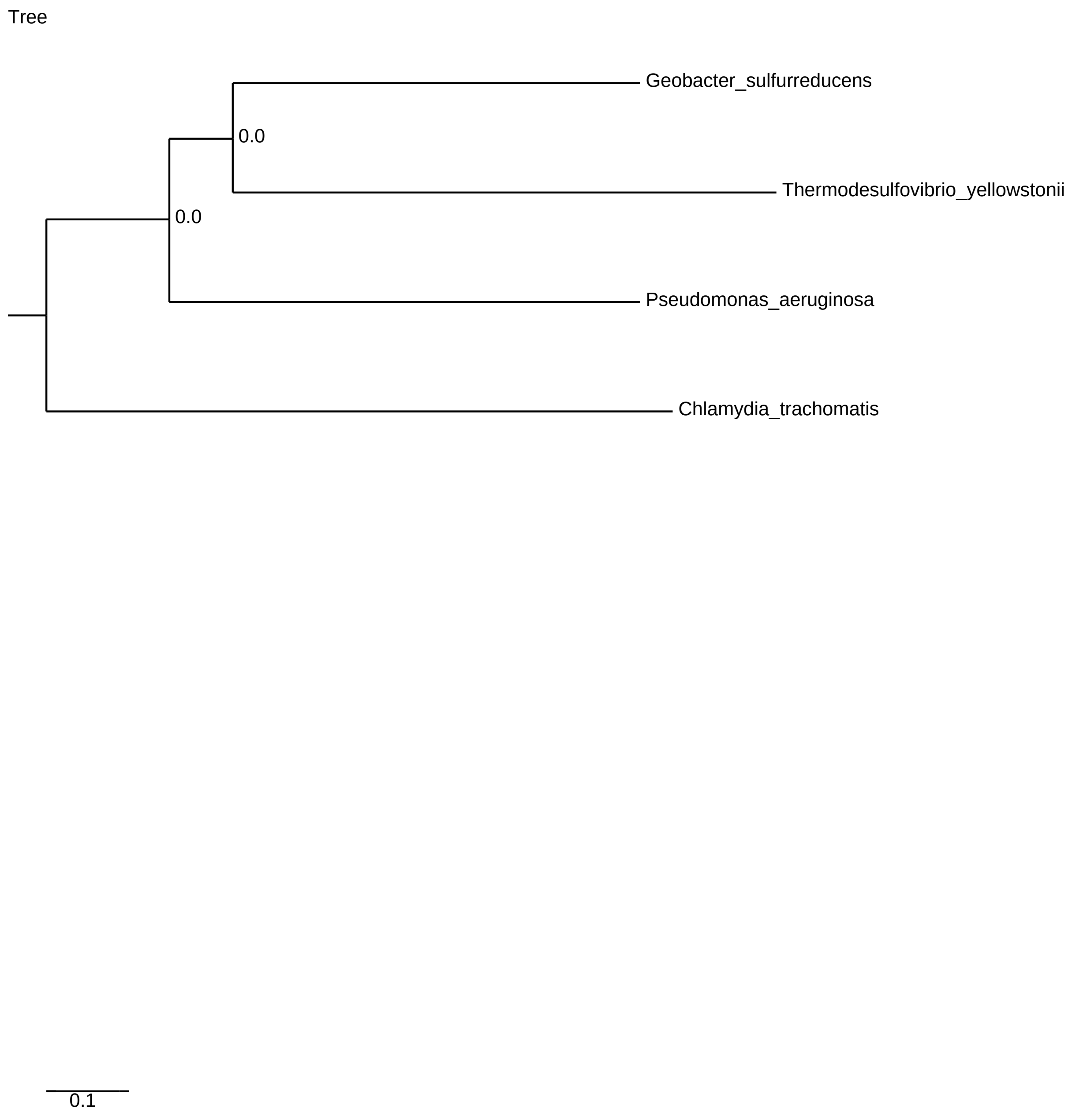

Figure 2: Tree after sanity check