Practical 2: Gene Prediction
Group number: 2
Group members: Maximilian Senftleben, Zhong Hao Daryl Boey

Summary: In this exercise, 2 programs were used for the purposes of gene prediction from the genomes given in practical 1. Additionally, some basic scripts were written to facilitate parsing and extraction of protein sequences from the prediction programs. Glimmer was first used for gene prediction in microbial DNA, which accounts for the majority of our given genomes. The practical addressed the shortcomings of glimmer in this aspect, due to the complications involved in long ORFs in eukaryotes, due to the possibility of gene splicing in more advanced organisms. The alternative program used was GENSCAN, which facilitated gene prediction and gave outputs of both nucleotide and amino acid sequences, with the advantage of visualising the locations of the various genes in the given genome.

Exercise 1 - Glimmer

Steps for the first part of the exercise. Please explain the parameters you are using to run Glimmer. Check those in the PDF provided.

1. Find long ORF from genome
 tigr-glimmer long-orfs -n -t 1.15 01.fa 01.long-orf-coords

long-orf
In this first step, a probability model of coding sequences is produced. The program *long-orfs* derives the training sequences needed for that probability model from large, non-overlapping ORFs from the given genome.
-t
With the value after the flag -t (cutoff), one can set the maximum entropy distance for genes. The genes with a higher entropy distance are cut out before calculating the algorithm.
-n
The program-settings information, which are normally in the header, are left out with -n.
01.fa
Input file.
01.long-orf-coords
Output file.

2. Extract long ORF
 tigr-glimmer extract -t 01.fa 01.long-orf-coords > 01.longorf

extract
The extract program takes a list of gene coordinates and its corresponding genome sequence as input and creates a multi-fasta file according to the by the coordinates specified regions.
01.fa
input sequences
01.long-orf-coords
input coordinates
01.longorf
output file

3. Prepare training set

tigr-glimmer build-icm -r 01.icm < 01.longorf

build-icm
With this program, the interpolated context model is built given the multi-fasta file previously produced.
01.longorf
multi-fasta input file
01.icm
output interpolated context model
-r
with the flag -r, the model is produced in the reverse direction, as it takes the input string in reverse.

4. Start glimmer
tigr-glimmer glimmer3 -o50 -g110 -t30 01.fa 01.icm 01.glimmer
glimmer 3
This program creates a file of gene predictions given the input sequences.
-o50
The maximum overlap length is set to 50
-g110
Here, the minimum length of the gene is set to 110 nucleotides.
-t30
With the -t flag, one can set a consideration-threshold (integer). This score determines, whether the region is considered a potential gene. It is compared to the in-frame score of the region.
01.fa
input genome
01.icm
input interpolated context model
01.glimmer
output tag, there are two output files created, a detailed file with information about all of the ORFs (01.glimmer.detailed) and a file only containing the resulting predictions (01.glimmer.predicted)

5. Long ORFs are provided to construct the training set, what other two sources of sequences can be used instead of or in addition to long ORFs?

Besides using long, non-overlapping ORFs, it is possible to use known genes from the genome, which can be derived from homology searches, or to use genes, which are evolutionary related, as in similar species.

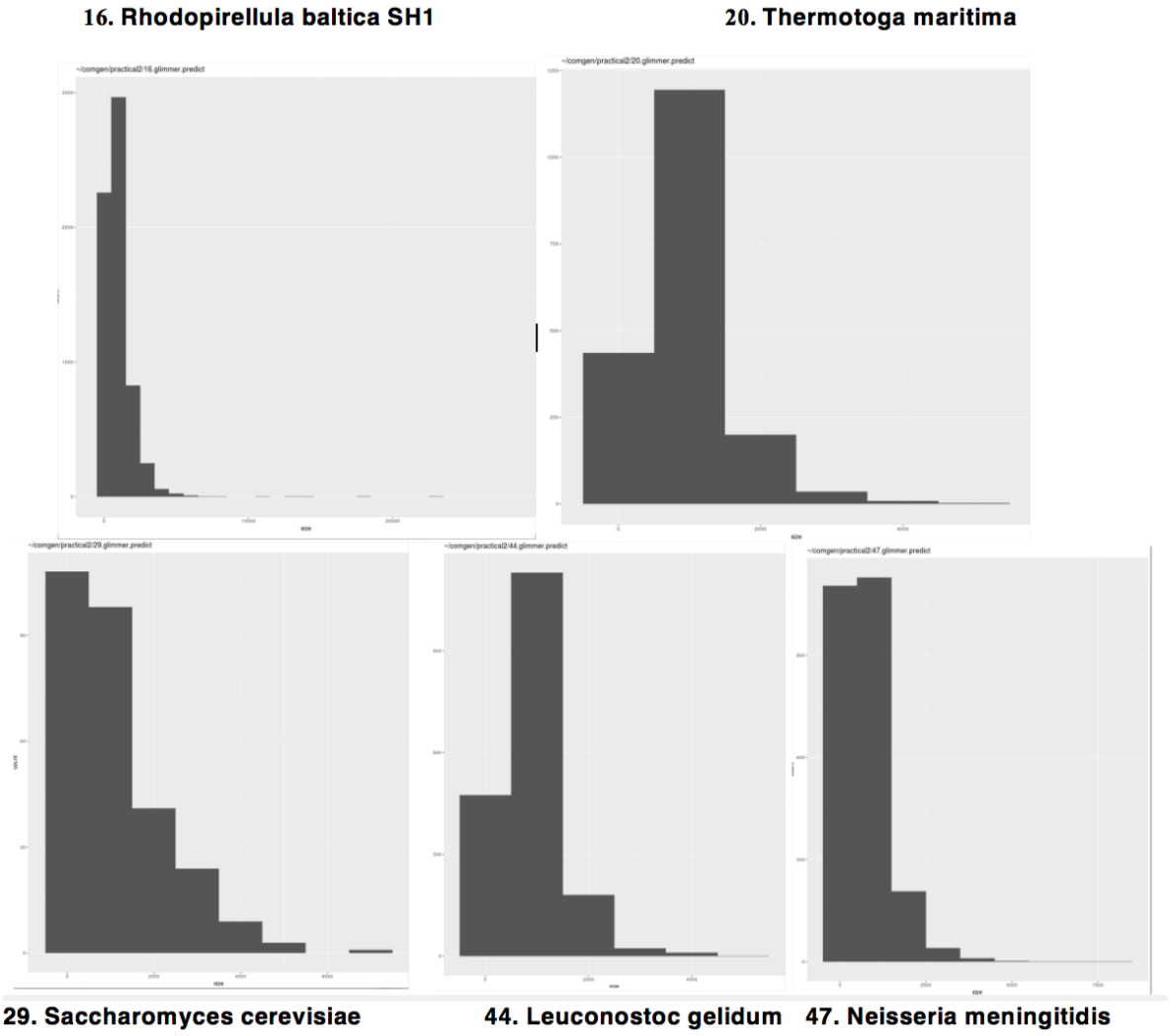
6. Is Glimmer suitable for all genomes? Why?

Glimmer and its programs are exclusively used for prediction of genes in microbial genomes. For eukaryotes, it does not account for the case that stop codons can be in introns and would therefore not terminate the ORF.

7. Make a histogram of predicted gene lengths for each genome in R

install.packages('ggplot2')

```
library(ggplot2)
plotGlimmer = function (file= '01.glimmer.predict' ) {t = read.table(file, header = F, skip = 1 )c =
data.frame(size=abs(t[, 2 ]-t[, 3 ]))ggplot(c, aes(size))+geom_histogram(binwidth=
1000 )+ggtitle(file)
}
plotGlimmer()
```



8. Do all gene sizes follow the same distribution in all genomes?

The shape of the prokaryotes histograms (16, 20, 44, 47) look more or less the same. Compared to prokaryotes (16, 20, 44, 47), eukaryotes (29) have less genes relative to their genome size, as eukaryotes have introns. The prokaryotic genes are less dense.

9. Extract the protein sequences from the predicted genes obtained. Use the script parseGlimmer.py.2 available in the script directory.

The pfa-files are attached.

Exercise 2 - GENSCAN

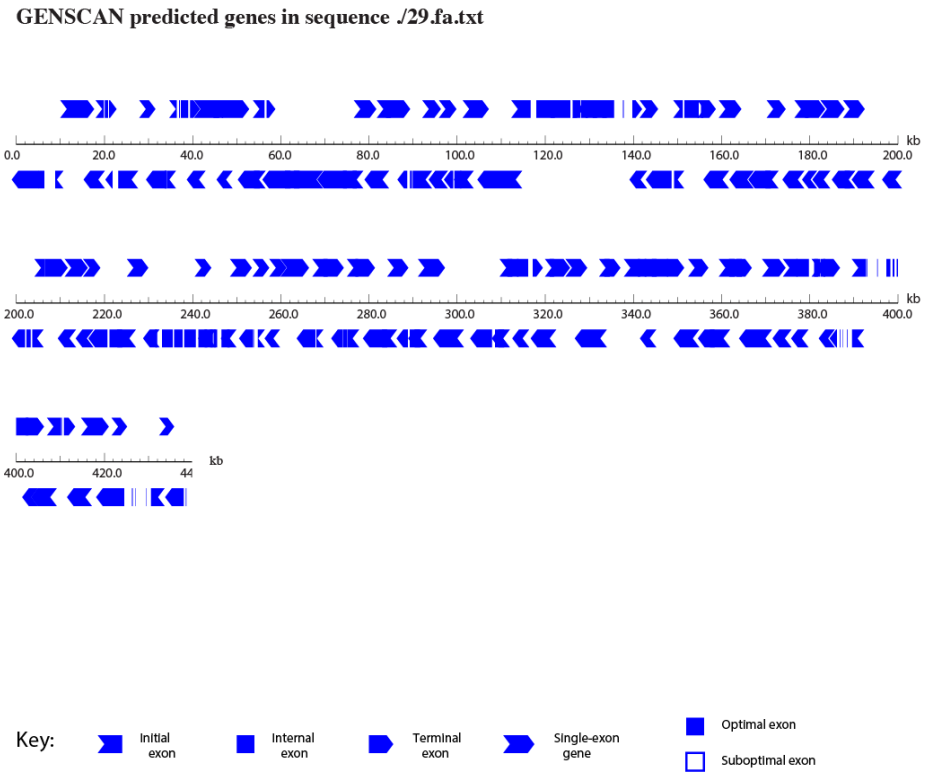
Steps for the second part of the exercise.

Run **GENSCAN** for the eukaryote provided in Practical 1. Run it, using **HumanIso.smat** .

8. From GENSCAN output, extract the amino acid and nucleotide sequences and make separate files for each.

   The files are called peptide_out and nucleotide_out respectively.

9. Create the PostScript (graphical) output, which is a diagram of the locations and DNA strand of all predicted exons/genes.



10. Using BLAST and the nucleotide sequences extracted from GENSCAN output, tell me the protein names of the first two nucleotide sequences.

    >./29.fa.txtlGENSCAN_predicted_peptide_1l2058_aa: Y' element ATP-dependent helicase protein 1

    >./29.fa.txtlGENSCAN_predicted_peptide_2l1549_aa: Signal sequence-binding protein