Comparative genomics final report: ORF predictor FuMiSh

by Shuhan Xu, Fuqi Xu, Milda Valiukonytė

In this project, we performed three types of investigations our five genomes: analysis of genetic composition (GC, nucleotide, dinucleotide, amino acid and diamino acid frequencies), prediction of open reading frames (ORFs) and calculation of distances between genomes to understand their evolutionary relationships. We wrote Python scripts for these investigations. Details about our python scripts can be found at the "Python Scripts" at the end of the report.

The five genomes includes four prokaryotes and one eukaryote: *Escherichia coli* of *Enterobacteriaceae* bacterial family (genome length 5443340 bp), *Streptomyces coelicolor* of *Streptomycetaceae* family (9054847 bp), *Rubrobacter xylanophilus* of *Rubrobacteraceae* family (3225748 bp), *Spiribacter curvatus* of *Ectothiorhodospiraceae* family (1926631 bp), and *Saccharomyces cerevisiae* from *Saccharomycetaceae* yeast family (1531933 bp).

1. Genetic composition analysis

The genetic composition analysis showed that there is a great variation among the five genomes regarding their GC, nucleotide, dinucleotide, amino acid and diamino acid frequencies. The nucleotide analyses were performed on the genomes given to us while the amino acid analyses were performed on the proteome predicted by our ORF predictor. The GC content was calculated using the formula:

$$GC = \frac{count(G) + count(C)}{length(genome)}.$$

Both *S.coelicolor* and *R.xylanophilus* have high GC content (0.72 and 0.70). *S.cerevisiae* had the low GC content (0.38) (Fig. 1)

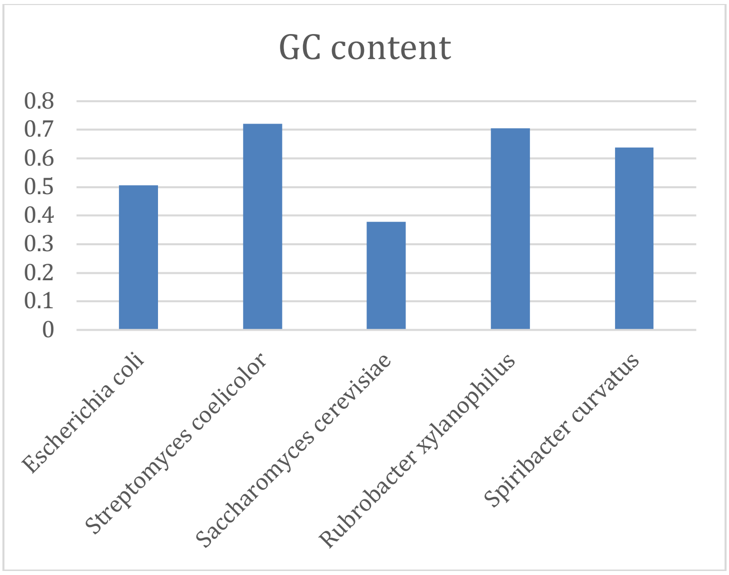

Figure 1: GC content among genomes

Nucleotide frequencies were calculated using the formula:

$$frequency(nucleotide) = \frac{count(nucleotide)}{length(genome)}$$

The results are shown in Fig. 2 below. *S.coelicolor* and *R.xylanophilus* have very similar nucleotide frequencies profiles. *S.curvatus* has slightly similar profiles to the two.

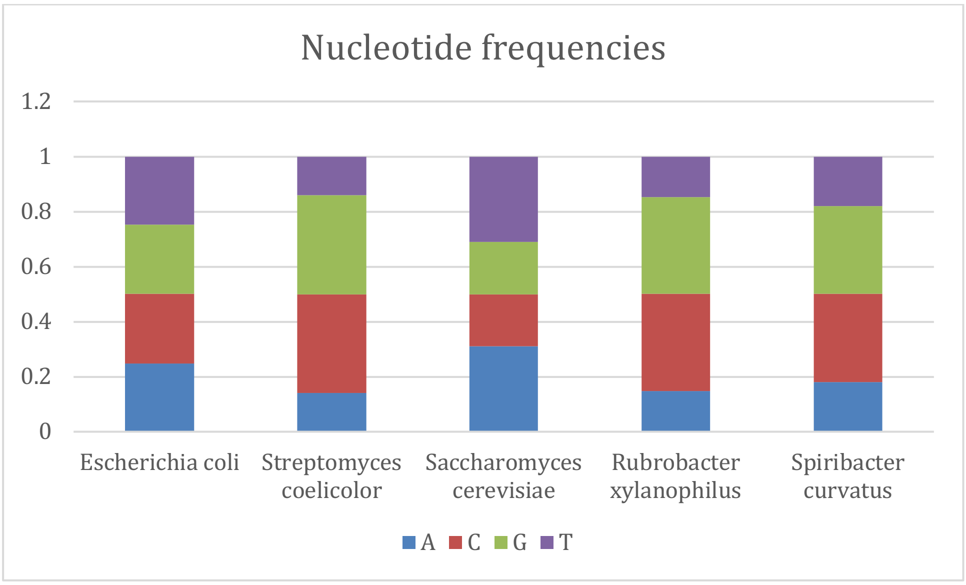

Figure 2: Nucleotide frequencies among genomes

Dinucleotide frequencies were calculated using the formula:

$$frequency(dinucleotide) = \frac{count(dinucleotide)}{length(genome) - 1}$$

The results are shown in Fig. 3. *S.coelicolor* and *R.xylanophilus* again have very similar dinucleotide profiles, particularly in CC, CG, GC and GG frequencies. This is not surprising since both organisms have high GC content results. *S.curvatus* have similar but slightly lower CC, CG, GC and GG frequencies as compared to the two organisms above. This is because *S.curvatus* have the third highest GC content after the two.

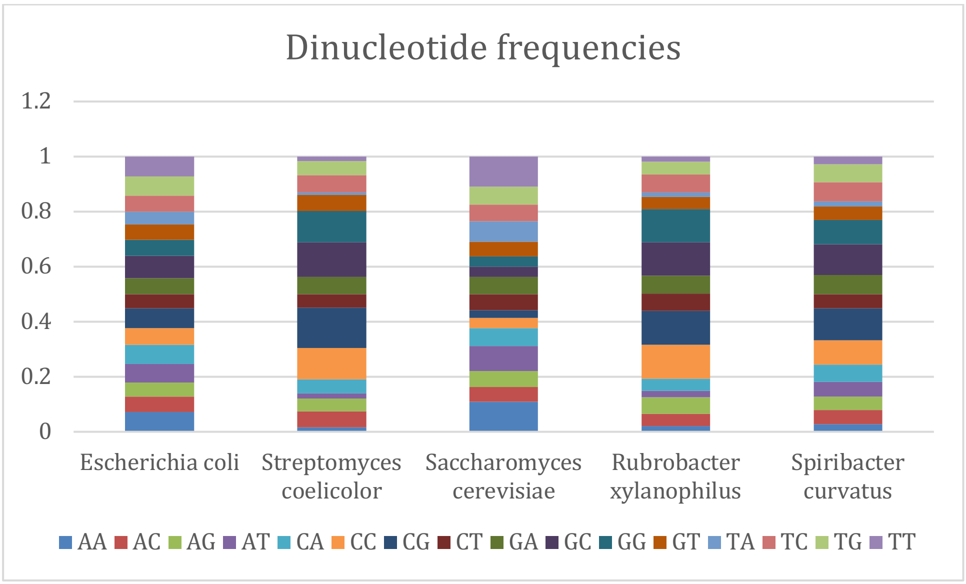

Figure 3:Dinucleotide frequencies among genomes

The following two equations are used to calculate amino acid and diamino acid frequencies:

$$frequency(amino\ acid) = \frac{count(amino\ acid)}{\sum length(protein)}$$

$$frequency(diamino\ acid) = \frac{count(diamino\ acid)}{\sum(length(protein) - 1)}$$

Amino acid frequencies are shown in Fig. 4. Diamino acid frequencies are not shown as there are 400 diamino acids. S.coelicolor and R.xylanophilus have very similar amino acid profiles as well, particularly in alanine (A), arginine (R), glycine (G) and proline (P). This is because codons for A, R, G and P start with the dinucleotides GC, CG, GG and CC respectively and both organism have high GC content. S.curvatus have similar but lower frequencies for A, R, G and P since it has lower GC content than the two organisms.

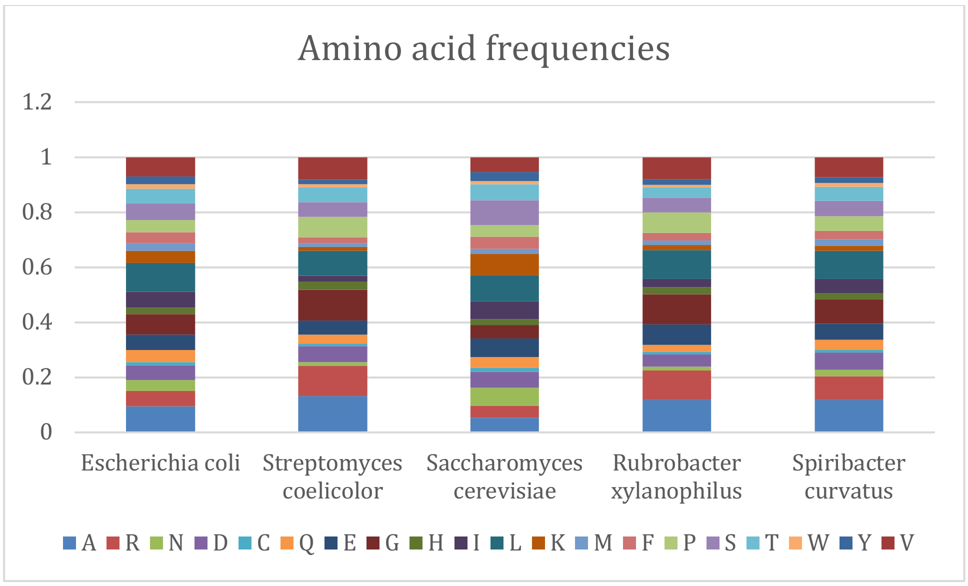

Figure 4:Amino acid frequencies among genomes

From this exercise, we can draw two conclusions. First, we inferred that *S.coelicolor* and *R.xylanophilus* are more closely related to each other than to the rest of the studied organisms. *S.curvatus* is next closest to the two. Second, we can see that GC content influences nucleotide frequencies which influences dinucleotide frequencies which subsequently influences amino acid frequencies.

2. ORF prediction

Next, we used our own written script to predict open reading frames (ORFs) in our genomes. For the prediction, we made the following basic assumptions. First, an ORF starts with a start codon (ATG) and ends with a stop codon (TAA, TAG, TGA). Second, if more than one ORF share the same stop codon, the longest ORF is kept. This is because in a random sequence, a stop codon will appear approximately every 20 codons (3 stop codons in 64 possible codons). Hence, the longer a sequence without a stop codon, the more likely that it is not random but an ORF. Third, the minimum length of genes is 200bp [1] in prokaryote and 300bp [2] in eukaryotes (excluding the stop codons). Fourth, the maximum overlap of ORF is 60bp [3]. Fifth, if two ORFs overlap by more than 60bp, we keep the longer ORF. We predicted 5079 ORFs in *Escherichia coli*, 7172 ORFs in *Streptomyces coelicolor*, 724 ORFs in the fourth chromosome of *Saccharomyces cerevisiae*, 2778 ORFs in *Rubrobacter xylanophilus* and 1758 ORFs in *Spiribacter curvatus*. We named our predictor ‘FuMiSh’ based on the first two characters of our first names.

Evaluation of predictor

To evaluate the performance of FuMiSh, we first compared it with the state of art predictor, GLIMMER and assumed that the predictions of GLIMMER are real ORFs. We then compared the predicted ORFs of FuMiSh with the proteomes from UniProt.

1) Comparison with GLIMMER

For this comparison, we ran FuMiSh using minimum gene size of 110bp and maximum overlap of 50bp because these were the settings we used for GLIMMER in practical 2. We compared the predictions of FuMiSh against the predictions of GLIMMER at the nucleotide level. For each reading frame, each nucleotide was classified as True Positive (TP), True Negative (TN), False Positive (FP) or False Negative (FN). Nucleotide predicted as ORF nucleotide by both GLIMMER and FuMiSh is a TP; nucleotide predicted as ORF nucleotide by GLIMMER but not FuMiSh is a FN; nucleotide predicted as ORF nucleotide by FuMiSH but not GLIMMER is a FP; and nucleotide not predicted as ORF nucleotide by both GLIMMER and FuMiSh is a FN. The following accuracy metrics were calculated:

$$\text{Sensitivity} = \frac{TP}{TP + FN}$$

$$\text{Specificity} = \frac{TP}{TP + FP}$$

$$\text{Approximate Correlation Coeffiicient(AC)} = \frac{1}{2} * \left(\frac{TP}{TP + FN} + \frac{TP}{TP + FP} + \frac{TN}{TN + FP} + \frac{TN}{TN + FN}\right) - 1$$

The results are summarized in Table 1.

Table 1: The comparison of ORF predictions between GLIMMER and FuMiSh

Species	No. of GLIMMER ORF	No. of FuMiSh ORF	Ave. GLIMMER length(bp)	Ave. FuMiSh length(bp)	Sensitivity	Specificity	AC
E.coli	5323	6329	888	773	0.95	0.917	0.922
S.coelicolor	8548	8175	939	954	0.59	0.61	0.529
S.cerevisiae ch4	921	1604	1235	782	0.97	0.88	0.91
R.xylanophilus	3375	3167	879	869	0.623	0.671	0.586
S.curvatus	1862	1955	973	897	0.841	0.869	0.829

For *S.coelicolor*, *R.xylanophilus* and *S.curvatus*, the number of ORFs predicted by FuMiSh is close to the number of ORFs predicted by GLIMMER. For *E.coli* and *S.cerevisiae*, FuMiSh greatly overpredicted the number of ORFs. The average length of ORFs predicted by FuMiSh is shorter than the average length of ORFs predicted by GLIMMER. To better understand these results, we plotted the size distributions of the ORFs predicted by GLIMMER and FuMiSh for each organism (Fig. 5). From Fig. 5, we saw that for ORFs longer than 500bp, GLIMMER and FuMiSh have similar distribution. However, for ORFs shorter than 500bp, FuMiSh tends to overpredict the number of ORFs, especially for *E.coli* and *S.cerevisiae*. This also explains why FuMiSh has shorter average ORFs length than GLIMMER. The reason why GLIMMER did not overpredict the number of short ORFs even when we applied the same minimum ORF length for both predictors is that GLIMMER employed interpolated Markov Model. It used long ORFs which tend to be correct to train the model. From the long ORFs, the model learnt the promoter and ribosomal binding site signals and used them to filter out the incorrect short ORFs.

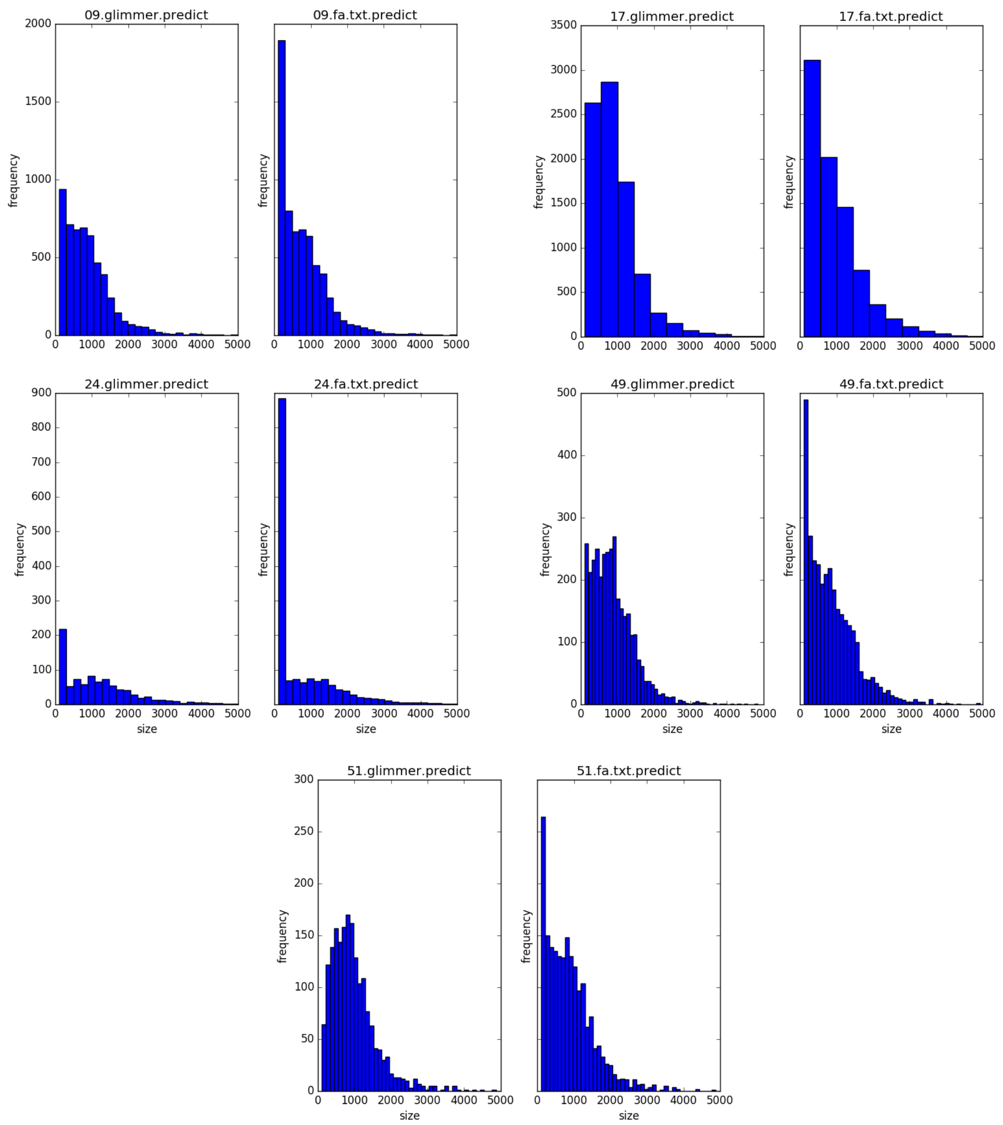

Figure 5: The gene length distribution of GLIMMER(left) and Fumish(right). 09 is *E.coli*, 17 is *S.coelicolor*, 24 is *S.cerevisiae*, 49 is *R.xylanophilus* and 51 is *S.curvatus*.

Next, we examined the prediction accuracy of FuMiSh. From Table 1, FuMiSh performed well for *E.coli*, *S.cerevisiae* and *S.curvatus*, with approximate correlation coefficient (AC) of 0.922, 0.91 and 0.829 respectively. However, FuMiSh performed poorly for *S.coelicolor* (0.529) and *R.xylanophilus* (0.586). This is because *S.coelicolor* and *R.xylanophilus* used alternate start codons, GTG and TTG, which code for valine and Leucine, in addition to ATG, which codes for Methionine (see Table 2). In fact, more than 40 percent of start codons in *S.coelicolor* and *R.xylanophilus* are GTG. This could be due to the fact that the two organisms are GC rich, and thus have a higher tendency to use GTG as a start codon. Because FuMiSh is unable to predict the actual start codons of *S.coelicolor* and *R.xylanophilus*, the AC values are much lower for these two organisms.

Table 2: Alternate start codons usage

Species	**Fraction of GTG and TTG start codons**	**Fraction of GTG start codons**
E.coli	0.199	0.140
S.coelicolor	0.454	0.408
S.cerevisiae ch4	0.194	0.106
R.xylanophilus	0.562	0.438
S.curvatus	0.249	0.186

2) Comparison with UniProt proteomes

To better evaluate the performance of FuMiSh, we translated the predicted ORFs for each organism and compared them with the organism's proteome in UniProt. To identify the proteins that were correctly predicted, we performed two BLASTp searches. First, we used the UniProt proteome as database and the predicted proteins as queries. Then, we used the UniProt proteins as queries and the predicted proteome as dataset. An E-value threshold of 0.001 was used for both searches. If two proteins were reciprocal best-scoring BLAST hits of each other, the pair was considered as a true positive prediction.

True positive (TP): reciprocal best-scoring BLAST hits protein pairs

False positive (FP): Proteins that were in predicted proteome but not in reciprocal best-scoring BLAST hit protein pairs

False negative (FN): Proteins that were in the UniProt proteome but not in reciprocal best-scoring BLAST hit protein pairs

The sensitivity and specificity are calculated using formulas in comparison with GLIMMER. The F1 score measures the overall prediction accuracy.

$$\mathrm{F1} = \frac{2 * \mathrm{TP}}{2 * \mathrm{TP} + \mathrm{FP} + \mathrm{FN}}$$

Table 3 is the summary of the results.

Table 3: The comparison of proteins predicted by FuMiSh and UniProt proteomes

Species	UniProt protein number	Predicted gene number	Reciprocal best hit number	Specificity	Sensitivity	F1 score
E.coli	4313	5079	3691	0.727	0.856	0.786
S.coelicolor	7731	7172	4009	0.559	0.519	0.538
S.cerevisiae	769	724	711	0.982	0.925	0.952
R.xylanophilus	3127	2778	1945	0.700	0.622	0.659
S.curvatus	1861	1752	1460	0.833	0.785	0.808

The results agreed with our comparison with GLIMMER. The F1 score is high for *E.coli* (0.786), *S.cerevisiae* (0.952) and *S.curvatus* (0.808) but low for *S.coelicolor* (0.538) and *R.xylanophilus* (0.659).

Areas to improve

1) Adding promoter information

Adding promoter information filters small ORFs and selects the true ORF among ORFs which share the same stop codon or ORFs which overlap. Pribnow boxes are used as the promoter in prokaryotes. The Pribnow box is located at 10bp upstream the transcriptional start site [4], and the transcription start site is 20 to 40 nucleotides upstream the start codon [5]. We can search for the Pribnow box and select the start codons that are 30bp-50bp downstream the Pribnow box. In eukaryotes, TATA boxes information can be used in the same way as the Pribnow box.

In our *E.coli* genome, there are only around 700 Pribnow boxes(TATAAT) for 5000 genes. The major explanation is that in prokaryotes, one operon is shared by several genes. In such case, the Pribnow box can't be used directly as a selection criterion in ORF finding. Instead, it can be used to validate prediction, for example selecting ORFs which share the same stop or overlapping ORFs. This information can be applied to the model training in machine learning based predictions.

2) The minimum gene length

We selected the minimum gene length in *E.coli*(200bp) [1] as the minimum gene length for prokaryotes and the minimum gene length in yeast (300bp) [2] to filter small genes in eukaryotes. The minimum gene length varies in different species and the threshold should be set according to different species.

3) Overlapping genes

Overlaps in different reading frames that are longer than 60bp [3] are forbidden in our predictor. In this case, we only select the longest ORF in the overlapping genes. To improve the prediction accuracy, we can apply different maximum overlapping length in different species. Also, overlapping genes have preference in orientation, convergent overlapping genes are three times more likely than divergent overlapping genes. [6] We should also consider the orientation when selecting overlaps.

4) Alternative start codons

Besides ATG, GTG and TTG should also be considered when finding searching for start codons. In the prediction result of FuMiSh, the prediction accuracy (AC) decreased with the increase of GC content (Fig. 6). This is because high GC content organisms have a higher tendency to use alternative start codons such as GTG instead of ATG. Hence, FuMiSh had a lower prediction accuracy for these organisms. Adding alternative start codon information can improve the prediction accuracy.

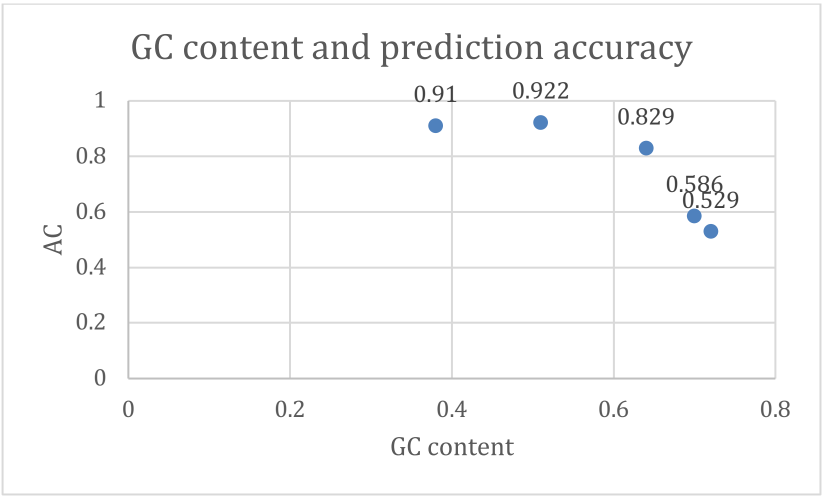

Figure 6: GC content and prediction accuracy

Evolutionary relationship

For the final part, evolutionary relationship analysis, we chose dinucleotide frequencies to calculate the distance between genomes and visualized the evolutionary relationship between species in a phylogenetic tree using neighbor-joining method (Fig 7). The motivation for choosing dinucleotide frequencies for distance calculation is that it has been shown that dinucleotide frequency distributions are quite species specific and are even considered a genomic signature [7]. Secondly, we had chosen this characteristic over amino acid or diamino acid frequencies because the latter two are based on predictions which many not be completely correct.

To calculate distance, we first normalized the values for each feature (e.g. 'AA', 'AC', 'AG', …) across all organisms using the formula: $x' = \frac{x - \bar{x}}{\sigma}$, where $\bar{x}$ and σ are the mean and standard deviation of a feature across all organisms. Then we calculated Euclidean distance using the formula: $d = \sqrt{\sum_{i=1}^{n}(X_i^{(1)} - X_i^{(2)})^2}$, where $X^{(1)}$ and $X^{(2)}$ are the vectors of features for the first and second organisms respectively.

From the distances, we used Belvu[8] to construct a Neighbor Joining tree. We choose Neighbor Joining over UPGMA because the former does not assume a constant evolutionary rate. The tree is displayed in Fig. 7.

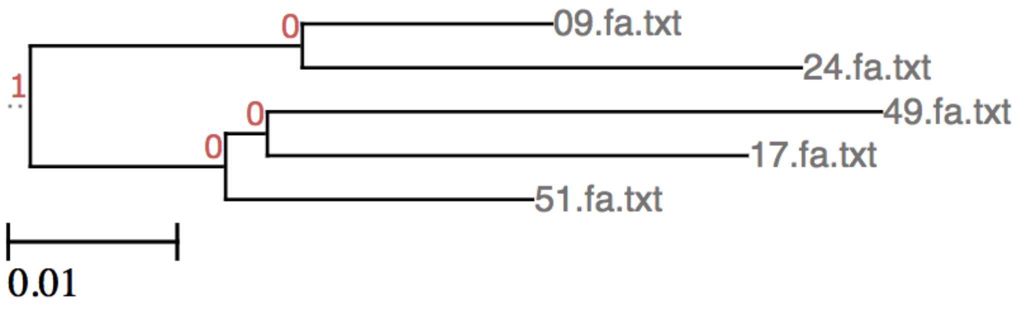

Figure 7 : Evolutionary relationship between species based on dinucleotide frequencies

The topology of the tree is consistent with the topologies of the trees we created in previous practicals (Fig. 8: 16s rRNA, Fig. 9: metagene of 10 orthologue clusters, Fig. 10: consensus tree of 10 orthologue clusters and Fig 11: gene order). Even though dinucleotide frequencies has less information content than the features we used to build the other trees, we can still draw the same conclusion about the species' evolutionary relationships. This is perhaps due to the fact that these species are evolutionary distant from each other, thus allowing dinucleotide frequencies to accurately capture their evolutionary relationships.

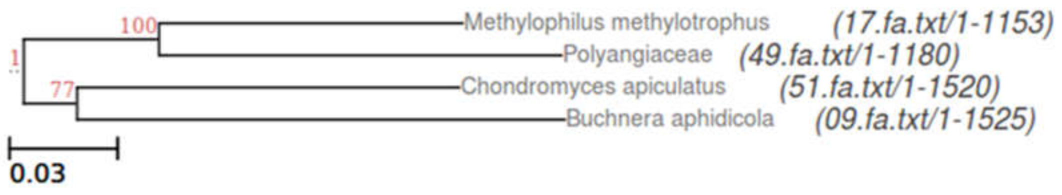
100
1
77
Methylophilus methylotrophus (17.fa.txt/1-1153)
Polyangiaceae (49.fa.txt/1-1180)
Chondromyces apiculatus (51.fa.txt/1-1520)
Buchnera aphidicola (09.fa.txt/1-1525)
0.03

Figure 8: Evolutionary relationship between species based on 16s rRNA

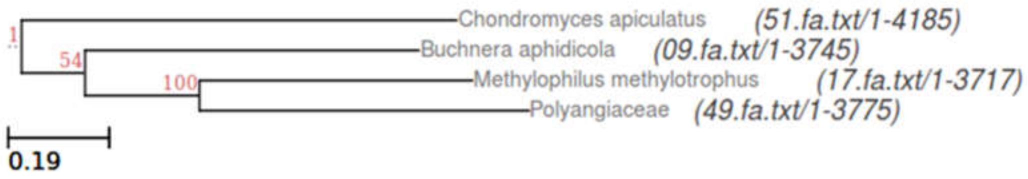
1
54
100
Chondromyces apiculatus (51.fa.txt/1-4185)
Buchnera aphidicola (09.fa.txt/1-3745)
Methylophilus methylotrophus (17.fa.txt/1-3717)
Polyangiaceae (49.fa.txt/1-3775)
0.19

Figure 9: Evolutionary relationship between species based on metagene of 10 orthologue clusters

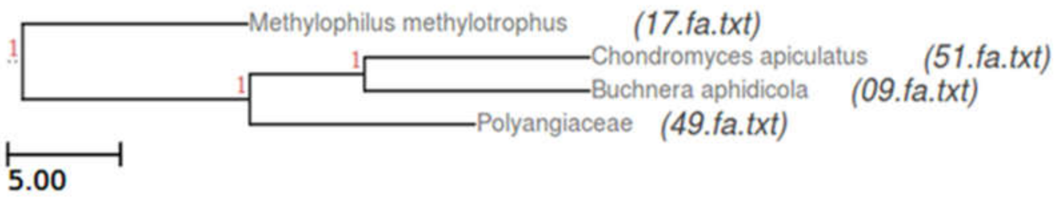
1
1
1
Methylophilus methylotrophus (17.fa.txt)
Chondromyces apiculatus (51.fa.txt)
Buchnera aphidicola (09.fa.txt)
Polyangiaceae (49.fa.txt)
5.00

Figure 10: Evolutionary relationship between species based on consensus tree of 10 orthologue clusters

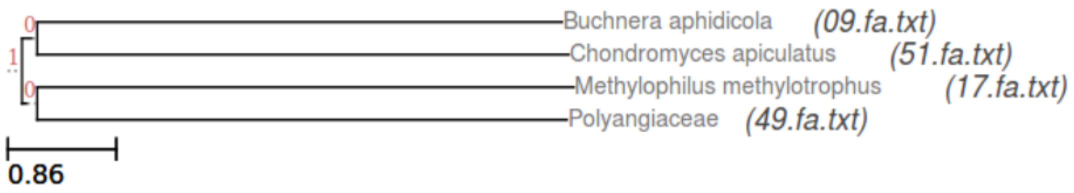
0
1
0
Buchnera aphidicola (09.fa.txt)
Chondromyces apiculatus (51.fa.txt)
Methylophilus methylotrophus (17.fa.txt)
Polyangiaceae (49.fa.txt)
0.86

Figure 11: Evolutionary relationship between species based on gene order

Reference:

1. Blattner, Frederick R., et al. "The complete genome sequence of Escherichia coli K-12." Science 277.5331 (1997): 1453-1462.

2. Clément-Ziza, Mathieu, et al. "Natural genetic variation impacts expression levels of coding, non-coding, and antisense transcripts in fission yeast." Molecular systems biology 10.11 (2014): 764.

3. Basrai, Munira A., Philip Hieter, and Jef D. Boeke. "Small open reading frames: beautiful needles in the haystack." Genome research 7.8 (1997)

4. Zvelebil, Marketa J., and Jeremy O. Baum. Understanding bioinformatics. Garland Science, 2007.

5. Mendoza-Vargas, Alfredo, et al. "Genome-wide identification of transcription start sites, promoters and transcription factor binding sites in E. coli." *PLoS One* 4.10 (2009): e7526.

6. Hyatt, Doug, et al. "Prodigal: prokaryotic gene recognition and translation initiation site identification." BMC bioinformatics 11.1 (2010): 119.

7. Kariin, Samuel, and Chris Burge. "Dinucleotide relative abundance extremes: a genomic signature." *Trends in genetics* 11.7 (1995): 283-290.

8. Sonnhammer, Erik LL, and Volker Hollich. "Scoredist: a simple and robust protein sequence distance estimator." *BMC bioinformatics* 6.1 (2005): 108.

Python Scripts

1. Statistics tool

Script for calculating GC content, nucleotide frequencies and dinucleotide frequencies: *nucl_statistics.py*

Usage: python3 nucl_statistics.py <genome1.fa.txt> <genome2.fa.txt> <genome3.fa.txt> …

Description: The script takes any number of genome fasta files and calculates the GC content, nucleotide frequencies, and dinucleotide frequencies. The results are output to three .csv files ‘gc_freq.csv’, ‘nucl_freq.csv’ and ‘dinucl_freq.csv’ in the working directory.

Script for calculating amino acids frequencies and diamino acids frequencies: *prot_statistics.py*

Usage: python3 prot_statistics.py <proteome1.fa.txt.pfa> <proteome2.fa.txt.pfa> <proteome3.fa.txt> …

Description: The script takes any number of proteome fasta files and calculates the amino acid frequencies and diamino acid frequencies. The results are output to three .csv files ‘amino_freq.csv’ and ‘diamino_freq.csv’ in the working directory.

2. ORF finder

Script for finding open reading frame: *predict_orf.py*

Usage: python3 predict_orf.py <genome.fa.txt>

Description: The script takes one genome fasta file and find ORFs that satisfy the conditions stated in the report above. The coordinates of the predicted ORFs are then saved to ‘genome.fa.txt.predict’ in the working directory.

The ‘genome.fa.txt.predict’ file has the same format as GLMMER’s .predict file. To retrieve the nucleotide or protein sequences, the script ‘parseGlimmer.py.2’ can be used.

python2 parseGlimmer.py.2 <genome.fa.txt> <genome.fa.txt.predict>

or

python2 parseGlimmer.py.2 <genome.fa.txt> <genome.fa.txt.predict> --translate

Script for evaluating the nucleotide level accuracy of our predicted ORFs against GLIMMER ORFs: *evaluate.py*

Usage:	python3 evaluate.py <genome.fa.txt> <genome.glimmer.predict> <genome.fa.txt.predict>
Description:	The script takes the genome fasta file, GLIMMER prediction file and the prediction file generated above and calculates sensitivity, specificity and approximate correlation coefficient based on nucleotide level accuracy in all six reading frames. In addition, it also gives a summary of the number of ORFs and average ORF length of both prediction files. The results are output to the standard output.

Script for plotting the gene length distribution of two predictors: *plotGeneLength.py*

Usage: python3 plotGeneLength.py <predictionResult1> < predictionResult1>

Description: This script takes two gene prediction results and plots the gene distribution of the two predictions.
The input prediction result file should be written in GLIMMER output format.

<column0>	<column1>	<column2>	<others>
Gene name	Start codon position	Stop codon position	…

Script for count the proteins that appears in both BLAST results: *blast_count.py*

Usage: python3 blast_count.py <blastresult1> <blastresult2>

Description: This script takes two blast result files and finds the number of overlapping hits in both files.

3. Evolutionary distance

Script for calculating evolutionary distance: *calculate_distance.py*

Usage: python3 calculate_distance.py <frequency.csv>

Description: This script takes any file generated from *nucl_statistics.py* or *prot_statistics.py* and normalised the data. Then it calculate the Euclidean distances and outputs the results to the standard output. The output can be used as input to Belvu.

Output files

Folder for prediction results:	prediction_result.zip
Description:	contains coordinates, nucleotide sequences and protein sequences of predicted ORFs
Folder for blastp searches:	blastp_query_database.zip
Description:	contains fasta files for building databases and querying databases
Folder for blastp results:	blast_result.zip Contains results from blastp searches.