FACULTY OF BIOSCIENCE ENGINEERING
Comparative Genomics

Assignment Comparative Genomics

Antonietta Salerno r0829410

January 2022

1 Introduction

Comparative genomics primary objective is to identify orthologous elements among species, in order to study molecular evolution and/or transfer functional information. Indeed, the main pillar in this field is that sequence conservation across multiple species is likely to be constrained, which implies biological function. In this report, two species are taken into account to examine their potential orthology relationship [1].

The species analysed are *Python bivittatus*, one of the largest species of snakes and native to Southeast Asia [5], and *Latimeria chalumnae*, which belongs to a rare order of vertebrates related to lungfish and tetrapods and it is distributed mostly in the western part of the Indian Ocean [9]. The genome of the first species under question contains 90 genes, of which all of them are protein coding. On the other hand, the latter presents 23516 out of 23535 protein coding genes, thus 19 of them are non-protein coding.

The set of orthologs is identified using the Best Bi-directional Hit (BBH) approach. A protein tree and lately a species tree are generated from the Multiple Sequence Alignment (MSA) of the orthologous proteins previously spotted and the 16S rRNA respectively, and compared. This will lead to the identification of the most conserved sites in the MSA, thus suggesting the most likely location of functional regions in the sequence under question.

2 Methods

2.1 Best Bi-directional Hits

The complete set of protein sequences for *Python bivittatus* (90 sequences) and *Latimeria chalumnae* (23535 sequences) is retrieved from UniProt database as FASTA files. Successively, a BLAST search is carried out in order to identify the BBH. First, since the computation is too expensive for the web-based tool, the search is done using the package BioPython, which implements the command-line-based version BLAST+ [2]. This choice is more convenient than the direct usage from the command-line for downstream data management. First, the search is done using *Python bivittatus* sequences as query and *Latimeria chalumnae* file as database, while setting the maximum amount of target sequences to 1. Then, the roles have been reversed in order to obtain results for both directions. The two commands are the following:

- blastp -out lat_pyt.tab -outfmt 6 -query latimeria.fasta -max_target_seqs 1 -subject python.fasta
- blastp -out lat_pyt.tab -outfmt 6 -query python.fasta -max_target_seqs 1 -subject latimeria.fasta

The output is a tab-separated dataset (one per BLAST search) containing the query and the subject sequence names, along with the percentage of identity, the coverage, the query sequence's length, the bitscore and the E-value. Other columns present in the datasets have been discarded, being not needed for the purpose of this analysis.

In order to compare the different results and identify the best hit for both the forward and the reverse search, the bitscore value is used as metric. It coincides with the sum of the qualities of the aligned symbols over the whole alignment, thus it is reported as an accurate measure of the alignment strength. However, it is biased towards the length of the sequence, therefore the metric has been corrected by dividing it by the length of the query sequence to obtain a normalised bitscore, which would grant a fairer comparison.

The two datasets are then merged with an inner join, keeping the normalised bitscore and the E-value for both directions. The number of bi-directional hits identified is 17755. After filtering for duplicate rows and taking the maximum value for each duplicated match, the number of hits is reduced to 90, exactly the number of sequences contained in the smaller dataset (*Python bivittatus*). Then, the density of the normalised bitscores is plotted to spot bad quality hits (Figure 1). It is noticeable that the majority of results have a low bitscore, thus the number of orthologs between the two species is supposed to be small.

Finally, it is possible to reduce the set of bi-directional hits in order to identify the best ones, which can be defined as orthologs. To do so, arbitrary thresholds on the evaluation metrics have been set. By first setting an E-value (indicator for significance) < 0.001, 19 hits are discared and thus 71 BBH can be defined as orthologs. If we go further, the density plot in Figure 1 provides hints to decide that matches having a normalised bitscores < 300 for the forward search and < 40 for the reverse search are there to be discared as well. In the end, two orthologs are left, having very high normalised bitscore and very low E-value:

- A0A1C9ZN22_PYTBI - H3A1H8_LATCH, identity = 97.959, norm_bitscore_x = 478, norm_bitscore_y = 49, E-value_x = $1.7xe^{-29}$, E-value_y = $4.5xe^{-30}$
- A0A3G9I9P9_PYTBI - M3XHS5_LATCH, identity = 97.561, norm_bitscore_x = 375, norm_bitscore_y = 49, E-value_x = $2.4xe^{-23}$, E-value_y = $3.9xe^{-31}$

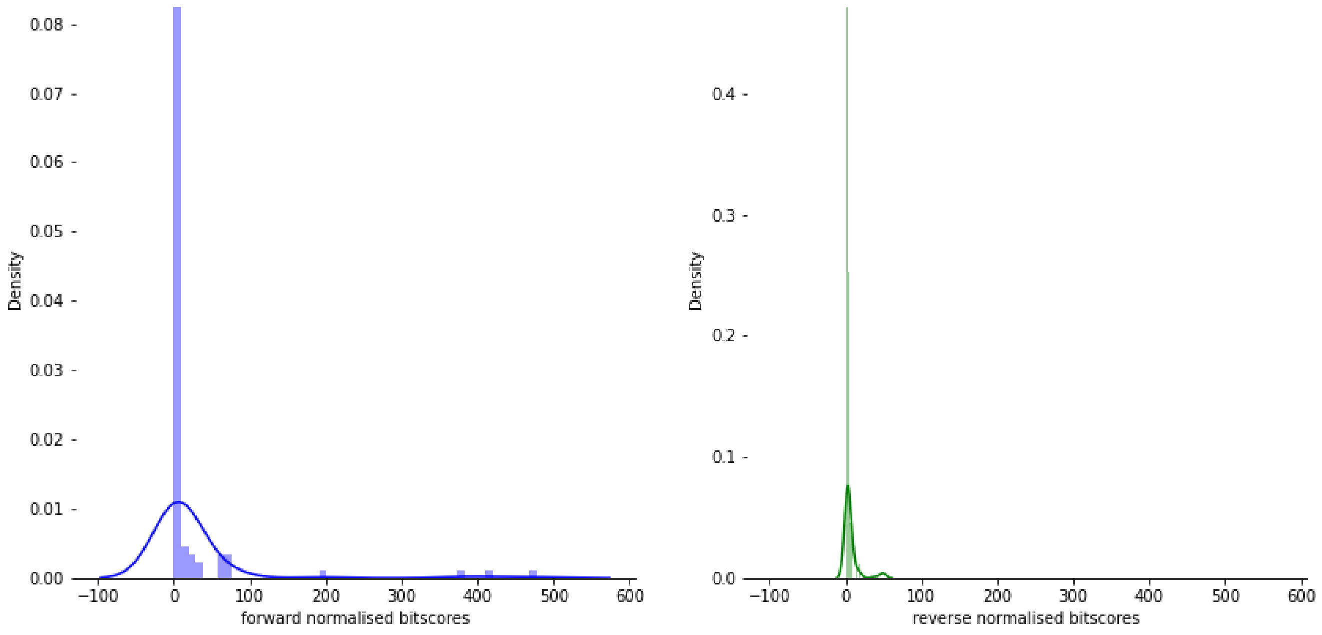

Figure 1: Density plot of the BLAST normalised bitscores of the bi-directional search between *Latimeria chalumnae* and *Python bivittatus*.

The in-house Python script used to carry out this analysis can be found under the section Python script A: Best Bi-directional Hits.

2.2 Phylogenetic tree

The best bi-directional hit, defined as co-ortholog for the two species under question, is used to build a phylogenetic tree with sequences belonging to 26 different organisms in total. The sequence is H3A1H8_LATCH, from *Latimeria chalumnae* genome and it is retrieved from UniProt as FASTA file. It is used to run a query on BLAST and the other 24 sequences from different species have

been selected, their FASTA file downloaded and joined in a unique file. This is used to further run a Multiple Sequence Alignment with ClustalW [8]. From the same platform, a tree-building procedure is executed using PhyML [4] algorithm and 100 as bootstrap parameter. The resulting tree is finally visualised with iTOL interactively and it is illustrated in Figure 2.
Secondly, a species tree is constructed using the DNA sequences coding for the 16S rRNA for every species in the previous phylogenetic tree. These are collected from the European Nucleotide Archive (ENA) manually and joined in a unique FASTA file that will become the input to perform a MSA with ClustalW [8]. After that, PhyML [4] is used to build a species tree, which is then visualised with iTOl, as before (Figure 3).

2.3 Functional Conservation

The ClustalW alignment file from the previous section (Phylogenetic tree) is used as input to highlight sequence conservation patterns from the Multiple Sequence Alignment under question. BioPython is the tool chosen to parse the alignment file. Shannon entropy is used as a metric for sequence conservation, thus it is calculated for every column in the MSA, while implementing a penalty of +9 for those positions in the alignment having a very high number of gaps (> 20, arbitrary chosen). This is needed to prevent the confusion between gaps and highly conserved position during the entropy calculation. The formula is the following:

$$H = -\sum_{i=1}^{M} P_i \, log_2 \, P_i$$

where i is the number of sequences in the alignment, and thus the number of amino acids in the given column, while P is the frequency of the i^{th} amino acid.

The script (see the section Python script B: Conserved regions) returns the most conserved position and produces a dataframe containing the Shannon entropy for every position in the alignment (having total length 1728), sorted by conservation. This is used to generate a plot that highlights the sites having lowest Shannon entropy (Figure 4).

3 Results

3.1 Trees reconciliation

The first analysis reveals that *Python bivittatus* and *Latimeria chalumnae* belong to the same clade in the protein tree generated from the protein coded by the WAC gene (Figure 2). These are indeed orthologs. A comparison with the species tree (Figure 3) produced in the second analysis is needed to distinguish whether some events can be associated to duplication or speciation.

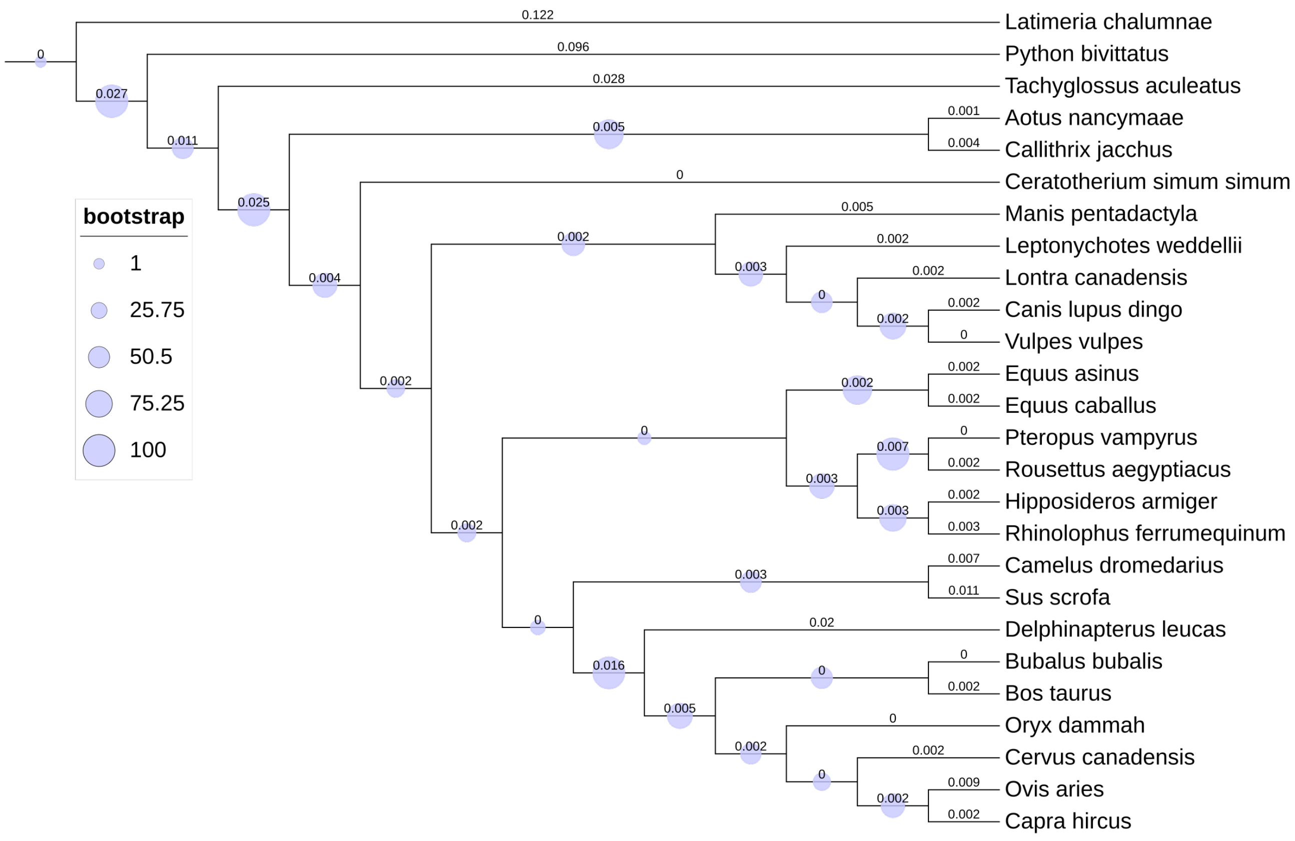

Figure 2: Phylogenetic tree build upon *Latimeria chalumnae* WW domain-containing protein involved in chromatin organisation (coded by WAC gene). The most similar protein sequences coming from 25 different species are nodes in the tree. Bootstrap values and branch lengths are directly shown on the branches.

The species tree in Figure 3 is very different with respect to the phylogenetic tree in Figure 2. The only consistent clade is the one formed by *Python bivittatus*, *Latimeria chalumnae*, *T. aculeatus*, *A. nancymaae* and *C. jacchus*, although it is not clear if these tree last species originated from *Python bivittatus* or *Latimeria chalumnae*. Other constants are the orthology between the following couples:

- *B. taurus* - *B. bubalis*
- *R. ferrumequinum* - *H. armiger*, in the same clade as the couple *P. vampyrus* - *R. aegyptiacus*
- *E. asinus* - *E. caballus*

- *C. lupus dingo* - *V. vulpes*

In addition, it is noticeable that *M. pentadactyla* is much more evolutionary distant to *C. simum simum* and *L. weddellii* in the species tree than in the protein tree and that *C. dromedarius* and *S. scrofa* don't share the same most recent common ancestor (MRCA) anymore.
Finally, it is not possible to identify duplication events since all the sequences belong to different species. Thus, all the events observed are associated to speciation.

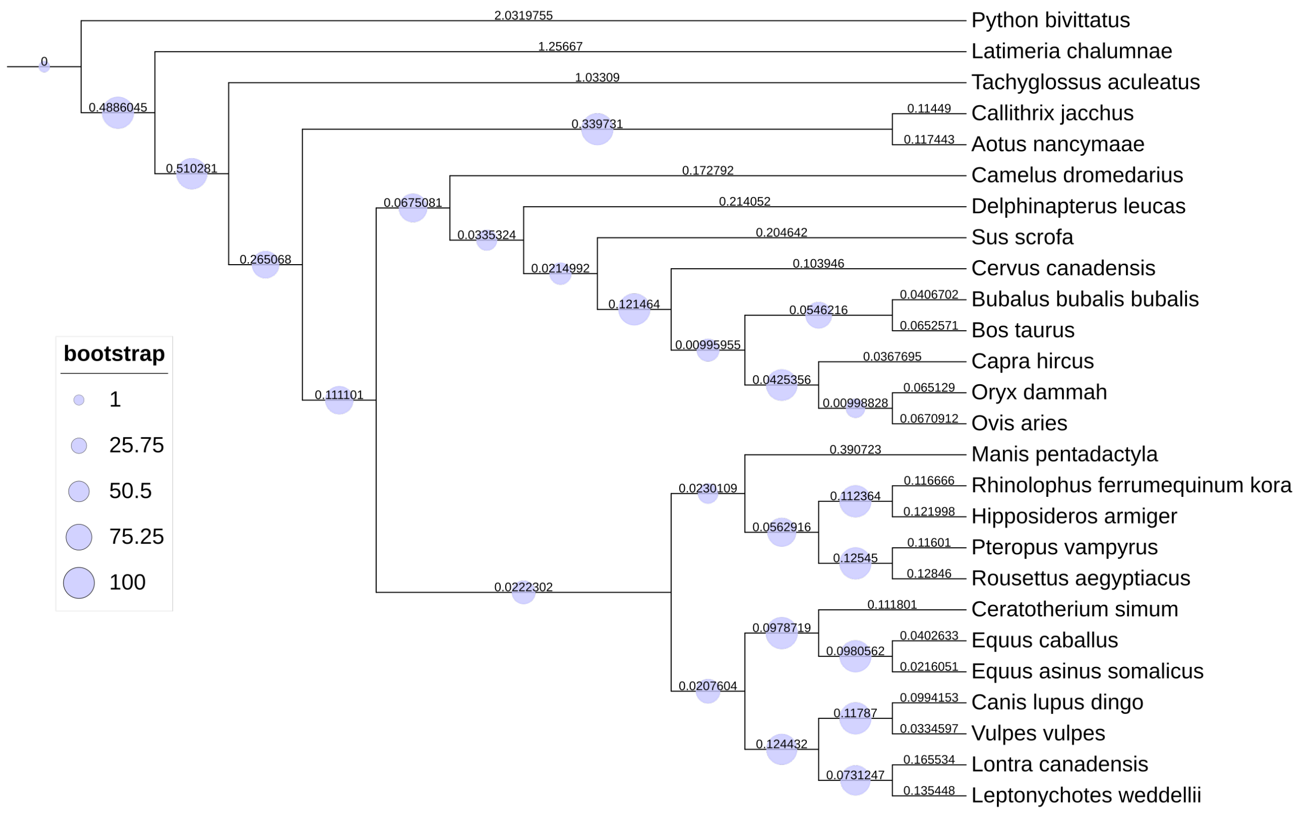

Figure 3: Species tree built upon the 16S rRNA sequences of the species in 2. Bootstrap values and branch lengths are directly shown on the branches.

3.2 Functional Conservation

Eventually, the third analysis revealed that there are actually certain regions in the sequence of a protein that are conserved throughout species. From the Multiple Sequence Alignment of the 16S rRNA for 26 species under question (Figure 4) it is possible to notice that the regions having highest Shannon entropy, thus the least conserved regions, coincide with the 3' and 5'-UTRs, as well as other three main regions in the MSA:

- 425-481 bp
- 536-982 bp
- 1178-1256 bp

Further investigation should be carried out to provide a functional annotation to sequence conservation patterns identified.

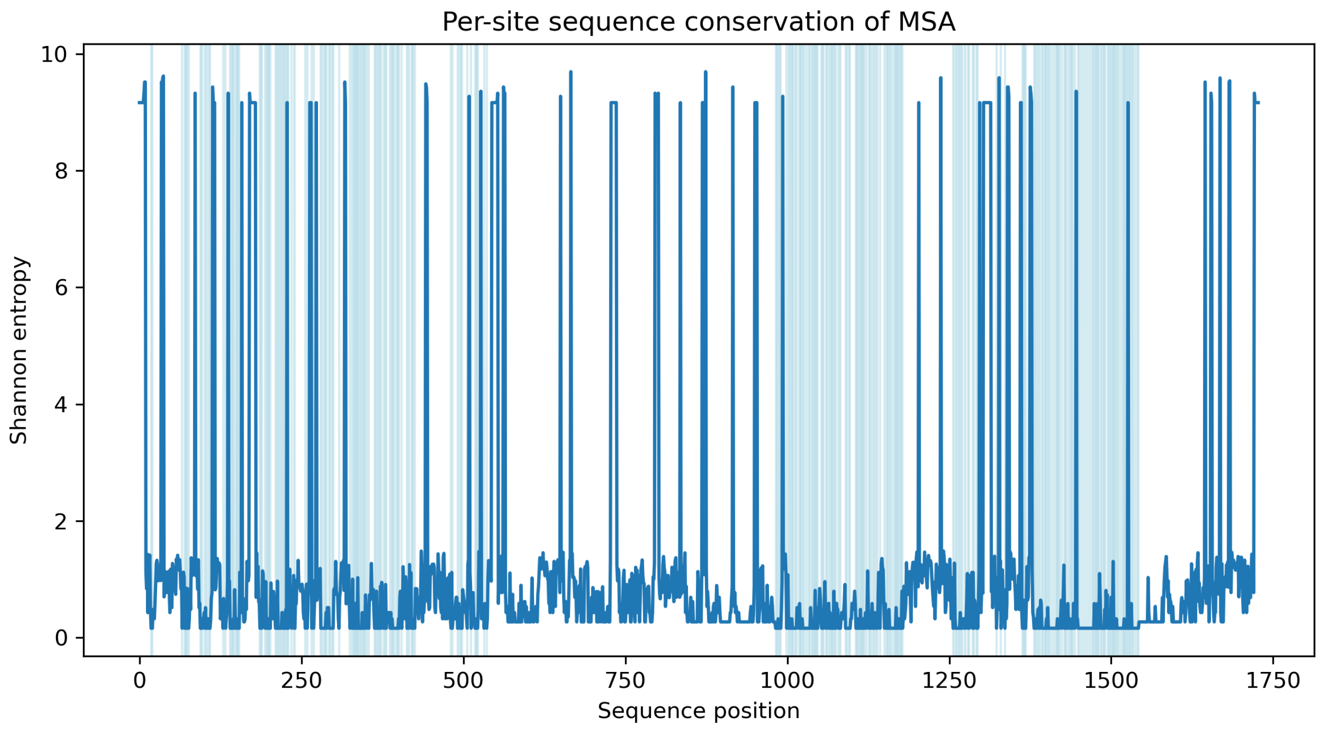

Figure 4: Per-site sequence conservation of the MSA for the 26 16S rRNA sequences of the species in 2. The metric used is Shannon entropy. The most conserved regions are highlighted in light blue.

4 Discussion

A comparative genomics approach is used to identify orthology and functional conservation between *Python bivittatus* and *Latimeria chalumnae*, two species for which a whole genome sequence is publicly available.

The number of orthologous proteins detected varied depending on how stringent was the threshold employed to define the Best Bi-directional Hits found as such. In the end, two significant orthologs were chosen as best candidates and one of them was used to build a phylogenetic tree for 26 sequences belonging to distinct species. The relationships described in this last were questioned from the comparison with the species tree for the same organisms, built on the basis of their 16S rRNA codes. The orthology between *Python bivittatus* [5] and *Latimeria chalumnae* [9] was confirmed, but many clades differed significantly in the comparison. These differences in topology between the protein tree in Figure 2 and the species tree in Figure 3 might be attributed to imprecision related to the algorithm or the tool used during the tree-building process. Further investigation should be done to infer the most efficient approach to detect orthology among these two species, since a higher level of consistency between the protein and the species tree might be possibly achieved when using BioNJ [7] rather than PhyML [4] algorithm or a tool like MUSCLE [3] or T-Coffee [6] rather than ClustalW [8] for the MSA. Finally, the number of bootstrapped sample might be increased to grasp a better accuracy.

By calculating the Shannon entropy of every alignment position in the MSA building up the species tree (Figure 3), it was possible to lately estimate the per-site sequence conservation of the 16S rRNA protein for the species considered. The most conserved positions were highlighted in light blue and from the plot in Figure 4 it is observed a clear conservation pattern, leading to the detection of functional and non-functional regions. The analysis might be followed-up by the actual functional annotation of these areas.

5 Data Availability

The input and output files used in the analyses, along with the Python scripts and the resulting plots, can be accessed on the following GitHub repository.

References

[1] L.-T. K. Alföldi J. Comparative genomics as a tool to understand evolution and disease. *Genome Res.*, 2013.

[2] M. W. M. E. L. D. Altschul SF, Gish W. Basic local alignment search tool. *J Mol Biol.*, 1990.

[3] R. Edgar. Muscle: a multiple sequence alignment method with reduced time and space complexity. *BMC Bioinformatics.*, 2004.

[4] D. J. G. O. Guindon S, Delsuc F. Estimating maximum likelihood phylogenies with phyml. *Methods Mol Biol.*, 2009.

[5] A. M. Jacobs HJ. On the taxonomy of the burmese python, python molurus bivittatus kuhl, 1820, specifically on the sulawesi population. *Sauria*, 2009.

[6] C. Notredame, D. G. Higgins, and J. Heringa. T-coffee: a novel method for fast and accurate multiple sequence alignment. *Journal of Molecular Biology*, 2000.

[7] G. O. Bionj: an improved version of the nj algorithm based on a simple model of sequence data. *Mol Biol Evol.*, 1997 Jul.

[8] G. T. Thompson JD, Higgins DG. Clustal w: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Res.*, 1994.

[9] K.-K. A. Wägele H. The taxonomist - an endangered race. a practical proposal for its survival. *Frontiers in Zoology*, October 2011.

6 Supplementary Materials

6.1 Python script A: Best Bi-directional Hits

```python
"""
@author: Antonietta Salerno
@date: 4/1/2022
@title: Comparative Genomics Assignment - Best Bi-directional Hits

Note: To run the following lines, BioPython installation is needed.
On the terminal run the following command: conda install biopython
"""
# Import packages
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from Bio.Blast.Applications import NcbiblastpCommandline as cmd
import warnings
warnings.filterwarnings('ignore')

# 1. Run the BLAST query in the forward and reverse direction taking as input the
    # FASTA files of the proteomes and returning as output a tab-separated file.
```

```python
blastp_fwd = cmd(query="latimeria.fasta", db="python.fasta", outfmt=6,
                 out="lat_pyt.tab", max_target_seqs=1)
blastp_bwd = cmd(query="python.fasta", db="latimeria.fasta", outfmt=6,
                 out="pyt_lat.tab", max_target_seqs=1)
stderr, stdout = blastp_fwd()
stderr, stdout = blastp_bwd()

# 2. Read the files produced as pandas dataframes
pyt_lat = pd.read_csv("pyt_lat.tab", sep="\t", header=None)
lat_pyt = pd.read_csv("lat_pyt.tab", sep="\t", header=None)

# 3. Remove the columns not needed
pyt_lat = pyt_lat.drop(columns=[5,6,7,8,11])
lat_pyt = lat_pyt.drop(columns=[5,6,7,8,11])

# 4. Add headers to forward and reverse results dataframes
headers = ["query", "subject", "identity", "coverage", "qlength",
           "bitscore", "E-value"]
pyt_lat.columns = headers
lat_pyt.columns = headers

# 5. Calculate the normalised bitscore and add the column to the datasets
pyt_lat['norm_bitscore'] = pyt_lat.bitscore.div(pyt_lat.qlength)
pyt_lat.loc[~np.isfinite(pyt_lat['norm_bitscore']), 'norm_bitscore'] = 0
    # Solves division by zero problem
lat_pyt['norm_bitscore'] = lat_pyt.bitscore.div(lat_pyt.qlength)
lat_pyt.loc[~np.isfinite(lat_pyt['norm_bitscore']), 'norm_bitscore'] = 0
    # Solves division by zero problem

# 6. Merge forward and reverse results by inner join, while keeping norm_bitscore
    # and E-value for both directions
bbh = pd.merge(pyt_lat, lat_pyt[['query', 'subject',"norm_bitscore", "E-value"]],
               left_on='subject', right_on='query',
               how='inner')

# 7. Group duplicate RBH rows, taking the maximum value in each column
bbh = bbh.groupby(['query_x', 'subject_x']).max()

# 8. Sort the values by the normalised bitscore of one of the datasets and save
    # into a .csv file = 90 hits
bbh.sort_values(by='norm_bitscore_x', ascending=False)
bbh.to_csv("BBH_python_latimeria.csv")

# 9. Explore the distribution of normalised bitscores
f, axes = plt.subplots(1, 2, figsize=(14, 7), sharex=True)
sns.despine(left=True)
sns.distplot(bbh.norm_bitscore_x, color="b", ax=axes[0],
                axlabel="forward normalised bitscores")
sns.distplot(bbh.norm_bitscore_y, color="g", ax=axes[1],
                axlabel="reverse normalised bitscores")
plt.savefig("density_norm_bitscore.png")

# 10. Filter by significance (E-value) based on the exploratory plots = 71 hits
bbh = bbh.drop(bbh[bbh["E-value_x"] > 0.001].index)
bbh = bbh.drop(bbh[bbh["E-value_y"] > 0.001].index)

# 11. Filter by quality of the alignment (normalised bitscore)
    # based on the exploratory plots = 2 hits
bbh = bbh.drop(bbh[bbh["norm_bitscore_x"] < 300].index)
bbh = bbh.drop(bbh[bbh["norm_bitscore_y"] < 40].index)
bbh.to_csv("BBH_python_latimeria_orthologs.csv")
print(bbh.norm_bitscore_x)
```

6.2 Python script B: Conserved regions

```
"""
@author: Antonietta Salerno
@date: 8/1/2022
@title: Comparative Genomics Assignment - Functional Conservation in MSA
Note: To run the following lines, BioPython installation is needed.
On the terminal run the following command: conda install biopython
"""

# Import packages
from Bio import AlignIO
import pandas as pd
import math
import scipy.stats as st
import matplotlib.pyplot as plt
from itertools import groupby
from operator import itemgetter

# Retrieve the ClustalW alignment file and parse it
filename = "species_tree.aln"
format = "clustal"
alignment = AlignIO.read(filename, format)

# Retrieve the MSA of a single column
def get_column(position):
    return [sequence[position] for sequence in alignment]

# Calculate the Shannon entropy for a single column (measure of conservation)
def get_entropy(column):
    entropy = 0
```

```python
    if column.count('-')>20:
        entropy=+9
    entropy+= st.entropy(pd.Series(column).value_counts())
    return entropy

# Create a dictionary containing the Shannon entropy for every position in the MSA
sites = {}
for position in range(0, alignment.get_alignment_length()):
    col = get_column(position)
    sites[position] = get_entropy(col)

# Order the dictionary (most conserved sequences first: the smaller the entropy
    # the higher the conservation) and save to csv
sites_ordered = sorted(sites.items(), key=lambda x: x[1])
sites_ordered = pd.DataFrame(sites_ordered, columns = ("position", "entropy"))
sites_ordered.to_csv("sites_ordered_entropy.csv")

# Check whether the site is actually conserved
most_conserved = sites_ordered["position"][0]
print("The most conserved position is ", most_conserved,
      "and has the following amino acid composition: ",
      ' '.join(get_column(int(most_conserved))))

# Getting regions of highest conservation
minimum = sites_ordered["entropy"] == sites_ordered["entropy"].min()
get_regions = list(sites_ordered[minimum]["position"])
ranges = []
for k, g in groupby(enumerate(get_regions), lambda ix:ix[0]-ix[1]):
    group = list(map(itemgetter(1), g))
    ranges.append((group[0], group[-1]))

# Plot sequence conservation regions by site for the whole MSA
fig = plt.figure(figsize=(10,5))
ax = fig.add_subplot(111)
x, y = zip(*sites.items()) # unpack a list of pairs into two tuple
plt.plot(x,y)
for i in range(0, len(ranges)):
    plt.axvspan(ranges[i][0], ranges[i][1], color='lightblue', alpha=0.5)

plt.title("Per-site sequence conservation of MSA")
plt.xlabel("Sequence position")
plt.ylabel("Shannon entropy")
plt.show()
fig.savefig("sites_conservation.png", dpi = 300)
```