Look for co-evolution

Elena Montenegro, Sara González, Sofía Cerdá, Fernando Freire

December 26, 2018

Contents

1 Look for co-evolution Streptococcus-Lactobacillus

Co-evolution is everywhere. Because no species can be considered totally isolated from the others. And not only on a biological level, we perceive it in the lines of change in our societies, in political, scientific, religious ideas and even in the evolution of software engineering.

The difficulty is to know how much co-evolution is due to the interaction between a pair of species, because the tandem can not be totally isolated from the rest of the universe either. No doubt the problem can not be solved only from the bioinformatic perspective, it is also necessary the contribution of other branches of knowledge such as microbiology, physics, mathematics and computer theory.

Our approach consists in comparing the rates of protein evolutionary change between strains of Streptococcus and Lactobacillus linked by symbiotic pathways, and the rates between the rest of species of each genus.

We postulate that the proteins that reflects a rate change with more significance, i.e., more speedy or more slowly that intra genus rates, are proteins that could be influenced by the symbiotic environment.

To do so we need to compute separately both species of symbiotic tandem. The detailed steps are:

1) Obtain all the proteome from this four groups of species:

- Genus streptococcus
- Strains of streptococcus termophilus.
- Genus lactobacillus
- Strains of lactobacillus bulgaricus.

2) For each protein of each group we obtain the phylogenetic tree and compute the mean branch length. We suppose that we have ultrametricicy or almost ultrametricity. It can be not exact but it could serve as a reference. We use the clulstalw multialignment method, but other methods as T-COFFEE or MUSCLE could be used too. The branch length is a measure of the substitution rate.

3) For each protein of the first an second group, we calculate the ration between branch length of termophilus and branch length of protein in genus. And also we compute the overall mean of all ratios. Finally we select the proteins with the most extreme values (80% percentile, two tailed).

4) We do the same with lactobacillus (third an fourth groups). At this stage we have two sets of proteins.

5) We obtain the biologic pathways of each of the sets of proteins.

6) The proteins involved in the same pathways from lactobacillus ans streptoccocus are the target proteins influenced by the coevolution between the two species. It will be necessary a later microbiological study that confirms this expectancies and dive deeply in the details of this coevolution.

1.1 Obtain data from servers

Script 1.1.1 (python)

```
1 import requests, sys
```

Script 1.1.2 (python)

```python
def load_taxa(scientific_prefix):
    """
    """
    requestURL = "https://www.ebi.ac.uk/proteins/api/taxonomy/name/" + scientific_prefix +\
                 "%20?pageNumber=1&pageSize=100&searchType=STARTSWITH&fieldName=SCIENTIFICNAM
E"

    r = requests.get(requestURL, headers={ "Accept" : "application/json"})

    if not r.ok:
      r.raise_for_status()
      sys.exit()

    jsonBody = json.loads(r.text)
    taxa = []
    names = []
    for taxonomy in jsonBody["taxonomies"]:
        print(taxonomy['taxonomyId'])
        print(taxonomy['scientificName'])
        taxa.append(taxonomy['taxonomyId'])
        names.append(taxonomy['scientificName'])
    return taxa, names

termophilus_taxa,  termophilus_names = load_taxa("Streptococcus thermophilus")
streptococcus_taxa,  streptococcus_names = load_taxa("Streptococcus")
```

Output

```
264199
Streptococcus thermophilus (strain ATCC BAA-250 / LMG 18311)
299768
Streptococcus thermophilus (strain CNRZ 1066)
322159
Streptococcus thermophilus (strain ATCC BAA-491 / LMD-9)
767463
Streptococcus thermophilus (strain ND03)
1042404
Streptococcus thermophilus CNCM I-1630
1051074
Streptococcus thermophilus JIM 8232
1073569
Streptococcus thermophilus MTCC 5460
1073570
Streptococcus thermophilus MTCC 5461
1091038
Streptococcus thermophilus DSM 20617
1187956
Streptococcus thermophilus MN-ZLW-002
1263110
Streptococcus thermophilus CAG:236
```

```
1268061
Streptococcus thermophilus DGCC 7710
1408178
Streptococcus thermophilus ASCC 1275
1415776
Streptococcus thermophilus TH1435
1423145
Streptococcus thermophilus TH1436
1433288
Streptococcus thermophilus MTH17CL396
1433289
Streptococcus thermophilus M17PTZA496
1435972
Streptococcus thermophilus TH985
1435974
Streptococcus thermophilus TH982
1435981
Streptococcus thermophilus 1F8CT
1436725
Streptococcus thermophilus TH1477
1302
Streptococcus gordonii
1303
Streptococcus oralis
1304
Streptococcus salivarius
1305
Streptococcus sanguinis
1306
Streptococcus sp.
1307
Streptococcus suis
1308
Streptococcus thermophilus
1309
Streptococcus mutans
1310
Streptococcus sobrinus
1311
Streptococcus agalactiae
1313
Streptococcus pneumoniae
1314
Streptococcus pyogenes
1317
Streptococcus downei
1318
Streptococcus parasanguinis
1319
Streptococcus sp. 'group B'
1320
Streptococcus sp. group G
```

```
1324
Streptococcus sp. G148
1325
Streptococcus sp. GX7805
1326
Streptococcus acidominimus
1328
Streptococcus anginosus
1329
Streptococcus canis
1332
Streptococcus criae
1333
Streptococcus criceti
1334
Streptococcus dysgalactiae
1335
Streptococcus equinus
1336
Streptococcus equi
1337
Streptococcus hyointestinalis
1338
Streptococcus intermedius
1339
Streptococcus macacae
1340
Streptococcus porcinus
1341
Streptococcus ratti
1343
Streptococcus vestibularis
1345
Streptococcus ferus
1346
Streptococcus iniae
1348
Streptococcus parauberis
1349
Streptococcus uberis
10728
Streptococcus pneumoniae phage HB-3
10747
Streptococcus phage Cp-1
10748
Streptococcus phage Cp-7
10749
Streptococcus phage Cp-9
12344
Streptococcus phage Cp-5
12366
Streptococcus pyogenes phage H4489A
```

```
12402
Streptococcus phage EJ-1
28037
Streptococcus mitis
29389
Streptococcus alactolyticus
29390
Streptococcus gordonii str. Challis
33040
Streptococcus milleri
33972
Streptococcus sp. 'group C'
35344
Streptococcus pyogenes phage T12
36470
Streptococcus sp. 'group A'
39425
Streptococcus phage T270
40041
Streptococcus equi subsp. zooepidemicus
40287
Streptococcus mutans serotype C
45634
Streptococcus cristatus
53354
Streptococcus gallolyticus
55085
Streptococcus thoraltensis
59310
Streptococcus macedonicus
64186
Streptococcus virus Sfi21
68891
Streptococcus peroris
68892
Streptococcus infantis
69017
Streptococcus sp. (strain 19909)
72638
Streptococcus virus Sfi19
73422
Streptococcus phage TP-J34
73492
Streptococcus pyogenes phage
74382
Streptococcus phage SFi18
76860
Streptococcus constellatus
78535
Streptococcus viridans
78541
Streptococcus virus Sfi11
```

```
82269
Streptococcus sp. 28D
82348
Streptococcus pluranimalium
82806
Streptococcus ovis
83545
Streptococcus sp. KN1
83546
Streptococcus sp. KN2
83547
Streptococcus sp. KN3
83549
Streptococcus sp. TW1
85154
Streptococcus phage 01205
86065
Streptococcus pyogenes phage H10403
90410
Streptococcus virus DT1
99822
Streptococcus dysgalactiae subsp. dysgalactiae
102143
Streptococcus phage S3b
102144
Streptococcus phage S92
102145
Streptococcus phage ST3
102146
Streptococcus phage ST64
102147
Streptococcus phage Sfi16A
102150
Streptococcus phage J1
102684
Streptococcus infantarius
102886
Streptococcus didelphis
104215
Streptococcus sp. 1400-98
112023
Streptococcus virus 7201
113107
Streptococcus australis
114652
Streptococcus orisratti
116154
Streptococcus sp. Z1227
116155
Streptococcus sp. Z12
116156
Streptococcus sp. Z89
```

```
118670
Streptococcus sp. PSH2
118671
Streptococcus sp. PSH1a
118672
Streptococcus sp. PSH1b
119224
Streptococcus phocae
119602
Streptococcus dysgalactiae subsp. equisimilis
119603
Streptococcus dysgalactiae group
```

Script 1.1.3 (python)

```python
def load_proteome(taxids, size=10, protein=["LDH"]):
    """
    """
    taxids_str = ",".join(str(x) for x in taxids)
    protein_str = ",".join(x for x in protein)
    print(taxids_str)
    requestURL = "https://www.ebi.ac.uk/proteins/api/proteins?offset=0&size=" + str(size) + "&taxid=" +\
                  taxids_str + "&reviewed=false"
    if protein != []:
        requestURL += "&gene=" + protein_str
    print(requestURL)
    r = requests.get(requestURL, headers={ "Accept" : "text/x-fasta"})

    if not r.ok:
      r.raise_for_status()
      sys.exit()

    proteome = r.text
    return proteome

termophilus_taxids = termophilus_taxa[0:19]
streptococcus_taxids = streptococcus_taxa[0:19]
print(streptococcus_taxids)
print(termophilus_taxids)

streptococcus_proteome = load_proteome(streptococcus_taxids, -1, protein = ["LDH", "CAS2", "CAS3"])
termophilus_proteome = load_proteome(termophilus_taxids, -1, protein = ["LDH", "CAS2", "CAS3"])
```

Output

```
[1302, 1303, 1304, 1305, 1306, 1307, 1308, 1309, 1310, 1311, 1313, 1314, 1317, 1318, 1319, 1320, 1324, 1325, 1326]
```

```
[264199, 299768, 322159, 767463, 1042404, 1051074, 1073569, 1073570, 1091038, 1187956,
↪  1263110, 1268061, 1408178, 1415776, 1423145, 1433288, 1433289, 1435972, 1435974]
1302,1303,1304,1305,1306,1307,1308,1309,1310,1311,1313,1314,1317,1318,1319,1320,1324,1325,1326
https://www.ebi.ac.uk/proteins/api/proteins?offset=0&size=-1&taxid=1302,1303,1304,1305,1306,1
↪  307,1308,1309,1310,1311,1313,1314,1317,1318,1319,1320,1324,1325,1326&reviewed=false&gene=
↪  LDH,CAS2,CAS3
264199,299768,322159,767463,1042404,1051074,1073569,1073570,1091038,1187956,1263110,1268061,1
↪  408178,1415776,1423145,1433288,1433289,1435972,1435974
https://www.ebi.ac.uk/proteins/api/proteins?offset=0&size=-1&taxid=264199,299768,322159,76746
↪  3,1042404,1051074,1073569,1073570,1091038,1187956,1263110,1268061,1408178,1415776,1423145
↪  ,1433288,1433289,1435972,1435974&reviewed=false&gene=LDH,CAS2,CAS3
```

1.2 Compute substitution rates

Script 1.2.1 (python)

```python
import re

def proteome2dict(proteome_fasta):
    """
    Returns a dict with keys protein accession and values the list of fasta format for all
    taxids
    This is the basis for clustalw alignments and tree generation
    """
    proteome = {}
    key_found = False
    for line in proteome_fasta.splitlines():
        if len(line) > 0:
            if line[0] == ">":
                if key_found:
                    if key in proteome:
                        proteome[key].append(seq)
                    else:
                        proteome[key] = [seq]
                key_found = True
                search_gene_name = re.search('GN=(\w*)', line)
                key = search_gene_name.group(1).upper()
                #print(key)
                seq = line + '\n'
            elif key_found:
                seq += line + '\n'
    if key_found:
        if key in proteome:
            proteome[key].append(seq)
        else:
            proteome[key] = [seq]
    return proteome
```

Script 1.2.2 (python)

```python
# Phylo tree with clustalw. We need to measure the substitution rate.
import matplotlib
import matplotlib.pyplot as plt
%matplotlib inline
from Bio import Phylo
from io import StringIO
import os
from Bio.Align.Applications import ClustalwCommandline
CLUSTALW = r"./clustalw2"
assert os.path.isfile(CLUSTALW), "Clustal W executable missing"
plt.rcParams["figure.figsize"] = (20,30)
matplotlib.rc('font', size=12)

def compute_mean_subst_rate(proteome, verbose=False, show_tree=False):
    """
    """
    clustalw_cline = ClustalwCommandline(CLUSTALW, infile=proteome + ".fasta")
    stdout, stderr = clustalw_cline()
    f = open(proteome + ".dnd", "r")
    s_tree = f.read()
    f.close()
    #print(s_tree)
    branch_len = 0
    num_branches = 0
    search_branch_length = re.findall(':([-.0123456789]*)', s_tree)
    for branch_length in search_branch_length:
        #print(branch_length)
        if branch_length != "0.00000":
            branch_len += float(branch_length)
            num_branches += 1
    if verbose: print(branch_len, num_branches, branch_len/num_branches)
    if show_tree:
        tree = Phylo.read(proteome + ".dnd", "newick")
        Phylo.draw(tree)
    return branch_len/num_branches

def compute_subst_rates(proteome, proteome_name, verbose=False):
    """
    """
    subst_rates = {}
    for protein in proteome.keys():
        if verbose: print(protein)
        protein_sequence = ""
        # Only for proteins with enough sequences to make a tree
        if len(proteome[protein]) >= 3:
            for sequence in proteome[protein]:
                protein_sequence += sequence
            fasta_file_name = proteome_name + "_" + protein
            f = open(fasta_file_name + ".fasta", "w")
            if verbose: print(protein_sequence)
            f.write(protein_sequence)
```

```
            f.close()
            mean_subst_rate = compute_mean_subst_rate(fasta_file_name)
            subst_rates[protein] = mean_subst_rate
    return subst_rates


# Proteomes in fasta to dictionnaries
proteome_termophilus = proteome2dict(termophilus_proteome)
proteome_streptococcus = proteome2dict(streptococcus_proteome)

# Compute branch lengths
subst_rates_groups = {}
subst_rates_groups["termophilus"] = compute_subst_rates(proteome_termophilus, "termophilus",
↪  False)
subst_rates_groups["streptococcus"] = compute_subst_rates(proteome_streptococcus,
↪  "streptococcus", False)
print(subst_rates_groups)

# Compute branch ratios
# Compute mean of branch ratios and standard deviation
# Obtain the most extreme values. These are the proteins that could have been a slowdown or
↪  from his initial state
```

Output

```
{'termophilus': {'CAS2': 0.10117692307692308, 'LDH': 0.007116666666666667}, 'streptococcus':
↪  {'CAS2': 0.05127011363636366, 'LDHA': 0.09308272727272726, 'LDH_2': 0.1056288888888889,
↪  'LDH_1': 0.12399615384615387, 'LDHA_1': 0.14062904761904763, 'LDHA_3':
↪  0.1900181818181818, 'LDH': 0.038460561797752806, 'LDHA_2': 0.13067238095238096, 'LDHD':
↪  0.04888}}
```

1.3 Selection of first set of target proteins

At this step is necessary to calculate the ratios, the means of the ratios and select the more extreme between them (two tailed percentile 80%).

1.4 Obtain protein pathways from servers.

At this stage we obtain the pathways for proteins.

1.5 Selection of final set of target proteins

At this stage we select the proteins that belong tho the same pathways on Lactobacillus and Streptoccus. These are the proteins that could be affected by co-evolucion.

2 Generate document outputs.

Script 2.0.1 (text)

```
%%bash
#cd /Users/nandoide/Desktop/uni/STRBI.practical
jupyter nbconvert --to=latex --template=~/report.tplx coevolution.ipynb 1> /dev/null
pdflatex -shell-escape coevolution 1> /dev/null
```

Output

```
[NbConvertApp] Converting notebook coevolution.ipynb to latex
[NbConvertApp] Writing 41101 bytes to coevolution.tex
```