In [2]:
```
import altair as alt
from altair_saver import save
from IPython.display import Image
import numpy as np
import os
import pandas as pd
import re
from scipy import stats
```

In [19]:
```
def isNaN(num):
    return num != num
```

Marinobacter Metapangenome

Tutorial following http://merenlab.org/2019/03/14/ncbi-genome-download-magic/ (http://merenlab.org/2019/03/14/ncbi-genome-download-magic/)

Download *Marinobacter* genomes from NCBI.

In [ ]:
```
!ncbi-genome-download bacteria --assembly-level all --genus Marinobacter
--metadata metadata.txt
```

Process NCBI genomes into anvio

In [5]:
```
!anvi-script-process-genbank-metadata -m metadata.txt --output-dir NCBI_
genomes --output-fasta-txt Marinobacter.txt --exclude-gene-calls-from-fa
sta-txt
```

```
Input metadata file ..........................: metadata.txt
Output directory .............................: /Users/tito_miniconda/J
OYE_LAB_ANVIO_PROJECTS/SK_BACKUP/p28_pangenomes/Marinobacter/NCBI_genom
es
Num entries in metadata ......................: 113
Output FASTA descriptor ......................: Marinobacter.txt
9m ETA: 0s
```

Supporting code for the paper:

Colwellia and Marinobacter metapangenomes reveal species-specific responses to oil and dispersant exposure in deepsea microbial communities

Tito David Peña-Montenegro, Sara Kleindienst, Andrew E. Allen, A. Murat Eren, John P. McCrow, Juan David Sánchez-Calderón, Jonathan Arnold, Samantha B. Joye

Preprint available at: https://doi.org/10.1101/2020.09.28.317438

In [6]:
```
!anvi-run-workflow -w contigs --get-default-config default_config.txt
```

```
WARNING
================================================
If you publish results from this workflow, please do not forget to cite
snakemake (doi:10.1093/bioinformatics/bts480)


WARNING
================================================
We are initiating parameters for the contigs workflow

Default config file ............................: Stored for workflow 'co
ntigs' as 'default_config.txt'.
```

In [7]:
```
!cat default_config.txt
```

```
{
    "fasta_txt": "Marinobacter.txt",
    "anvi_gen_contigs_database": {
        "--project-name": "{group}",
        "--description": "",
        "--skip-gene-calling": "",
        "--external-gene-calls": "",
        "--ignore-internal-stop-codons": "",
        "--skip-mindful-splitting": "",
        "--contigs-fasta": "",
        "--split-length": "",
        "--kmer-size": "",
        "--prodigal-translation-table": "",
        "threads": 6
    },
    "centrifuge": {
        "threads": 6,
        "run": "",
        "db": ""
    },
    "anvi_run_hmms": {
        "run": true,
        "threads": 6,
        "--installed-hmm-profile": "",
        "--hmm-profile-dir": ""
    },
    "anvi_run_ncbi_cogs": {
        "run": true,
        "threads": 6,
        "--cog-data-dir": "",
        "--sensitive": "",
        "--temporary-dir-path": "",
        "--search-with": ""
    },
    "anvi_script_reformat_fasta": {
        "run": true,
        "--keep-ids": "",
        "--exclude-ids": "",
        "--min-len": "",
        "threads": 6
    },
    "emapper": {
        "--database": "bact",
        "--usemem": true,
        "--override": true,
        "path_to_emapper_dir": "",
        "threads": 6
    },
    "anvi_script_run_eggnog_mapper": {
        "--use-version": "0.12.6",
        "run": "",
        "--cog-data-dir": "",
        "--drop-previous-annotations": "",
        "threads": 6
    },
    "gen_external_genome_file": {
        "threads": 6
```

```
        },
        "export_gene_calls_for_centrifuge": {
            "threads": 6
        },
        "anvi_import_taxonomy": {
            "threads": 6
        },
        "annotate_contigs_database": {
            "threads": 6
        },
        "anvi_get_sequences_for_gene_calls": {
            "threads": 6
        },
        "gunzip_fasta": {
            "threads": 6
        },
        "reformat_external_gene_calls_table": {
            "threads": 6
        },
        "reformat_external_functions": {
            "threads": 6
        },
        "import_external_functions": {
            "threads": 6
        },
        "anvi_run_pfams": {
            "run": "",
            "--pfam-data-dir": "",
            "threads": 6
        },
        "output_dirs": {
            "FASTA_DIR": "01_FASTA",
            "CONTIGS_DIR": "02_CONTIGS",
            "LOGS_DIR": "00_LOGS"
        },
        "max_threads": 6
    }
```

Run this, if anvio is newly installed.

```
anvi-setup-scg-databases
```

and

```
anvi-setup-ncbi-cogs
```

Then, the following line was run.

```
anvi-run-workflow -w contigs -c default_config.txt --additional-params --jobs 6 --
resources nodes=6
```

Now creating the database of Marinobacter genomes.

In [11]:
```
!anvi-gen-genomes-storage -e external_genomes.txt -o MARINOBACTER_GENOME
S.db
```

```
ETA: None
WARNING
===============================================
Good news! Anvi'o found all these functions that are common to all of y
our
genomes and will use them for downstream analyses and is very proud of
 you:
'COG_CATEGORY, COG_FUNCTION'.

Internal genomes ..............................: 0 have been initialize
d.
External genomes ..............................: 113 found.
9m ETA: 0s

[0m
JUST FYI
===============================================
Some of your genomes had gene calls identified by gene callers other th
an the
anvi'o default, 'prodigal', and will not be processed. Use the `--debug
` flag if
this sounds important and you would like to see more of this message.


[0m
* M_adhaerens_GCF_001717765 is stored with 3,896 genes (15 of which wer
e partial)
* M_algicola_DG893_GCF_000170835 is stored with 4,106 genes (72 of whic
h were partial)                                 [0m
* M_antarcticus_GCF_900142385 is stored with 3,440 genes (10 of which w
ere partial)                                    [0m
* M_aromaticivorans_GCF_002806975 is stored with 3,778 genes (36 of whi
ch were partial)                                [0m
* M_bohaiensis_GCF_003258515 is stored with 4,291 genes (7 of which wer
e partial)                                      [0m
* M_confluentis_GCF_004785685 is stored with 3,534 genes (110 of which
 were partial)                                   [0m
* M_confluentis_GCF_008795935 is stored with 3,432 genes (16 of which w
ere partial)                                    [0m
* M_daepoensis_DSM_16072_GCF_000421165 is stored with 3,515 genes (18 o
f which were partial)                           [0m
* M_daqiaonensis_GCF_900115285 is stored with 3,558 genes (6 of which w
ere partial)                                    [0m
* M_excellens_HL_55_GCF_000934705 is stored with 3,665 genes (1 of whic
h were partial)                                 [0m
* M_excellens_LAMA_842_GCF_001574445 is stored with 4,046 genes (44 of
 which were partial)                             [0m
* M_flavimaris_GCF_002933295 is stored with 4,148 genes (51 of which we
re partial)                                     [0m
* M_flavimaris_GCF_003363485 is stored with 4,120 genes (25 of which we
re partial)                                     [0m
* M_fonticola_GCF_008122265 is stored with 4,005 genes (1 of which were
partial)                                       [0m
* M_fuscus_GCF_003007675 is stored with 3,868 genes (95 of which were p
artial)                                         [0m
* M_gudaonensis_GCF_900115175 is stored with 3,459 genes (4 of which we
```

```
re partial)                                    [0m
* M_guineae_GCF_002744735 is stored with 4,153 genes (20 of which were
 partial)                                      [0m
* M_halophilus_GCF_003007685 is stored with 3,564 genes (10 of which we
re partial)                                    [0m
* M_halotolerans_GCF_008795985 is stored with 3,471 genes (1 of which w
ere partial)                                   [0m
* M_hydrocarbonoclasticus_ATCC_49840_GCF_000284615 is stored with 3,655
genes (1 of which were partial)                [0m
* M_hydrocarbonoclasticus_GCF_001895225 is stored with 3,929 genes (13
 of which were partial)                        [0m
* M_hydrocarbonoclasticus_GCF_003315555 is stored with 4,007 genes (22
 of which were partial)                        [0m
* M_hydrocarbonoclasticus_GCF_003337515 is stored with 4,100 genes (42
 of which were partial)                        [0m
* M_hydrocarbonoclasticus_GCF_003337655 is stored with 4,005 genes (21
 of which were partial)                        [0m
* M_hydrocarbonoclasticus_GCF_003634635 is stored with 3,697 genes (6 o
f which were partial)                          [0m
* M_hydrocarbonoclasticus_GCF_006516615 is stored with 3,679 genes (14
 of which were partial)                        [0m
* M_hydrocarbonoclasticus_GCF_009650625 is stored with 3,850 genes (42
 of which were partial)                        [0m
* M_hydrocarbonoclasticus_VT8_GCF_000015365 is stored with 4,453 genes
 (0 of which were partial)                     [0m
* M_lipolyticus_BF04_CF_4_GCF_000372805 is stored with 3,684 genes (166
of which were partial)                         [0m
* M_lipolyticus_SM19_GCF_000397065 is stored with 3,641 genes (14 of wh
ich were partial)                              [0m
* M_litoralis_GCF_003336705 is stored with 3,148 genes (3 of which were
partial)                                       [0m
* M_lutaoensis_GCF_001981305 is stored with 3,507 genes (27 of which we
re partial)                                    [0m
* M_manganoxydans_MnI7_9_GCF_000235625 is stored with 4,219 genes (92 o
f which were partial)                          [0m
* M_maritimus_GCF_007671675 is stored with 3,974 genes (8 of which were
partial)                                       [0m
* M_mobilis_GCF_900106945 is stored with 3,614 genes (22 of which were
 partial)                                      [0m
* M_nanhaiticus_D15_8W_GCF_000364845 is stored with 4,834 genes (6 of w
hich were partial)                             [0m
* M_nitratireducens_GCF_000708045 is stored with 3,512 genes (16 of whi
ch were partial)                               [0m
* M_pelagius_GCF_003315345 is stored with 3,830 genes (24 of which were
partial)                                       [0m
* M_pelagius_GCF_900114925 is stored with 3,515 genes (15 of which were
partial)                                       [0m
* M_persicus_GCF_002934305 is stored with 3,313 genes (68 of which were
partial)                                       [0m
* M_persicus_GCF_002934325 is stored with 3,312 genes (68 of which were
partial)                                       [0m
* M_persicus_GCF_002934485 is stored with 3,319 genes (75 of which were
partial)                                       [0m
* M_persicus_GCF_900114155 is stored with 2,945 genes (21 of which were
partial)                                       [0m
* M_piscensis_GCF_007671655 is stored with 3,022 genes (6 of which were
partial)                                       [0m
```

```
* M_profundi_GCF_002744715 is stored with 3,632 genes (36 of which were
partial)                                    [0m
* M_psychrophilus_GCF_001043175 is stored with 3,597 genes (0 of which
 were partial)                               [0m
* M_salarius_GCF_000831005 is stored with 5,369 genes (1 of which were
 partial)                                    [0m
* M_salarius_GCF_002116735 is stored with 4,272 genes (3 of which were
 partial)                                    [0m
* M_salarius_GCF_003986605 is stored with 4,168 genes (2 of which were
 partial)                                    [0m
* M_salexigens_GCF_002806945 is stored with 3,460 genes (24 of which we
re partial)                                 [0m
* M_salinus_GCF_001854125 is stored with 3,780 genes (2 of which were p
artial)                                     [0m
* M_salsuginis_GCF_009617755 is stored with 3,782 genes (6 of which wer
e partial)                                  [0m
* M_salsuginis_GCF_009617795 is stored with 4,349 genes (78 of which we
re partial)                                 [0m
* M_salsuginis_SD_14B_GCF_004936695 is stored with 8,035 genes (1,052 o
f which were partial)                       [0m
* M_santoriniensis_NKSG1_GCF_000347775 is stored with 3,729 genes (34 o
f which were partial)                       [0m
* M_segnicrescens_GCF_900111555 is stored with 3,934 genes (110 of whic
h were partial)                             [0m
* M_shengliensis_GCF_003007715 is stored with 3,780 genes (30 of which
 were partial)                               [0m
* M_similis_GCF_000830985 is stored with 4,590 genes (0 of which were p
artial)                                     [0m
* M_sp__3_2_GCF_003751355 is stored with 4,084 genes (8 of which were p
artial)                                     [0m
* M_sp__AC_23_GCF_001858325 is stored with 4,440 genes (125 of which we
re partial)                                 [0m
* M_sp__ANT_B65_GCF_002407605 is stored with 3,822 genes (8 of which we
re partial)                                 [0m
* M_sp__Arc7_DN_1_GCF_003441595 is stored with 3,958 genes (2 of which
 were partial)                               [0m
* M_sp__BSs20148_GCF_000283275 is stored with 3,694 genes (1 of which w
ere partial)                                [0m
* M_sp__BW6_GCF_008107725 is stored with 3,914 genes (104 of which were
partial)                                    [0m
* M_sp__C18_GCF_001924925 is stored with 4,591 genes (20 of which were
 partial)                                    [0m
* M_sp__C1S70_GCF_000475355 is stored with 3,848 genes (80 of which wer
e partial)                                  [0m
* M_sp__CLL7_20_GCF_009193265 is stored with 3,977 genes (14 of which w
ere partial)                                [0m
* M_sp__CP1_GCF_001266795 is stored with 4,395 genes (2 of which were p
artial)                                     [0m
* M_sp__DS40M8_GCF_004936715 is stored with 4,610 genes (1,224 of which
were partial)                               [0m
* M_sp__DSM_26291_GCF_900114695 is stored with 4,154 genes (57 of which
were partial)                               [0m
* M_sp__DSM_26671_GCF_900112835 is stored with 4,458 genes (107 of whic
h were partial)                             [0m
* M_sp__EC_HK377_GCF_902498775 is stored with 4,022 genes (5 of which w
ere partial)                                [0m
* M_sp__ELB17_GCF_000169375 is stored with 4,653 genes (90 of which wer
```

e partial) [0m
* M_sp__EN3_GCF_000475315 is stored with 3,722 genes (79 of which were
 partial) [0m
* M_sp__ES_1_GCF_000475255 is stored with 3,360 genes (85 of which were
partial) [0m
* M_sp__EVN1_GCF_000475375 is stored with 4,017 genes (67 of which were
partial) [0m
* M_sp__EhC06_GCF_001650915 is stored with 4,235 genes (23 of which wer
e partial) [0m
* M_sp__EhN04_GCF_001650765 is stored with 4,236 genes (26 of which wer
e partial) [0m
* M_sp__F3R11_GCF_003318275 is stored with 3,051 genes (20 of which wer
e partial) [0m
* M_sp__HL_58_GCF_000686085 is stored with 3,895 genes (2 of which were
partial) [0m
* M_sp__JB02H27_GCF_008795955 is stored with 4,519 genes (30 of which w
ere partial) [0m
* M_sp__JH2_GCF_004353225 is stored with 3,340 genes (4 of which were p
artial) [0m
* M_sp__LQ44_GCF_001447155 is stored with 4,066 genes (0 of which were
 partial) [0m
* M_sp__LV10MA510_1_GCF_002563885 is stored with 4,245 genes (1 of whic
h were partial) [0m
* M_sp__LV10R510_11A_GCF_900215155 is stored with 4,279 genes (1 of whi
ch were partial) [0m
* M_sp__LV10R510_8_GCF_002846515 is stored with 4,244 genes (0 of which
were partial) [0m
* M_sp__LV10R520_4_GCF_002563815 is stored with 4,187 genes (1 of which
were partial) [0m
* M_sp__LZ_6_GCF_005871095 is stored with 4,356 genes (16 of which were
partial) [0m
* M_sp__LZ_8_GCF_005871205 is stored with 3,979 genes (16 of which were
partial) [0m
* M_sp__MCTG268_GCF_000744695 is stored with 4,098 genes (21 of which w
ere partial) [0m
* M_sp__N1_GCF_902506385 is stored with 3,971 genes (1 of which were pa
rtial) [0m
* M_sp__N4_GCF_002933275 is stored with 3,997 genes (56 of which were p
artial) [0m
* M_sp__NP_4_2019__GCF_003994855 is stored with 4,240 genes (5 of which
were partial) [0m
* M_sp__NP_6_GCF_003997005 is stored with 4,095 genes (6 of which were
 partial) [0m
* M_sp__P4B1_GCF_001447135 is stored with 3,452 genes (2 of which were
 partial) [0m
* M_sp__PJ_16_GCF_005298175 is stored with 4,021 genes (2 of which were
partial) [0m
* M_sp__PT19DW_GCF_003046275 is stored with 4,085 genes (9 of which wer
e partial) [0m
* M_sp__R17_GCF_003789045 is stored with 4,184 genes (41 of which were
 partial) [0m
* M_sp__THAF197a_GCF_009363275 is stored with 3,908 genes (1 of which w
ere partial) [0m
* M_sp__THAF39_GCF_009363515 is stored with 3,965 genes (1 of which wer
e partial) [0m
* M_sp__W62_GCF_004792665 is stored with 3,612 genes (10 of which were
 partial) [0m

```
* M_sp__X15_166B_GCF_001752365 is stored with 3,345 genes (1 of which w
ere partial)                                [0m
* M_sp__YJ_S3_2_GCF_004327985 is stored with 4,989 genes (287 of which
 were partial)                               [0m
* M_sp__YWL01_GCF_001601275 is stored with 5,183 genes (1,051 of which
 were partial)                               [0m
* M_sp__ZYF650_GCF_008370345 is stored with 3,920 genes (145 of which w
ere partial)                                [0m
* M_sp__es_042_GCF_900188315 is stored with 3,571 genes (0 of which wer
e partial)                                  [0m
* M_sp__es_048_GCF_900188435 is stored with 3,754 genes (3 of which wer
e partial)                                  [0m
* M_sp__lvr2a5a20_GCF_004365955 is stored with 4,529 genes (2 of which
 were partial)                               [0m
* M_subterrani_GCF_001045555 is stored with 4,193 genes (3 of which wer
e partial)                                  [0m
* M_vinifirmus_GCF_002258215 is stored with 3,579 genes (54 of which we
re partial)                                 [0m
* M_vulgaris_GCF_003344045 is stored with 3,560 genes (61 of which were
partial)                                   [0m
* M_vulgaris_GCF_007559285 is stored with 3,514 genes (20 of which were
partial)                                   [0m
* M_zhejiangensis_GCF_900114775 is stored with 3,668 genes (14 of which
were partial)                               [0m

The new genomes storage ....................: MARINOBACTER_GENOMES.db
(v6, signature: hashcbee1777)
Number of genomes ..........................: 113 (internal: 0, exter
nal: 113)
Number of gene calls .......................: 447,074
Number of partial gene calls ...............: 6,726
```

Now we can run the pangenomic analysis

```
anvi-pan-genome -g MARINOBACTER_GENOMES.db --project-name "Marinobacter" --output-
dir MARINOBACTER --num-threads 6 --minbit 0.5 --mcl-inflation 10 --use-ncbi-blast -
-enforce-hierarchical-clustering
```

To display the pangenome.

```
anvi-display-pan -p MARINOBACTER/Marinobacter-PAN.db -g MARINOBACTER_GENOMES.db
```

In [12]:
```
!ls MARINOBACTER/
```

```
Marinobacter-PAN.db             combined-aas.fa.unique.phr
SUMMARY                         combined-aas.fa.unique.pin
blast-search-results.txt        combined-aas.fa.unique.psq
blast-search-results.txt.unique log.txt
combined-aas.fa                 mcl-clusters.txt
combined-aas.fa.unique          mcl-input.txt
combined-aas.fa.unique.names
```

Selection of a genomic reference

The anvio pangenome analysis doesn't offer to generate a *consensus* reference from a set of genomes. Therefore, if we have 27 metatranscriptomic libraries and 113 genomes in the pangenome, this translates into 27 x 113 = 3051 mapping procedures (aand ultimately 891 BAM files), which is not feasible to plot or to process in a single figure. Therefore we need to chose only one genome as mapping reference for the metapangenomic analysis. We are going to choose the genome that recruits the largest amount of reads from our transcriptomic libraries.

To do so, we are going to add species name to each of the contigs of our list of `NCBI_genomes` by running the next two lines. The following lines were run in GACRC cluster.

```
ls | sed -e 's/_1-contigs.fa//' | awk '{print "%%%%%%%%_" $0 "&&&&&&&" $0 "_1-contigs.fa > " $0 "_v2.fa" }' | sed -e "s/%%%%%%%%/awk \'\/^>\/{print \$0 \"/" | sed -e "s/&&&&&&&/\"; next}{print}\' < /" > run_fix_names.sh
```

```
bash run_fix_names.sh
```

Now we run and place bowtie indexes in 04_MAPPING

```
mkdir 04_MAPPING
```

Then we run `bowtie2-build` on the references by submittin script
`99_SCRIPTS/run_bowtie2_build.sh`

Next, for each of the transcriptomic libraries we submit a mapping script. Template script is located in `99_SCRIPTS` as ***t-sub.sh_map_MT_2_refgenomes***. STDOUT files are located in `98_SCREENING`

In the next lines we are going to process the STDOUT files into a table.

In [ ]:
```
#!ls 02_CONTIGS/*contigs.db | cut -d'/' -f2 | sed -e 's/_1-contigs.db//' | awk '{print "anvi-export-contigs -c 02_CONTIGS/" $0 "_1-contigs.db -o " $0 "_contigs.fa"}'
#!mkdir NCBI_genomes_v2
#!mv *.fa NCBI_genomes_v2
#!tar czfv NCBI_genomes_v2.tgz NCBI_genomes_v2
```

In [13]:
```
!ls 98_SCREENING/
```

```
map_OIL11.e2001555 map_OIL32.e2002742 map_OIL53.e2002748 map_OIL81.e2002755
map_OIL14.e2001556 map_OIL34.e2002743 map_OIL61.e2002750 map_OIL82.e2002756
map_OIL17.e2001557 map_OIL37.e2002744 map_OIL64.e2002751 map_OIL84.e2002757
map_OIL25.e2001558 map_OIL44.e2002745 map_OIL67.e2002752 map_OIL85.e2002758
map_OIL28.e2001559 map_OIL47.e2002746 map_OIL70.e2002753 map_OIL87.e2002759
map_OIL29.e2001560 map_OIL5.e2002749  map_OIL78.e2002754 map_OIL90.e2002761
map_OIL31.e2002741 map_OIL50.e2002747 map_OIL8.e2002760
```

In [14]:
```
#!ls 98_SCREENING/ | awk '{print "XXXXXXXXX" $0"XXXXXXXXX" }' | sed -e
 "s/XXXXXXXXX/\'/g" | tr '\n' ','
```

In [ ]:

```
map_e_files = ['map_OIL11.e2001555','map_OIL32.e2002742', 'map_OIL53.e20
02748', 'map_OIL81.e2002755', 'map_OIL14.e2001556', 'map_OIL34.e2002743'
, 'map_OIL61.e2002750', 'map_OIL82.e2002756', 'map_OIL17.e2001557', 'map
_OIL37.e2002744', 'map_OIL64.e2002751', 'map_OIL84.e2002757', 'map_OIL2
5.e2001558', 'map_OIL44.e2002745', 'map_OIL67.e2002752', 'map_OIL85.e200
2758', 'map_OIL28.e2001559', 'map_OIL47.e2002746', 'map_OIL70.e2002753',
'map_OIL87.e2002759', 'map_OIL29.e2001560', 'map_OIL5.e2002749', 'map_OI
L78.e2002754', 'map_OIL90.e2002761', 'map_OIL31.e2002741', 'map_OIL50.e2
002747', 'map_OIL8.e2002760']
#map_e_files = ['map_OIL11.e1994842','map_OIL32.e1994849', 'map_OIL53.e1
994855', 'map_OIL81.e1994862', 'map_OIL14.e1994843', 'map_OIL34.e199485
0', 'map_OIL61.e1994857', 'map_OIL82.e1994863', 'map_OIL17.e1994844', 'm
ap_OIL37.e1994851', 'map_OIL64.e1994858', 'map_OIL84.e1994864', 'map_OIL
25.e1994845', 'map_OIL44.e1994852', 'map_OIL67.e1994859', 'map_OIL85.e19
94865', 'map_OIL28.e1994846', 'map_OIL47.e1994853', 'map_OIL70.e199486
0', 'map_OIL87.e1994866', 'map_OIL29.e1994847', 'map_OIL5.e1994856', 'ma
p_OIL78.e1994861', 'map_OIL90.e1994868', 'map_OIL31.e1994848', 'map_OIL5
0.e1994854', 'map_OIL8.e1994867']
row_names = [  'od_0',                'od_1_2',            'd2',
'o_4_1',              'd_0',               'd_1',               'bc_3'
,               'o_4_2',            'odn_0',             'odn_1',
'o_3',                'od_4_1',            'bc_1',              'bc_2'
,               'od_3',             'od_4_2',            'o_1_1',
'o_2',                'd_3',               'd_4',               'o_1_
2',               'bc_0',             'bc_4',              'odn_4',
'od_1_1',             'od_2',              'o_0']
data_mapped = []
data_rates = []
totalreads_vec = []
for e_file in map_e_files:
    #file = map_e_files[0]
    root = r'/Users/tito_miniconda/JOYE_LAB_ANVIO_PROJECTS/SK_BACKUP/p28
_pangenomes/Marinobacter/98_SCREENING/'
    location = root + e_file
    with open(location) as f:
        lines = f.readlines()
    lines = [x.strip() for x in lines]
    j = 0

    mapped_vec = []
    rates_vec = []
    for i in lines:
        if j == 0:
            fields = i.split(' ')
            total_reads = int(fields[0]) * 2
        if j == 3:
            fields = i.split(' ')
            l1 = int(fields[0]) * 2
        if j == 4:
            fields = i.split(' ')
            l2 = int(fields[0]) * 2
        if j == 7:
            fields = i.split(' ')
            l3 = int(fields[0]) * 2
        if j == 12:
            fields = i.split(' ')
```

```
            l4 = int(fields[0])
        if j == 13:
            fields = i.split(' ')
            l5 = int(fields[0])
            mapped = l1 + l2 + l3 + l4 + l5
            mapped_vec.append(mapped)
            rate = mapped * 100 / total_reads
            rates_vec.append(rate)
        if j == 14:
            j = -1
        j = j +1

    totalreads_vec.append(total_reads)
    data_mapped.append(mapped_vec)
    data_rates.append(rates_vec)
```

In [ ]:
```
location_script =  r'/Users/tito_miniconda/JOYE_LAB_ANVIO_PROJECTS/SK_BACKUP/p28_pangenomes/Marinobacter/99_SCRIPTS/t-sub.sh_map_MT_2_refgenomes'
with open(location_script) as f:
    lines = f.readlines()

lines = [x.strip() for x in lines]

species = []
for i in lines:
    if re.match(r"^bowtie2", i):
        fields = i.split(' ')
        fields0 = fields[5].split('/')
        fields1 = fields0[1].split('_in')
        species.append(fields1[0])
```

In [ ]:
```
# Creating pandas DataFrames
mapped_df = pd.DataFrame(data_mapped, columns = species, index=row_names)
maprates_df = pd.DataFrame(data_rates, columns = species, index=row_names)
```

Based on the average of mapped reads per library:

In [ ]:
```
mapped_df.mean(axis = 0).sort_values(ascending=False).head(10)
```

```
Marinobacter_sp__C18_GCF_001924925          63195.703704
Marinobacter_sp__NP_6_GCF_003997005         58878.814815
Marinobacter_sp__DSM_26291_GCF_900114695    57842.444444
Marinobacter_sp__EC_HK377_GCF_902498775     57531.370370
Marinobacter_sp__N1_GCF_902506385           57522.296296
Marinobacter_salarius_GCF_002116735         55153.407407
Marinobacter_salarius_GCF_000831005         55143.814815
Marinobacter_salarius_GCF_003986605         52839.259259
Marinobacter_sp__MCTG268_GCF_000744695      52154.185185
Marinobacter_sp__DS40M8_GCF_004936715       49617.370370
dtype: float64
```

Based on the average of the greatest (maximum) number of mapped reads in a library :

```
In [ ]: mapped_df.max(axis = 0).sort_values(ascending=False).head(10)
```

```
Marinobacter_sp__C18_GCF_001924925          626644
Marinobacter_sp__NP_6_GCF_003997005         597868
Marinobacter_sp__N1_GCF_902506385           582802
Marinobacter_sp__EC_HK377_GCF_902498775     582778
Marinobacter_sp__DSM_26291_GCF_900114695    565753
Marinobacter_salarius_GCF_002116735         556510
Marinobacter_salarius_GCF_000831005         550926
Marinobacter_salarius_GCF_003986605         524002
Marinobacter_sp__DS40M8_GCF_004936715       519884
Marinobacter_sp__MCTG268_GCF_000744695      516853
dtype: int64
```

Based on the average of mapping rates per library:

```
In [ ]: maprates_df.mean(axis = 0).sort_values(ascending=False).head(10)
```

```
Marinobacter_sp__C18_GCF_001924925          0.964301
Marinobacter_sp__NP_6_GCF_003997005         0.903556
Marinobacter_sp__EC_HK377_GCF_902498775     0.884463
Marinobacter_sp__N1_GCF_902506385           0.884388
Marinobacter_sp__DSM_26291_GCF_900114695    0.881725
Marinobacter_salarius_GCF_002116735         0.849136
Marinobacter_salarius_GCF_000831005         0.845815
Marinobacter_salarius_GCF_003986605         0.809907
Marinobacter_sp__MCTG268_GCF_000744695      0.798265
Marinobacter_sp__DS40M8_GCF_004936715       0.768613
dtype: float64
```

Based on the average of mapping rates per library:

In [ ]:
```
maprates_df.max(axis = 0).sort_values(ascending=False).head(10)
```

```
Marinobacter_sp__C18_GCF_001924925          11.821272
Marinobacter_sp__NP_6_GCF_003997005         11.278430
Marinobacter_sp__N1_GCF_902506385           10.994219
Marinobacter_sp__EC_HK377_GCF_902498775     10.993766
Marinobacter_sp__DSM_26291_GCF_900114695    10.672599
Marinobacter_salarius_GCF_002116735         10.498236
Marinobacter_salarius_GCF_000831005         10.392897
Marinobacter_salarius_GCF_003986605          9.884991
Marinobacter_sp__DS40M8_GCF_004936715        9.807308
Marinobacter_sp__MCTG268_GCF_000744695       9.750130
dtype: float64
```

The largest yield of reads recruitment is obtained using the genome of ***Marinobacter sp. C18***

Now I export the dataframes to prepare an excel sheet.

In [ ]:
```
mapped_df.to_csv('00_Marinobacter_mapping_reads.csv')
maprates_df.to_csv('01_Marinobacter_mapping_reads_rates.csv')
```

anvi BAM profile and downstream analysis

Now that we identified the best possible genomic reference, we proceed to obtain a RAM-bam file by running the `t-sub.sh_samtools1` script

In [18]:
```
#!ls /Volumes/Transcend/SK/p28_pangenome/Marinobacter/OIL* | cut -d'/' -
f7 | cut -d'-' -f1 | awk '{print "anvi-init-bam /Volumes/Transcend/SK/p2
8_pangenome/Marinobacter/" $0 "-RAW.bam -o " $0 ".bam"}'
```

```
anvi-init-bam /Volumes/Transcend/SK/p28_pangenome/Marinobacter/OIL11-RAW.bam
-o OIL11.bam
anvi-init-bam /Volumes/Transcend/SK/p28_pangenome/Marinobacter/OIL14-RAW.bam
-o OIL14.bam
...
```

Above `anvi-init-bam` commands were run in a bash console.

In [20]:
```
#!ls OIL*.bam | awk '{print "anvi-profile -i " $0 " -c 02_CONTIGS/Marino
bacter_sp__C18_GCF_001924925_1-contigs.db"}'
```

```
anvi-profile -i OIL11.bam --sample-name od_0   -c 02_CONTIGS/Marinobacter_sp
__C18_GCF_001924925_1-contigs.db
anvi-profile -i OIL14.bam --sample-name d_0    -c 02_CONTIGS/Marinobacter_sp
__C18_GCF_001924925_1-contigs.db
...
```

Above `anvi-profile` commands were run in a bash console.

```
!ls
```

```
00_LOGS                                    OIL5.bam.bai
00_Marinobacter_mapping_reads.csv          OIL50.bam
01_FASTA                                   OIL50.bam-ANVIO_PROFILE
01_Marinobacter_mapping_reads_rates.csv    OIL50.bam.bai
02_CONTIGS                                 OIL53.bam
98_SCREENING                               OIL53.bam-ANVIO_PROFILE
99_SCRIPTS                                 OIL53.bam.bai
MARINOBACTER                               OIL61.bam
MARINOBACTER_GENOMES.db                    OIL61.bam-ANVIO_PROFILE
Marinobacter.txt                           OIL61.bam.bai
NCBI_genomes                               OIL64.bam
NCBI_genomes_v2                            OIL64.bam-ANVIO_PROFILE
NCBI_genomes_v2.tgz                        OIL64.bam.bai
OIL11.bam                                  OIL67.bam
OIL11.bam-ANVIO_PROFILE                    OIL67.bam-ANVIO_PROFILE
OIL11.bam.bai                              OIL67.bam.bai
OIL14.bam                                  OIL70.bam
OIL14.bam-ANVIO_PROFILE                    OIL70.bam-ANVIO_PROFILE
OIL14.bam.bai                              OIL70.bam.bai
OIL17.bam                                  OIL78.bam
OIL17.bam-ANVIO_PROFILE                    OIL78.bam-ANVIO_PROFILE
OIL17.bam.bai                              OIL78.bam.bai
OIL25.bam                                  OIL8.bam
OIL25.bam-ANVIO_PROFILE                    OIL8.bam-ANVIO_PROFILE
OIL25.bam.bai                              OIL8.bam.bai
OIL28.bam                                  OIL81.bam
OIL28.bam-ANVIO_PROFILE                    OIL81.bam-ANVIO_PROFILE
OIL28.bam.bai                              OIL81.bam.bai
OIL29.bam                                  OIL82.bam
OIL29.bam-ANVIO_PROFILE                    OIL82.bam-ANVIO_PROFILE
OIL29.bam.bai                              OIL82.bam.bai
OIL31.bam                                  OIL84.bam
OIL31.bam-ANVIO_PROFILE                    OIL84.bam-ANVIO_PROFILE
OIL31.bam.bai                              OIL84.bam.bai
OIL32.bam                                  OIL85.bam
OIL32.bam-ANVIO_PROFILE                    OIL85.bam-ANVIO_PROFILE
OIL32.bam.bai                              OIL85.bam.bai
OIL34.bam                                  OIL87.bam
OIL34.bam-ANVIO_PROFILE                    OIL87.bam-ANVIO_PROFILE
OIL34.bam.bai                              OIL87.bam.bai
OIL37.bam                                  OIL90.bam
OIL37.bam-ANVIO_PROFILE                    OIL90.bam-ANVIO_PROFILE
OIL37.bam.bai                              OIL90.bam.bai
OIL44.bam                                  Selection_Marinobacter.xlsx
OIL44.bam-ANVIO_PROFILE                    default_config.txt
```

In [25]:
```
!anvi-merge OIL*ANVIO_PROFILE/PROFILE.db -c 02_CONTIGS/Marinobacter_sp__
C18_GCF_001924925_1-contigs.db -o SAMPLES-MERGED
```

```
[0m
WARNING
============================================
Anvi'o just set the normalization values for each sample based on how m
any
mapped reads they contained. This information will only be used to calc
ulate the
normalized coverage table. Here are those values: od_0: 0.22, d_0: 0.3
7, odn_0:
0.24, bc_1: 0.21, o_1_1: 0.12, o_1_2: 0.10, od_1_1: 0.13, od_1_2: 0.31,
d_1:
0.31, odn_1: 0.31, bc_2: 0.29, o_2: 0.01, bc_0: 0.22, od_2: 0.27, d_2:
 0.54,
bc_3: 0.56, o_3: 0.00, od_3: 0.85, d_3: 0.83, bc_4: 1.00, o_0: 0.21, o_
4_1:
0.02, o_4_2: 0.00, od_4_1: 0.17, od_4_2: 0.90, d_4: 0.09, odn_4: 0.23

profiler_version ..............................: 31
output_dir ....................................: /Users/tito_miniconda/J
OYE_LAB_ANVIO_PROJECTS/SK_BACKUP/p28_pangenomes/Marinobacter/SAMPLES-ME
RGED
sample_id .....................................: SAMPLES_MERGED
description ...................................: None
profile_db ....................................: /Users/tito_miniconda/J
OYE_LAB_ANVIO_PROJECTS/SK_BACKUP/p28_pangenomes/Marinobacter/SAMPLES-ME
RGED/PROFILE.db
merged ........................................: True
contigs_db_hash ...............................: hash4921dbbc
num_runs_processed ............................: 27
merged_sample_ids .............................: bc_0, bc_1, bc_2, bc_3,
bc_4, d_0, d_1, d_2, d_3, d_4, o_0, o_1_1, o_1_2, o_2, o_3, o_4_1, o_4_
2, od_0, od_1_1, od_1_2, od_2, od_3, od_4_1, od_4_2, odn_0, odn_1, odn_
4
Common layer additional data keys .............: default
total_reads_mapped ............................: 8297, 8607, 6383, 3299,
1850, 5056, 6021, 3430, 2226, 21708, 8801, 15724, 19142, 304200, 47915
6, 113551, 626644, 8530, 14463, 5878, 6792, 2176, 10617, 2060, 7719, 59
28, 8026
cmd_line ......................................: /Users/tito_miniconda/o
pt/miniconda3/envs/anvio-6.2/bin/anvi-merge OIL11.bam-ANVIO_PROFILE/PRO
FILE.db OIL14.bam-ANVIO_PROFILE/PROFILE.db OIL17.bam-ANVIO_PROFILE/PROF
ILE.db OIL25.bam-ANVIO_PROFILE/PROFILE.db OIL28.bam-ANVIO_PROFILE/PROFI
LE.db OIL29.bam-ANVIO_PROFILE/PROFILE.db OIL31.bam-ANVIO_PROFILE/PROFIL
E.db OIL32.bam-ANVIO_PROFILE/PROFILE.db OIL34.bam-ANVIO_PROFILE/PROFIL
E.db OIL37.bam-ANVIO_PROFILE/PROFILE.db OIL44.bam-ANVIO_PROFILE/PROFIL
E.db OIL47.bam-ANVIO_PROFILE/PROFILE.db OIL5.bam-ANVIO_PROFILE/PROFILE.
db OIL50.bam-ANVIO_PROFILE/PROFILE.db OIL53.bam-ANVIO_PROFILE/PROFILE.d
b OIL61.bam-ANVIO_PROFILE/PROFILE.db OIL64.bam-ANVIO_PROFILE/PROFILE.db
OIL67.bam-ANVIO_PROFILE/PROFILE.db OIL70.bam-ANVIO_PROFILE/PROFILE.db O
IL78.bam-ANVIO_PROFILE/PROFILE.db OIL8.bam-ANVIO_PROFILE/PROFILE.db OIL
81.bam-ANVIO_PROFILE/PROFILE.db OIL82.bam-ANVIO_PROFILE/PROFILE.db OIL8
4.bam-ANVIO_PROFILE/PROFILE.db OIL85.bam-ANVIO_PROFILE/PROFILE.db OIL8
7.bam-ANVIO_PROFILE/PROFILE.db OIL90.bam-ANVIO_PROFILE/PROFILE.db -c 02
_CONTIGS/Marinobacter_sp__C18_GCF_001924925_1-contigs.db -o SAMPLES-MER
GED
clustering_performed ..........................: True
```

```
[0m
WARNING
================================================
Codon frequencies were not profiled, hence, these tables will be empty
 in the
merged profile database.


[0m
* Anvi'o hierarchical clustering of contigs...

New items order ..............................: "tnf:euclidean:ward" (t
ype newick) has been added to the database...
New items order ..............................: "tnf-cov:euclidean:war
d" (type newick) has been added to the database...
New items order ..............................: "cov:euclidean:ward" (t
ype newick) has been added to the database...

* Additional data and layer orders...

Auxiliary Data ...............................: Found: /Users/tito_mini
conda/JOYE_LAB_ANVIO_PROJECTS/SK_BACKUP/p28_pangenomes/Marinobacter/SAM
PLES-MERGED/AUXILIARY-DATA.db (v. 2)
Profile Super ................................: Initialized with all 23
8 splits: /Users/tito_miniconda/JOYE_LAB_ANVIO_PROJECTS/SK_BACKUP/p28_p
angenomes/Marinobacter/SAMPLES-MERGED/PROFILE.db (v. 31)

[0m
Layer orders added
================================================
* std_coverage
* mean_coverage
* mean_coverage_Q2Q3
* max_normalized_ratio
* relative_abundance
* detection
* abundance
* variability

Data groups added
================================================
* default (w/2 items)

* Happy ☘
```

In [24]:
```
!ls
```

```
00_LOGS                                    OIL47.bam-ANVIO_PROFILE
00_Marinobacter_mapping_reads.csv          OIL5.bam-ANVIO_PROFILE
01_FASTA                                   OIL50.bam-ANVIO_PROFILE
01_Marinobacter_mapping_reads_rates.csv    OIL53.bam-ANVIO_PROFILE
02_CONTIGS                                 OIL61.bam-ANVIO_PROFILE
98_SCREENING                               OIL64.bam-ANVIO_PROFILE
99_SCRIPTS                                 OIL67.bam-ANVIO_PROFILE
MARINOBACTER                               OIL70.bam-ANVIO_PROFILE
MARINOBACTER_GENOMES.db                    OIL78.bam-ANVIO_PROFILE
Marinobacter.txt                           OIL8.bam-ANVIO_PROFILE
NCBI_genomes                               OIL81.bam-ANVIO_PROFILE
OIL11.bam-ANVIO_PROFILE                    OIL82.bam-ANVIO_PROFILE
OIL14.bam-ANVIO_PROFILE                    OIL84.bam-ANVIO_PROFILE
OIL17.bam-ANVIO_PROFILE                    OIL85.bam-ANVIO_PROFILE
OIL25.bam-ANVIO_PROFILE                    OIL87.bam-ANVIO_PROFILE
OIL28.bam-ANVIO_PROFILE                    OIL90.bam-ANVIO_PROFILE
OIL29.bam-ANVIO_PROFILE                    Untitled.ipynb
OIL31.bam-ANVIO_PROFILE                    default_config.txt
OIL32.bam-ANVIO_PROFILE                    external_genomes.txt
OIL34.bam-ANVIO_PROFILE                    metadata.txt
OIL37.bam-ANVIO_PROFILE                    refseq
OIL44.bam-ANVIO_PROFILE
```

In [26]:
```
!anvi-export-gene-coverage-and-detection -p SAMPLES-MERGED/PROFILE.db -c
02_CONTIGS/Marinobacter_sp__C18_GCF_001924925_1-contigs.db -O gene_cov_n
_detection.txt
```

```
Auxiliary Data ..............................: Found: SAMPLES-MERGED/A
UXILIARY-DATA.db (v. 2)    [0m
Profile Super ...............................: Initialized with all 23
8 splits: SAMPLES-MERGED/PROFILE.db (v. 31)
Gene coverages ..............................: gene_cov_n_detection.tx
t-GENE-COVERAGES.txt       [0m
Gene detection ..............................: gene_cov_n_detection.tx
t-GENE-DETECTION.txt
```

The following R script was used to incorporate gene coverage and gene detection profiles from the metatranscriptomes into the pangenome.

```
#!/usr/bin/env Rscript
rm(list=ls());
graphics.off();
setwd("/Users/tito_miniconda/JOYE_LAB_ANVIO_PROJECTS/SK_BACKUP/p28_pangenome
s/Marinobacter")

#Libraries
library("dplyr")

#Input data

#Gene clusters from the pangenome
gene_clusters_df <- read.table(file = 'MARINOBACTER/SUMMARY/Marinobacter_gen
e_clusters_summary.txt', header = TRUE, sep = "\t", quote = "")

#Gene coverage profiles from the metatranscriptomic profiles
gene_coverages_df <- read.table(file='gene_cov_n_detection.txt-GENE-COVERAGE
S.txt', header=TRUE, sep="\t", quote="")

#Gene detection profiles from the metatranscriptomic profiles
gene_detection_df <- read.table(file = 'gene_cov_n_detection.txt-GENE-DETECT
ION.txt', header=TRUE, sep="\t", quote="")

#Let's remember that gene_clusters_df includes entries for all of the genome
s stored in the pangenome.
#At some point we need to subset those gene_callers_id entries that belong t
o the reference genome of the
#transcriptomic merged profiles.

#'M_sp__C18_GCF_001924925' was the tag utilized for Marinobacter sp. C18

ref_genome =  'M_sp__C18_GCF_001924925'

#Creating zero dataframes to store processed data
gene_cluster_names <- gene_clusters_df[,colnames(gene_clusters_df) %in% c('g
ene_cluster_id')]
gene_cluster_names <- unique(gene_cluster_names)
samples_names <- colnames(gene_coverages_df)
samples_names <- samples_names[2:length(samples_names)]
num_cols = length(samples_names)
num_rows = length(gene_cluster_names)
out_coverage_df = data.frame(matrix(0,ncol = num_cols, nrow = num_rows))
colnames(out_coverage_df ) <- samples_names
rownames(out_coverage_df ) <- gene_cluster_names
out_detection_df <- out_coverage_df

#Loop to calculate the maximum coverage and maximum detection observed on ea
ch gene cluster.
for (gene_cluster in levels(gene_clusters_df$gene_cluster_id)){
```

```
    df <- gene_clusters_df[gene_clusters_df$gene_cluster_id == gene_cluster, ]
    df2 <- df[df$genome_name == ref_genome,]
    if(length(df2$gene_callers_id) > 0){
      for(my_gene_callers_id in df2$gene_callers_id){
        #For coverages
        df3 <- gene_coverages_df[gene_coverages_df$key == my_gene_callers_id,
   ]
        x1 <- as.vector(df3)
        x2 <- x1[2:length(x1)]
        #For detections
        df4 <- gene_detection_df[gene_detection_df$key == my_gene_callers_id,
   ]
        x3 <- as.vector(df4)
        x4 <- x3[2:length(x3)]
```

In [41]:
```
#Next line removes the columns of coverage. Detection columns are enough
to be uploaded to anvio.

!cat gene_clusters_additional_data.txt | awk '{print $1, "\t", $29,
 "\t", $30, "\t", $31, "\t", $32, "\t", $33, "\t", $34, "\t",$35, "\t",
 $36, "\t", $37, "\t", $38, "\t", $39, "\t", $40, "\t", $41, "\t", $42,
 "\t", $43, "\t", $44, "\t", $45, "\t", $46, "\t", $47, "\t", $48, "\t",
$49, "\t", $50, "\t", $51, "\t", $52, "\t", $53, "\t", $54, "\t", $55}'
| sed -e 's/ //g' > gene_clusters_additional_data_v2.txt
```

In [43]:
```
!anvi-import-misc-data gene_clusters_additional_data_v2.txt -p MARINOBAC
TER/Marinobacter-PAN.db --target-data-table items --just-do-it
```

```
New data for 'items' in data group 'default'
================================================
Data key "det_bc_0" .........................: Predicted type: float
Data key "det_bc_1" .........................: Predicted type: float
Data key "det_bc_2" .........................: Predicted type: float
Data key "det_bc_3" .........................: Predicted type: float
Data key "det_bc_4" .........................: Predicted type: float
Data key "det_d_0" ..........................: Predicted type: float
Data key "det_d_1" ..........................: Predicted type: float
Data key "det_d_2" ..........................: Predicted type: float
Data key "det_d_3" ..........................: Predicted type: float
Data key "det_d_4" ..........................: Predicted type: float
Data key "det_o_0" ..........................: Predicted type: float
Data key "det_o_1_1" ........................: Predicted type: float
Data key "det_o_1_2" ........................: Predicted type: float
Data key "det_o_2" ..........................: Predicted type: float
Data key "det_o_3" ..........................: Predicted type: float
Data key "det_o_4_1" ........................: Predicted type: float
Data key "det_o_4_2" ........................: Predicted type: float
Data key "det_od_0" .........................: Predicted type: float
Data key "det_od_1_1" .......................: Predicted type: float
Data key "det_od_1_2" .......................: Predicted type: float
Data key "det_od_2" .........................: Predicted type: float
Data key "det_od_3" .........................: Predicted type: float
Data key "det_od_4_1" .......................: Predicted type: float
Data key "det_od_4_2" .......................: Predicted type: float
Data key "det_odn_0" ........................: Predicted type: float
Data key "det_odn_1" ........................: Predicted type: float
Data key "det_odn_4" ........................: Predicted type: float


WARNING
================================================
The following keys in your data dict will replace the ones that are alr
eady in
your pan database items table and default data group: det_bc_0, det_bc_
1,
det_bc_2, det_bc_3, det_bc_4, det_d_0, det_d_1, det_d_2, det_d_3, det_d
_4,
det_o_0, det_o_1_1, det_o_1_2, det_o_2, det_o_3, det_o_4_1, det_o_4_2,
 det_od_0,
det_od_1_1, det_od_1_2, det_od_2, det_od_3, det_od_4_1, det_od_4_2, det
_odn_0,
det_odn_1, det_odn_4.


WARNING
================================================
Data from the table 'items' for the following data keys in data group
 'default'
removed from the database: 'det_bc_0, det_bc_1, det_bc_2, det_bc_3, det
_bc_4,
det_d_0, det_d_1, det_d_2, det_d_3, det_d_4, det_o_0, det_o_1_1, det_o_
1_2,
det_o_2, det_o_3, det_o_4_1, det_o_4_2, det_od_0, det_od_1_1, det_od_1_
2,
```

```
det_od_2, det_od_3, det_od_4_1, det_od_4_2, det_odn_0, det_odn_1, det_o
dn_4'.
#SAD.


NEW DATA
================================================
Database ....................................: pan
Data group ..................................: default
Data table ..................................: items
New data keys ...............................: det_bc_0, det_bc_1, det
_bc_2, det_bc_3, det_bc_4, det_d_0, det_d_1, det_d_2, det_d_3, det_d_4,
det_o_0, det_o_1_1, det_o_1_2, det_o_2, det_o_3, det_o_4_1, det_o_4_2,
 det_od_0, det_od_1_1, det_od_1_2, det_od_2, det_od_3, det_od_4_1, det_
od_4_2, det_odn_0, det_odn_1, det_odn_4.
```

Spliting pangenome into accessory and pseudocore pangenomes

Let's define the accessory pangenome as the union of singletons and doubletons, while the pseudocore pangenome is the complement of the accessory pangenome.

First, we select in the search tab by using the expression `Det X` > 0, where `X` is any of the treatments, and then click on append to selected bin, which would be `T_signals` and `no_signals`. They were stored in the bin collection `bins_by_recovery`. Next we exoirt `T_signals` into a separate anvio analysis.

Check *Colwellia* notebook to see hwo this should look like before splitting.

We are going to procee to split by recovery

In [44]:
```
!anvi-split -p MARINOBACTER/Marinobacter-PAN.db -g MARINOBACTER_GENOMES.
db -C bins_by_recovery -o SPLIT_PANs
```

```
Genomes storage ..........................................: Initialized (storage hash: hashcbee1777)8;5;0m ETA: 0s
Num genomes in storage ...................................: 113
Num genomes will be used .................................: 113
Pan DB ...................................................: Initialized: MARINOBACTER/Marinobacter-PAN.db (v. 13)
Gene cluster homogeneity estimates .......................: Functional: [YES]; Geometric: [YES]; Combined: [YES]

[0m
* Gene clusters are initialized for all 37910 gene clusters in the database.


[0m
WARNING
================================================================
Anvi'o is about to start splitting your bins into individual, self-contained
anvi'o profiles. This is quite a tricky operation, and even if it finishes
successfully, you must double check everyting in the resulting profiles to make
sure things worked as expected. Although we are doing our best to test all
these, variation between projects make it impossible to be 100% sure.

Collections ..............................: The collection "DEFAULT" that describes 3,924 splits and 1 bins has been successfully added to the database at "SPLIT_PANs/T_signals/PAN.db". Here is a full list of the bin names in this collection: ALL_SPLITS.
New items order ..........................: "frequency:euclidean:ward" (type newick) has been added to the database...

[0m
WARNING
============================================
Clustering for "frequency:euclidean:ward" is already in the database. It will be
replaced with the new content.

New items order ..........................: "frequency:euclidean:ward" (type newick) has been added to the database...
New items order ..........................: "presence-absence:euclidean:ward" (type newick) has been added to the database...

WARNING
================================================================
It seems you have more than 20,000 splits in this particular bin. This is the
soft limit for anvi'o to attempt to create a hierarchical clustering of your
splits (which becomes the center tree in all anvi'o displays). If you want a
hierarchical clustering to be done anyway, you can re-run the splitting process
only for this bin by adding these parameters to your run: '--bin-id no_
```

```
signals
--enforce-hierarchical-clustering'. If you feel like you are lost, do
n't
hesitate to get in touch with anvi'o developers.

Collections ..................................: The collection "DEFAUL
T" that describes 33,986 splits and 1 bins has been successfully added
 to the database at "SPLIT_PANs/no_signals/PAN.db". Here is a full list
of the bin names in this collection: ALL_SPLITS.
Num bins processed ...........................................: 2
Output directory .............................................: /Users/t
ito_miniconda/JOYE_LAB_ANVIO_PROJECTS/SK_BACKUP/p28_pangenomes/Marinoba
cter/SPLIT_PANs
```

From now on we are interested on the fraction of the pangenome that recovered transcriptomic signals.

```
anvi-display-pan -p MARINOBACTER/Marinobacter-PAN.db -g MARINOBACTER_GENOMES.db
```

Differentially Expressed Genes Layers

Counting mapped reads per gene using htseq

This section I am following [https://metagenomics-workshop.readthedocs.io/en/latest/annotation/quantification.html (https://metagenomics-workshop.readthedocs.io/en/latest/annotation/quantification.html)](https://metagenomics-workshop.readthedocs.io/en/latest/annotation/quantification.html). First we need to extract the a GFF file from the annotation of anvio.

In [45]:
```
!anvi-get-sequences-for-gene-calls -c 02_CONTIGS/Marinobacter_sp__C18_GC
F_001924925_1-contigs.db -o ref_genome.gff --export-gff3
```

```
Contigs DB ...................................: Initialized: 02_CONTIG
S/Marinobacter_sp__C18_GCF_001924925_1-contigs.db (v. 14)

WARNING
================================================
You did not provide any gene caller ids. As a result, anvi'o will give
 you back
sequences for every 4593 gene call stored in the contigs database.

Output .......................................: ref_genome.gff
[0m
```

In [46]:
```
#Convert gff to gtf format which is compatible with htseq
!cat ref_genome.gff | grep -v "#" | grep "ID=" | cut -f1 -d ';' | sed
's/ID=//g' | cut -f1,4,5,7,9 | awk -v OFS='\t' '{print $1, "PRODIGAL","C
DS",$2,$3,".",$4,".","gene_id " $5}' >  ref_genome.gtf
```

The following lines were run in console:

```
!htseq-count -r pos -t CDS -f bam --minaqual 2 /Volumes/Transcend/SK/p28_pan
genome/Marinobacter/sorted_bam/OIL11.sorted.bam ref_genome.gtf > od_0_mapcou
nts.txt
!htseq-count -r pos -t CDS -f bam --minaqual 2 /Volumes/Transcend/SK/p28_pan
genome/Marinobacter/sorted_bam/OIL14.sorted.bam ref_genome.gtf > d_0_mapcoun
ts.txt
!htseq-count -r pos -t CDS -f bam --minaqual 2 /Volumes/Transcend/SK/p28_pan
genome/Marinobacter/sorted_bam/OIL17.sorted.bam ref_genome.gtf > odn_0_mapco
unts.txt
!htseq-count -r pos -t CDS -f bam --minaqual 2 /Volumes/Transcend/SK/p28_pan
genome/Marinobacter/sorted_bam/OIL25.sorted.bam ref_genome.gtf > bc_1_mapcou
nts.txt
!htseq-count -r pos -t CDS -f bam --minaqual 2 /Volumes/Transcend/SK/p28_pan
genome/Marinobacter/sorted_bam/OIL28.sorted.bam ref_genome.gtf > o_1_1_mapco
unts.txt
!htseq-count -r pos -t CDS -f bam --minaqual 2 /Volumes/Transcend/SK/p28_pan
genome/Marinobacter/sorted_bam/OIL29.sorted.bam ref_genome.gtf > o_1_2_mapco
unts.txt
!htseq-count -r pos -t CDS -f bam --minaqual 2 /Volumes/Transcend/SK/p28_pan
genome/Marinobacter/sorted_bam/OIL31.sorted.bam ref_genome.gtf > od_1_1_mapc
ounts.txt
!htseq-count -r pos -t CDS -f bam --minaqual 2 /Volumes/Transcend/SK/p28_pan
genome/Marinobacter/sorted_bam/OIL32.sorted.bam ref_genome.gtf > od_1_2_mapc
ounts.txt
!htseq-count -r pos -t CDS -f bam --minaqual 2 /Volumes/Transcend/SK/p28_pan
genome/Marinobacter/sorted_bam/OIL34.sorted.bam ref_genome.gtf > d_1_mapcoun
ts.txt
!htseq-count -r pos -t CDS -f bam --minaqual 2 /Volumes/Transcend/SK/p28_pan
genome/Marinobacter/sorted_bam/OIL37.sorted.bam ref_genome.gtf > odn_1_mapco
unts.txt
!htseq-count -r pos -t CDS -f bam --minaqual 2 /Volumes/Transcend/SK/p28_pan
genome/Marinobacter/sorted_bam/OIL44.sorted.bam ref_genome.gtf > bc_2_mapcou
nts.txt
!htseq-count -r pos -t CDS -f bam --minaqual 2 /Volumes/Transcend/SK/p28_pan
genome/Marinobacter/sorted_bam/OIL47.sorted.bam ref_genome.gtf > o_2_mapcoun
ts.txt
!htseq-count -r pos -t CDS -f bam --minaqual 2 /Volumes/Transcend/SK/p28_pan
genome/Marinobacter/sorted_bam/OIL5.sorted.bam ref_genome.gtf > bc_0_mapcoun
ts.txt
!htseq-count -r pos -t CDS -f bam --minaqual 2 /Volumes/Transcend/SK/p28_pan
genome/Marinobacter/sorted_bam/OIL50.sorted.bam ref_genome.gtf > od_2_mapcou
nts.txt
```

In [3]:
```
!cut -f4,5,9 ref_genome.gtf | sed 's/gene_id //g' | gawk '{print $3,$2
-$1+1}' | tr ' ' '\t' > ref_genome.genelengths
```

The following steps will calculate TPM values for contigs or genes based on count files

TPM values are defined as in Wagner et al (Theory in Biosciences) 2012.

$TPM_i = \frac{rg*rl*10^6}{f*T}$

rg: reads mapped to gene g

rl: read length

f: feature length

$T = \sum_i \frac{rg*rl}{f}$ for all i genes

For this calculation we need to know the average read length

In [4]:
```
read_files = ['OIL11_f.fa',  'OIL25_f.fa',  'OIL31_f.fa',  'OIL37_f.fa',
'OIL50_f.fa',  'OIL61_f.fa',  'OIL70_f.fa',  'OIL82_f.fa',  'OIL87_f.fa'
, 'OIL11_r.fa',  'OIL25_r.fa',  'OIL31_r.fa',  'OIL37_r.fa',  'OIL50_r.f
a',  'OIL61_r.fa',  'OIL70_r.fa',  'OIL82_r.fa',  'OIL87_r.fa','OIL14_f.
fa',  'OIL28_f.fa',  'OIL32_f.fa',  'OIL44_f.fa',  'OIL53_f.fa',  'OIL64
_f.fa',  'OIL78_f.fa',  'OIL84_f.fa',  'OIL8_f.fa'  ,'OIL14_r.fa',  'OIL
28_r.fa',  'OIL32_r.fa',  'OIL44_r.fa',  'OIL53_r.fa',  'OIL64_r.fa',
'OIL78_r.fa',  'OIL84_r.fa',  'OIL8_r.fa',    'OIL17_f.fa',  'OIL29_f.f
a',  'OIL34_f.fa',  'OIL47_f.fa',  'OIL5_f.fa',  'OIL67_f.fa',  'OIL81_
f.fa',  'OIL85_f.fa',  'OIL90_f.fa',   'OIL17_r.fa',  'OIL29_r.fa',  'OI
L34_r.fa',  'OIL47_r.fa',  'OIL5_r.fa',  'OIL67_r.fa',  'OIL81_r.fa',
'OIL85_r.fa',  'OIL90_r.fa']
sample_ids = ['od_0',         'bc_1',         'od_1_1',       'odn_1',
'od_2',         'bc_3',         'd_3',          'o_4_2',        'd_4',
'od_0',         'bc_1',         'od_1_1',       'odn_1',        'od_2',
'bc_3',         'd_3',          'o_4_2',        'd_4',        'd_0',
'o_1_1',        'od_1_2',       'bc_2',         'd_2',          'o_3',
'bc_4',         'od_4_1',       'o_0'         ,'d_0',         'o_1_1',
'od_1_2',       'bc_2',         'd_2',          'o_3',          'bc_4',
'od_4_1',       'o_0',          'odn_0',        'o_1_2',        'd_1',
'o_2',          'bc_0',        'od_3',         'o_4_1',        'od_4_2',
'odn_4',         'odn_0',        'o_1_2',        'd_1',          'o_2',
'bc_0',        'od_3',         'o_4_1',        'od_4_2',       'odn_4']
```

In [5]:
```
# THIS TAKES TIME TO RUN
#root = '/Volumes/Transcend/SK/p01_clean/p03_splited/'
#reads_vec = []
#total_bp_in_lib_vec = []

#for i in read_files:
#    my_location = root + i
#    command1 = 'cat ' + my_location + ' | grep \'^>\' | wc -l'
#    runcommand1 = os.popen(command1)
#    num_reads = runcommand1.read()
#    runcommand1.close()
#    num_reads = num_reads.strip()
#    reads_vec.append(num_reads)
#    command2 = 'cat ' + my_location + ' | grep -v  \'^>\' | awk \'{ pri
nt length }\' | awk \'{s+=$1}END{print s}\''
#    runcommand2 = os.popen(command2)
#    total_bp_in_lib = runcommand2.read()
#    runcommand2.close()
#    total_bp_in_lib = total_bp_in_lib.strip()
#    total_bp_in_lib_vec.append(total_bp_in_lib)
```

In [8]:
```
reads_vec = ['5319504', ' 2599528', ' 8665216', ' 4856793', ' 6533258',
' 2742491', ' 4059579', ' 2650493', ' 7410098', ' 5319504', ' 2599528',
' 8665216', ' 4856793', ' 6533258', ' 2742491', ' 4059579', ' 2650493',
' 7410098', ' 3008480', ' 7395105', ' 6898923', ' 3008495', ' 4951454',
' 4587833', ' 3455976', ' 8377386', ' 4570606', ' 3008480', ' 7395105',
' 6898923', ' 3008495', ' 4951454', ' 4587833', ' 3455976', ' 8377386',
' 4570606', ' 6753873', ' 6867899', ' 4699125', ' 2359617', ' 4238895',
' 5235701', ' 6756836', ' 2180731', ' 9152171', ' 6753873', ' 6867899',
' 4699125', ' 2359617', ' 4238895', ' 5235701', ' 6756836', ' 2180731',
' 9152171']
total_bp_in_lib_vec = ['529556914', ' 258358315', ' 860860737', ' 471734
413', ' 637162234', ' 260148861', ' 385237010', ' 247448659', ' 72097031
8', ' 527979712', ' 257355106', ' 861328463', ' 480966006', ' 648081155'
, ' 270020595', ' 396129359', ' 256912761', ' 728241939', ' 299481143',
' 733713272', ' 684447286', ' 299191768', ' 483932122', ' 434358626', '
 324037321', ' 788966972', ' 455633481', ' 298087834', ' 733573535', ' 6
87327923', ' 298745950', ' 490543446', ' 451526124', ' 335033361', ' 814
356799', ' 453712387', ' 667248186', ' 684652368', ' 460414958', ' 23269
1190', ' 422225840', ' 497983867', ' 640662788', ' 203200060', ' 8995786
10', ' 665617351', ' 681864181', ' 466017852', ' 233762389', ' 42071695
1', ' 511511621', ' 658014105', ' 211690251', ' 902781824']
read_length_d = {'library': read_files, 'sample_id' : sample_ids, 'num_r
eads_in_lib' : reads_vec, 'total_bp_in_lib': total_bp_in_lib_vec }
read_length_df = pd.DataFrame(read_length_d)
```

In [9]:
```
read_length_df
```

Out[9]:

	library	sample_id	num_reads_in_lib	total_bp_in_lib
0	OIL11_f.fa	od_0	5319504	529556914
1	OIL25_f.fa	bc_1	2599528	258358315
2	OIL31_f.fa	od_1_1	8665216	860860737
3	OIL37_f.fa	odn_1	4856793	471734413
4	OIL50_f.fa	od_2	6533258	637162234
5	OIL61_f.fa	bc_3	2742491	260148861
6	OIL70_f.fa	d_3	4059579	385237010
7	OIL82_f.fa	o_4_2	2650493	247448659
8	OIL87_f.fa	d_4	7410098	720970318
9	OIL11_r.fa	od_0	5319504	527979712
10	OIL25_r.fa	bc_1	2599528	257355106
11	OIL31_r.fa	od_1_1	8665216	861328463
12	OIL37_r.fa	odn_1	4856793	480966006
13	OIL50_r.fa	od_2	6533258	648081155
14	OIL61_r.fa	bc_3	2742491	270020595
15	OIL70_r.fa	d_3	4059579	396129359
16	OIL82_r.fa	o_4_2	2650493	256912761
17	OIL87_r.fa	d_4	7410098	728241939
18	OIL14_f.fa	d_0	3008480	299481143
19	OIL28_f.fa	o_1_1	7395105	733713272
20	OIL32_f.fa	od_1_2	6898923	684447286
21	OIL44_f.fa	bc_2	3008495	299191768
22	OIL53_f.fa	d_2	4951454	483932122
23	OIL64_f.fa	o_3	4587833	434358626
24	OIL78_f.fa	bc_4	3455976	324037321
25	OIL84_f.fa	od_4_1	8377386	788966972
26	OIL8_f.fa	o_0	4570606	455633481
27	OIL14_r.fa	d_0	3008480	298087834
28	OIL28_r.fa	o_1_1	7395105	733573535
29	OIL32_r.fa	od_1_2	6898923	687327923
30	OIL44_r.fa	bc_2	3008495	298745950
31	OIL53_r.fa	d_2	4951454	490543446
32	OIL64_r.fa	o_3	4587833	451526124
33	OIL78_r.fa	bc_4	3455976	335033361

	library	sample_id	num_reads_in_lib	total_bp_in_lib
34	OIL84_r.fa	od_4_1	8377386	814356799
35	OIL8_r.fa	o_0	4570606	453712387
36	OIL17_f.fa	odn_0	6753873	667248186
37	OIL29_f.fa	o_1_2	6867899	684652368
38	OIL34_f.fa	d_1	4699125	460414958
39	OIL47_f.fa	o_2	2359617	232691190
40	OIL5_f.fa	bc_0	4238895	422225840
41	OIL67_f.fa	od_3	5235701	497983867
42	OIL81_f.fa	o_4_1	6756836	640662788
43	OIL85_f.fa	od_4_2	2180731	203200060
44	OIL90_f.fa	odn_4	9152171	899578610
45	OIL17_r.fa	odn_0	6753873	665617351
46	OIL29_r.fa	o_1_2	6867899	681864181
47	OIL34_r.fa	d_1	4699125	466017852
48	OIL47_r.fa	o_2	2359617	233762389
49	OIL5_r.fa	bc_0	4238895	420716951
50	OIL67_r.fa	od_3	5235701	511511621
51	OIL81_r.fa	o_4_1	6756836	658014105
52	OIL85_r.fa	od_4_2	2180731	211690251
53	OIL90_r.fa	odn_4	9152171	902781824

In [10]:

```
read_length_df_v2 = read_length_df[['sample_id','num_reads_in_lib', 'tot
al_bp_in_lib']]
read_length_df_v2[['num_reads_in_lib', 'total_bp_in_lib']] = read_length
_df_v2[['num_reads_in_lib', 'total_bp_in_lib']].astype('int64')
read_length_df_v2 = read_length_df_v2.groupby(['sample_id']).sum()
read_length_df_v2['avg_read_length'] = read_length_df_v2['total_bp_in_li
b']/read_length_df_v2['num_reads_in_lib']
read_length_df_v2
```

```
/usr/local/lib/python3.7/site-packages/pandas/core/frame.py:2963: Setti
ngWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  self[k1] = value[k2]
```

Out[10]:

sample_id	num_reads_in_lib	total_bp_in_lib	avg_read_length
bc_0	8477790	842942791	99.429544
bc_1	5199056	515713421	99.193665
bc_2	6016990	597937718	99.374890
bc_3	5484982	530169456	96.658377
bc_4	6911952	659070682	95.352323
d_0	6016960	597568977	99.314102
d_1	9398250	926432810	98.575034
d_2	9902908	974475568	98.402971
d_3	8119158	781366369	96.237365
d_4	14820196	1449212257	97.786308
o_0	9141212	909345868	99.477604
o_1_1	14790210	1467286807	99.206624
o_1_2	13735798	1366516549	99.485778
o_2	4719234	466453579	98.840952
o_3	9175666	885884750	96.547188
o_4_1	13513672	1298676893	96.100963
o_4_2	5300986	504361420	95.144832
od_0	10639008	1057536626	99.401808
od_1_1	17330432	1722189200	99.373703
od_1_2	13797846	1371775209	99.419519
od_2	13066516	1285243389	98.361598
od_3	10471402	1009495488	96.404998
od_4_1	16754772	1603323771	95.693559
od_4_2	4361462	414890311	95.126430
odn_0	13507746	1332865537	98.674163
odn_1	9713586	952700419	98.079167
odn_4	18304342	1802360434	98.466278

In [11]:
```
!ls 96_htseq_output/
```

```
bc_0_mapcounts.txt   d_4_mapcounts.txt    od_1_1_mapcounts.txt
bc_1_mapcounts.txt   o_0_mapcounts.txt    od_1_2_mapcounts.txt
bc_2_mapcounts.txt   o_1_1_mapcounts.txt  od_2_mapcounts.txt
bc_3_mapcounts.txt   o_1_2_mapcounts.txt  od_3_mapcounts.txt
bc_4_mapcounts.txt   o_2_mapcounts.txt    od_4_1_mapcounts.txt
d_0_mapcounts.txt    o_3_mapcounts.txt    od_4_2_mapcounts.txt
d_1_mapcounts.txt    o_4_1_mapcounts.txt  odn_0_mapcounts.txt
d_2_mapcounts.txt    o_4_2_mapcounts.txt  odn_1_mapcounts.txt
d_3_mapcounts.txt    od_0_mapcounts.txt   odn_4_mapcounts.txt
```

In [14]:
```
sample_names = pd.Series(read_length_df_v2.index.values)
my_lengths = r'/Users/tito_miniconda/JOYE_LAB_ANVIO_PROJECTS/SK_BACKUP/p
28_pangenomes/Marinobacter/ref_genome.genelengths'
gene_lengths = pd.read_table(my_lengths, header=None, index_col=0, names
=['gene_id','gene_length'])
root = r'/Users/tito_miniconda/JOYE_LAB_ANVIO_PROJECTS/SK_BACKUP/p28_pan
genomes/Marinobacter/96_htseq_output/'
my_tpm_df = pd.DataFrame()

for i in np.arange(sample_names.size):
    counts_file = sample_names[i] + '_mapcounts.txt'
    location = root + counts_file
    rg = pd.read_table(location, header=None, index_col=0, names=['gene_
id', 'count'])
    rg_v2 = rg.drop(rg.tail(5).index)
    ## Intersect with genes in the gene length file
    rg_v3 = rg_v2.iloc[list(set(gene_lengths.index).intersection(set(rg_
v2.index.astype('int64'))))]
    gene_lengths_v2 = gene_lengths.iloc[list(rg_v3.index)]
    rg_v3.index = rg_v3.index.astype('int64')
    gene_lengths_v2.index = gene_lengths_v2.index.astype('int64')
    ## Average read length for sample
    rl = read_length_df_v2.at[sample_names[i],'avg_read_length']
    ## Calculate T for sample
    T = np.sum(rl * rg_v3['count'].divide(gene_lengths_v2['gene_length'
]))
    ## Calculate TPM for sample
    tpm = (1e6*rl/T)*rg_v3['count'].divide(gene_lengths_v2['gene_length'
])
    ## Create dataframe
    tpm_se = pd.DataFrame(tpm, columns=[sample_names[i]])
    ## Concatenate to results
    my_tpm_df = pd.concat([my_tpm_df,tpm_se],axis=1)
```

In [15]:
```
my_tpm_df
```

Out[15]:

gene_id	bc_0	bc_1	bc_2	bc_3	bc_4	d_0	d_
0	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000
1	852.475700	0.000000	1341.072151	721.506658	1223.849998	0.000000	1128.56667
10	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000
100	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000
1000	0.000000	1666.872316	0.000000	1118.480201	0.000000	0.000000	583.16832
...	...	...	...	...	...	...	
995	112.497648	0.000000	0.000000	0.000000	0.000000	173.832596	0.00000
996	806.508873	381.480889	0.000000	0.000000	0.000000	0.000000	533.85629
997	138.666507	131.179269	0.000000	0.000000	0.000000	0.000000	0.00000
998	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000
999	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.00000

4593 rows × 27 columns

In [16]:
```
my_tpm_df.to_csv('Marinobacter_tpm.csv', sep='\t')
```

DESeq in R

Run here the notebook located in

`/Users/tito_miniconda/JOYE_LAB_ANVIO_PROJECTS/SK_BACKUP/p28_pangenomes/DE_Analysis/`

Load DE profiles and convert from gene to gene_cluster table

In [1]:
```
!sqlite3 -header -csv SPLIT_PANs/T_signals/PAN.db "select * from gene_clusters;" > pan_split_gene_clusters.csv
```

In [2]:
```
!mv pan_split_gene_clusters.csv SPLIT_PANs/
```

In [4]:
```
pan_split_geneclusters_df = pd.read_csv(r'/Users/tito_miniconda/JOYE_LAB_ANVIO_PROJECTS/SK_BACKUP/p28_pangenomes/Marinobacter/SPLIT_PANs/pan_split_gene_clusters.csv')
```

In [3]:
```
pan_split_geneclusters_df = pan_split_geneclusters_df[pan_split_geneclusters_df.genome_name == 'M_sp__C18_GCF_001924925']
```

In [4]:
```
DESEQ_df = pd.read_csv(r'/Users/tito_miniconda/JOYE_LAB_ANVIO_PROJECTS/SK_BACKUP/p28_pangenomes/DE_Analysis/Marinobacter_DE_df.txt', sep='\t')
```

In [5]:
```
sig_df = DESEQ_df[((DESEQ_df.padj < 0.05) & (DESEQ_df.zeros_in_bc < 3))
| (DESEQ_df.t_test_p_value < 0.05)]
```

In [6]:
```
sig_df.rename(columns={'gene_id':'gene_caller_id'}, inplace=True)
```

```
/usr/local/lib/python3.7/site-packages/pandas/core/frame.py:4133: Setti
ngWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  errors=errors,
```

In [7]:
```
#Building output scheme
DE_for_anvio_df = pd.read_csv( r'/Users/tito_miniconda/JOYE_LAB_ANVIO_PR
OJECTS/SK_BACKUP/p28_pangenomes/Marinobacter/gene_clusters_additional_da
ta.txt', sep='\t')
DE_for_anvio_df = DE_for_anvio_df.iloc[:,0:9]
DE_for_anvio_df.columns= ['gene_cluster_id', 'DE_d!A', 'DE_d!B', 'DE_o!
A', 'DE_o!B', 'DE_od!A', 'DE_od!B', 'DE_odn!A', 'DE_odn!B',]
for col in ['DE_d!A', 'DE_d!B', 'DE_o!A', 'DE_o!B', 'DE_od!A', 'DE_od!B'
, 'DE_odn!A', 'DE_odn!B']:
    DE_for_anvio_df[col].values[:] = 0
ref_gc_list = list(pan_split_geneclusters_df.gene_cluster_id.unique())
DE_for_anvio_df = DE_for_anvio_df[DE_for_anvio_df['gene_cluster_id'].isi
n(ref_gc_list)]
```

In [8]:
```
DE_for_anvio_df
```

Out[8]:

	gene_cluster_id	DE_d!A	DE_d!B	DE_o!A	DE_o!B	DE_od!A	DE_od!B	DE_odn!A	DE_odn
0	GC_00000001	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0
1	GC_00000002	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0
2	GC_00000003	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0
3	GC_00000004	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0
4	GC_00000005	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0
...	...	...	...	...	...	...	...	...	
35698	GC_00035699	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0
36198	GC_00036199	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0
36467	GC_00036468	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0
36530	GC_00036531	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0
37029	GC_00037030	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0

3924 rows × 9 columns

In [9]:
```
temp_df = pd.merge(pan_split_geneclusters_df,sig_df,on='gene_caller_id')
```

In [10]:
```
temp_df
```

Out[10]:

	entry_id	gene_caller_id	gene_cluster_id	genome_name	alignment_summ
0	156	2789	GC_00000001	M_sp__C18_GCF_001924925	-|140|26|1|11|1|
1	156	2789	GC_00000001	M_sp__C18_GCF_001924925	-|140|26|1|11|1|
2	457	544	GC_00000001	M_sp__C18_GCF_001924925	|6|84|2|20|8|14|1|31|1|11|1|
3	497	2331	GC_00000002	M_sp__C18_GCF_001924925	-|65|34|65|23|65|328|8|1
4	909	4198	GC_00000003	M_sp__C18_GCF_001924925	-|2|11|7|132|2|2|2|3
...	...	...	...	...	
565	417097	3173	GC_00014438	M_sp__C18_GCF_001924925	.|130|38
566	421022	2516	GC_00016207	M_sp__C18_GCF_001924925	.
567	424030	1667	GC_00017711	M_sp__C18_GCF_001924925	.|2
568	425329	2700	GC_00018360	M_sp__C18_GCF_001924925	.
569	441939	384	GC_00032784	M_sp__C18_GCF_001924925	I

570 rows × 16 columns

In [15]:
```
temp_df.to_csv(r'/Users/tito_miniconda/JOYE_LAB_ANVIO_PROJECTS/SK_BACKUP/p28_pangenomes/Marinobacter/Marinobacter_DE_w_description.txt', index = False, sep='\t')
```

In [11]:
```
for index,row in DE_for_anvio_df.iterrows():
    #print(row[0])
    #my_genecluster_id = 'GC_00001516'
    my_genecluster_id = row[0]
    temp2_df = temp_df[temp_df.gene_cluster_id == my_genecluster_id]
    if not temp2_df.empty:
        up_df = temp2_df[temp2_df.log2FoldChange > 0]
        if not up_df.empty:
            for index2,row2 in up_df.iterrows():
                if row2[12] == 'd':
                    DE_for_anvio_df.at[index,'DE_d!B'] = DE_for_anvio_df
.loc[index,'DE_d!B'] + 1
                if row2[12] == 'o':
                    DE_for_anvio_df.at[index,'DE_o!B'] = DE_for_anvio_df
.loc[index,'DE_o!B'] + 1
                if row2[12] == 'od':
                    DE_for_anvio_df.at[index,'DE_od!B'] = DE_for_anvio_d
f.loc[index,'DE_od!B'] + 1
                if row2[12] == 'odn':
                    DE_for_anvio_df.at[index,'DE_odn!B'] = DE_for_anvio_
df.loc[index,'DE_odn!B'] + 1
        down_df = temp2_df[temp2_df.log2FoldChange < 0]
        if not down_df.empty:
            for index2,row2 in down_df.iterrows():
                if row2[12] == 'd':
                    DE_for_anvio_df.at[index,'DE_d!A'] = DE_for_anvio_df
.loc[index,'DE_d!A'] + 1
                if row2[12] == 'o':
                    DE_for_anvio_df.at[index,'DE_o!A'] = DE_for_anvio_df
.loc[index,'DE_o!A'] + 1
                if row2[12] == 'od':
                    DE_for_anvio_df.at[index,'DE_od!A'] = DE_for_anvio_d
f.loc[index,'DE_od!A'] + 1
                if row2[12] == 'odn':
                    DE_for_anvio_df.at[index,'DE_odn!A'] = DE_for_anvio_
df.loc[index,'DE_odn!A'] + 1
```

In [12]:
```
DE_for_anvio_df.to_csv(r'/Users/tito_miniconda/JOYE_LAB_ANVIO_PROJECTS/S
K_BACKUP/p28_pangenomes/Marinobacter/Marinobacter_DE_for_anvio.txt', ind
ex = False, sep='\t')
```

In [17]:
```
!anvi-import-misc-data Marinobacter_DE_for_anvio.txt -p SPLIT_PANs/T_sig
nals/PAN.db --target-data-table items --just-do-it
```

```
New data for 'items' in data group 'default'
================================================
Data key "DE_d!A" ..............................: Predicted type: stacked
bar
Data key "DE_d!B" ..............................: Predicted type: stacked
bar
Data key "DE_o!A" ..............................: Predicted type: stacked
bar
Data key "DE_o!B" ..............................: Predicted type: stacked
bar
Data key "DE_od!A" .............................: Predicted type: stacked
bar
Data key "DE_od!B" .............................: Predicted type: stacked
bar
Data key "DE_odn!A" ............................: Predicted type: stacked
bar
Data key "DE_odn!B" ............................: Predicted type: stacked
bar


WARNING
================================================
The following keys in your data dict will replace the ones that are alr
eady in
your pan database items table and default data group: DE_d!A, DE_d!B, D
E_o!A,
DE_o!B, DE_od!A, DE_od!B, DE_odn!A, DE_odn!B.


WARNING
================================================
Data from the table 'items' for the following data keys in data group
 'default'
removed from the database: 'DE_d!A, DE_d!B, DE_o!A, DE_o!B, DE_od!A, DE
_od!B,
DE_odn!A, DE_odn!B'. #SAD.


NEW DATA
================================================
Database .......................................: pan
Data group .....................................: default
Data table .....................................: items
New data keys ..................................: DE_d!A, DE_d!B, DE_o!A,
DE_o!B, DE_od!A, DE_od!B, DE_odn!A, DE_odn!B.
```

In [33]:
```
!anvi-import-state -p SPLIT_PANs/T_signals/PAN.db --state pan-state.json
--name default
```

```
WARNING
===============================================
Previous entries for "default" is being removed from "states"

Done ........................................: State "default" is adde
d to the database
```

In [30]:
```
!anvi-export-items-order -p SPLIT_PANs/T_signals/PAN.db --name frequency
-o order_items.txt
```

```
Database ....................................: SPLIT_PANs/T_signals/PA
N.db
Database type ...............................: pan
Order name ..................................: frequency
Order data type .............................: newick
Output file .................................: order_items.txt
```

In [34]:
```
#cat order_items.txt | tr "(" "\n" | tr "," "\n" | sed -e '/^$/d' | cut
 -d':' -f1 > order_items_v2.txt
#cat order_items.txt | tr "(" "\n" | tr "," "\n" | sed -e '/^$/d' | cut
 -d':' -f1 | awk '{print "        \"" $0 "\","}' > order_items_v3.txt
#cat order_items.txt | tr "(" "\n" | tr "," "\n" | sed -e '/^$/d' | cut
 -d':' -f1 | awk '{print "        \"" $0 "\","}' | sed -e 's/,/: \{%
\"color\": \"#000000\",%            \"height\": \"50\",%            \"ma
rgin\": \"15\",%            \"type\": \"bar\",%            \"color-start
\": \"#FFFFFF\"%        },/' | tr "%" "\n" > order_items_v4.txt
```

In [48]:
```
!anvi-export-state -p SPLIT_PANs/T_signals/PAN.db -o pan_state_v_1.json
 -s version_0
```

```
Output ......................................: pan_state_v_1.json
```

In [37]:
```
!anvi-delete-misc-data -p SPLIT_PANs/T_signals/PAN.db --list-available-k
eys
```

```
usage: anvi-delete-misc-data [-h] -p PAN_OR_PROFILE_DB -t NAME
                             [--keys-to-remove KEYS_TO_REMOVE]
                             [--groups-to-remove GROUPS_TO_REMOVE]
                             [--list-available-keys] [--just-do-it]
anvi-delete-misc-data: error: the following arguments are required: -p/
--pan-or-profile-db, -t/--target-data-table
```

In [41]:
```
!anvi-delete-misc-data -p SPLIT_PANs/T_signals/PAN.db -t items --list-available-keys --debug
```

```
DATA KEYS FOR "ITEMS" in 1 DATA GROUP(S)
================================================

* DATA GROUP "default" WITH 69 KEYS
    - DE_d!A (stackedbar, describes 3924 items)
    - DE_d!B (stackedbar, describes 3924 items)
    - DE_o!A (stackedbar, describes 3924 items)
    - DE_o!B (stackedbar, describes 3924 items)
    - DE_od!A (stackedbar, describes 3924 items)
    - DE_od!B (stackedbar, describes 3924 items)
    - DE_odn!A (stackedbar, describes 3924 items)
    - DE_odn!B (stackedbar, describes 3924 items)
    - SCG (int, describes 3924 items)
    - combined_homogeneity_index (float, describes 3924 items)
    - cov_bc_0 (float, describes 3924 items)
    - cov_bc_1 (float, describes 3924 items)
    - cov_bc_2 (float, describes 3924 items)
    - cov_bc_3 (float, describes 3924 items)
    - cov_bc_4 (float, describes 3924 items)
    - cov_d_0 (float, describes 3924 items)
    - cov_d_1 (float, describes 3924 items)
    - cov_d_2 (float, describes 3924 items)
    - cov_d_3 (float, describes 3924 items)
    - cov_d_4 (float, describes 3924 items)
    - cov_o_0 (float, describes 3924 items)
    - cov_o_1_1 (float, describes 3924 items)
    - cov_o_1_2 (float, describes 3924 items)
    - cov_o_2 (float, describes 3924 items)
    - cov_o_3 (float, describes 3924 items)
    - cov_o_4_1 (float, describes 3924 items)
    - cov_o_4_2 (float, describes 3924 items)
    - cov_od_0 (float, describes 3924 items)
    - cov_od_1_1 (float, describes 3924 items)
    - cov_od_1_2 (float, describes 3924 items)
    - cov_od_2 (float, describes 3924 items)
    - cov_od_3 (float, describes 3924 items)
    - cov_od_4_1 (float, describes 3924 items)
    - cov_od_4_2 (float, describes 3924 items)
    - cov_odn_0 (float, describes 3924 items)
    - cov_odn_1 (float, describes 3924 items)
    - cov_odn_4 (float, describes 3924 items)
    - det_bc_0 (float, describes 3924 items)
    - det_bc_1 (float, describes 3924 items)
    - det_bc_2 (float, describes 3924 items)
    - det_bc_3 (float, describes 3924 items)
    - det_bc_4 (float, describes 3924 items)
    - det_d_0 (float, describes 3924 items)
    - det_d_1 (float, describes 3924 items)
    - det_d_2 (float, describes 3924 items)
    - det_d_3 (float, describes 3924 items)
    - det_d_4 (float, describes 3924 items)
    - det_o_0 (float, describes 3924 items)
    - det_o_1_1 (float, describes 3924 items)
    - det_o_1_2 (float, describes 3924 items)
    - det_o_2 (float, describes 3924 items)
    - det_o_3 (float, describes 3924 items)
```

```
- det_o_4_1 (float, describes 3924 items)
- det_o_4_2 (float, describes 3924 items)
- det_od_0 (float, describes 3924 items)
- det_od_1_1 (float, describes 3924 items)
- det_od_1_2 (float, describes 3924 items)
- det_od_2 (float, describes 3924 items)
- det_od_3 (float, describes 3924 items)
- det_od_4_1 (float, describes 3924 items)
- det_od_4_2 (float, describes 3924 items)
- det_odn_0 (float, describes 3924 items)
- det_odn_1 (float, describes 3924 items)
- det_odn_4 (float, describes 3924 items)
- functional_homogeneity_index (float, describes 3924 items)
- geometric_homogeneity_index (float, describes 3924 items)
- max_num_paralogs (int, describes 3924 items)
- num_genes_in_gene_cluster (int, describes 3924 items)
- num_genomes_gene_cluster_has_hits (int, describes 3924 items)
```

In [50]:
```
!anvi-import-state -p SPLIT_PANs/T_signals/PAN.db --state pan_state_v_1.
json --name default
```

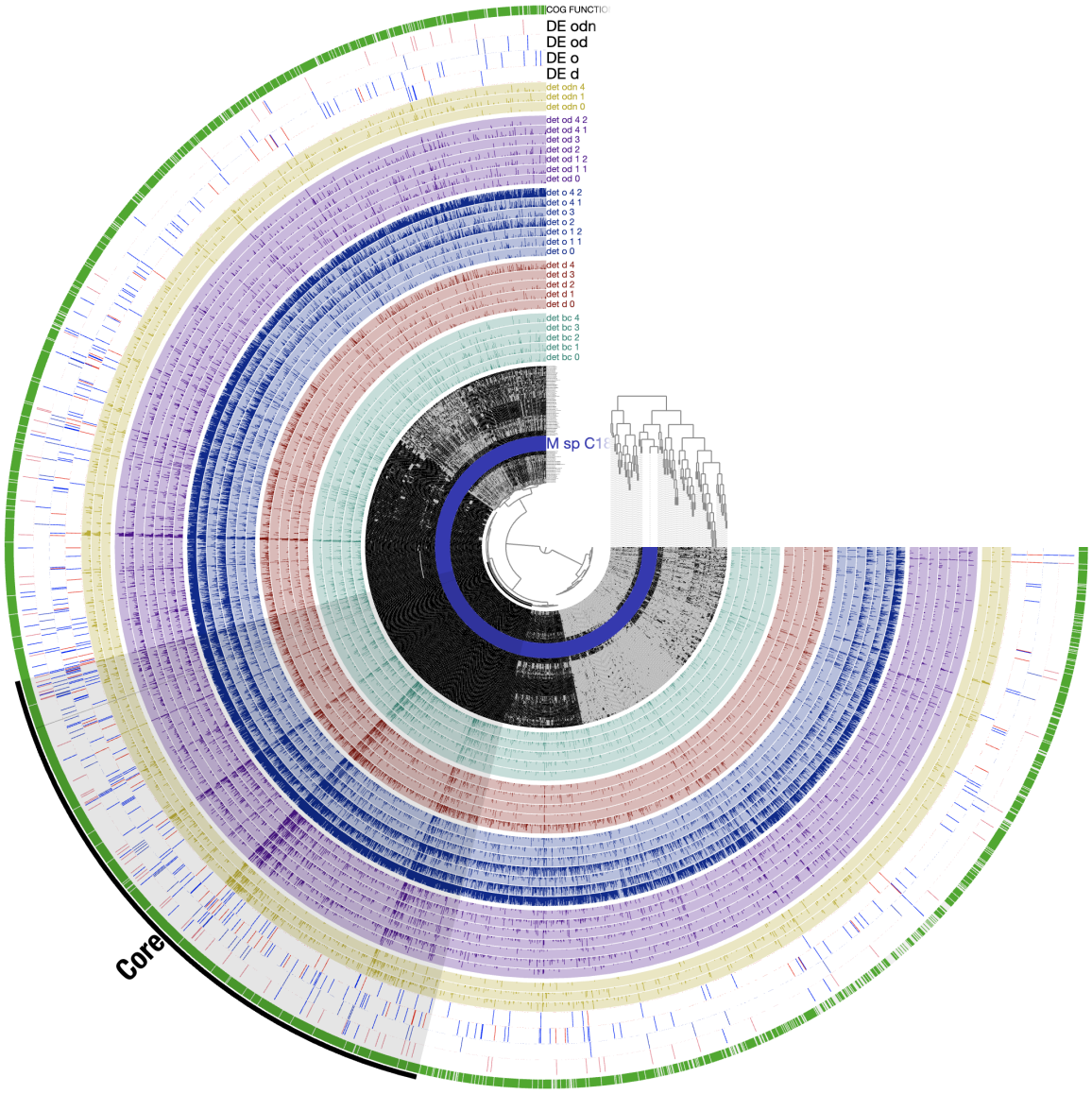
```
WARNING
================================================
Previous entries for "default" is being removed from "states"

Done ........................................: State "default" is adde
d to the database
```

This is the current stage of the figure.

In [4]:
```
Image(filename='Figure1.png')
```

Out[4]:



Barplot of up down DE genes

The following barplot will be added to the anvio plot. Anvio can add a layer to samples but not to the DE layer. So we do it here and later we add in inkscape.

In [2]:
```
DE_for_anvio_df = pd.read_csv(r'/Users/tito_miniconda/JOYE_LAB_ANVIO_PROJECTS/SK_BACKUP/p28_pangenomes/Marinobacter/Marinobacter_DE_for_anvio.txt', sep = '\t')
col_vec = DE_for_anvio_df.columns
col_vec = col_vec[1:9]
d = col_vec.str.split('!').tolist()
barplot_data_df = pd.DataFrame(d)
barplot_data_df = barplot_data_df.rename(columns={0: "Treatment", 1: "DE_type"})
n = barplot_data_df.shape[0]
d = np.zeros(n)
barplot_data_df['DE_total'] = d.tolist()
for index,row in DE_for_anvio_df.iterrows():
    barplot_data_df['DE_total'] = row[1:9].tolist() + barplot_data_df.DE_total
```

In [3]:
```
barplot_data_df
```

Out[3]:

	Treatment	DE_type	DE_total
0	DE_d	A	29.0
1	DE_d	B	76.0
2	DE_o	A	61.0
3	DE_o	B	179.0
4	DE_od	A	59.0
5	DE_od	B	40.0
6	DE_odn	A	89.0
7	DE_odn	B	37.0

In [4]:
```
barplot_data_df = barplot_data_df.replace(to_replace=r'^A$', value='B_DownReg', regex=True)
barplot_data_df = barplot_data_df.replace(to_replace=r'^B$', value='A_UpReg', regex=True)
```

In [5]:
```
barplot_4_anvio = alt.Chart(barplot_data_df).mark_bar().encode(
    x='Treatment',
    y='DE_total',
    color=alt.Color('DE_type', scale=alt.Scale(domain=['A_UpReg', 'B_DownReg'], range=['blue', 'red']))
).properties(
    width=100,
    height=100
)
```

In [8]:
```
alt.renderers.enable('altair_saver')
barplot_4_anvio.save('barplot_4_anvio.svg', method = 'selenium')
```

Layer of Interproscan annotation

In [2]:
```
my_evalue = 1e-10
```

In [10]:
```
#First we need to export the gene sequences present in
!anvi-get-sequences-for-gene-calls --get-aa-sequences -c 02_CONTIGS/Mari
nobacter_sp__C18_GCF_001924925_1-contigs.db -o ref_genome_genes.faa
```

```
Contigs DB ..................................: Initialized: 02_CONTIG
S/Marinobacter_sp__C18_GCF_001924925_1-contigs.db (v. 14)

WARNING
================================================
You did not provide any gene caller ids. As a result, anvi'o will give
 you back
sequences for every 4593 gene call stored in the contigs database.


[0m
WARNING
================================================
Gene caller IDs 4591, 4592 have empty AA sequences and skipped.

Output ......................................: ref_genome_genes.faa
```

In [5]:
```
#Now we need to select those genes that are in the SPLIT
pan_split_geneclusters_df = pan_split_geneclusters_df[pan_split_geneclus
ters_df.genome_name == 'M_sp__C18_GCF_001924925']
```

In [6]:
```
list_genes_in_split = pan_split_geneclusters_df.gene_caller_id.unique().
tolist()
```

In [7]:
```
with open('list_genes_in_split.txt', 'w') as f:
    for item in list_genes_in_split:
        f.write("%s\n" % item)
```

In [8]:
```
!cut -c 1- list_genes_in_split.txt | xargs -n 1 samtools faidx ref_genom
e_genes.faa > ref_genome_genes_in_split.faa
```

In [10]:
```
#scp ref_genome_genes_in_split.faa  tdp56207@sapelo2.gacrc.uga.edu:/scra
tch/tdp56207/SK_BACKUP/p28_pangenome/Marinobacter/95_interproscan/
#module load Perl/5.20.3-foss-2016b
#perl ~/scripts/splitfasta.pl ref_genome_genes_in_split.faa splitseqs 50
#bring back results
#scp tdp56207@sapelo2.gacrc.uga.edu:/scratch/tdp56207/SK_BACKUP/p28_pang
enome/Marinobacter/95_interproscan/results/tsv_files.tgz .
#mv tsv_files.tgz 95_interproscan/
#tar xzvf 95_interproscan/tsv_files.tgz
#cat tsv_files/splitseqs.*tsv > ref_genome_interproscan.tsv
```

In [17]:
```
#load each output file and process it here
location = r'/Users/tito_miniconda/JOYE_LAB_ANVIO_PROJECTS/SK_BACKUP/p28
_pangenomes/Marinobacter/95_interproscan/ref_genome_intersproscan.tsv'
my_int_df = pd.read_csv(location, sep='\t', header=None, names=['protein
_id', 'md5', 'seq_len', 'source', 'accession', 'description', 'start',
'end', 'evalue', 'status', 'date', 'ipr_accession', 'ipr_description'])
```

In [20]:
```
#Remove NA descriptions
mask1 = isNaN(my_int_df.description)
mask2 = isNaN(my_int_df.ipr_description)
mask3 = mask1 & mask2
mask4 = [not bool for bool in mask3]
my_int_df = my_int_df[mask4]
my_int_df = my_int_df[my_int_df.evalue != '-']
my_int_df['evalue'] = my_int_df['evalue'].astype(float)
my_int_df = my_int_df[my_int_df.evalue < my_evalue]
```

In [21]:
```
genes_w_interproscan = my_int_df.protein_id.unique()
```

In [22]:
```
location = r'/Users/tito_miniconda/JOYE_LAB_ANVIO_PROJECTS/SK_BACKUP/p28
_pangenomes/Marinobacter/Marinobacter_DE_for_anvio.txt'
scheme_df = pd.read_csv(location, sep='\t')
scheme_df = scheme_df[['gene_cluster_id','DE_d!A']]
scheme_df.columns = ['gene_cluster_id','INTERPROSCAN']
scheme_df['INTERPROSCAN'] = scheme_df['INTERPROSCAN'].astype(str)
for index,row in scheme_df.iterrows():
    scheme_df.iloc[index,1]='UNKNOWN'
```

In [23]:
```
for index,row in pan_split_geneclusters_df.iterrows():
    #if gene, which is row[1] is in the genes with annotation hit(s)
    if row[1] in genes_w_interproscan:
        my_index = scheme_df[scheme_df['gene_cluster_id'] == row[2]].ind
ex
        scheme_df.iloc[my_index[0],1] = 'KNOWN'
```

In [24]:
```
scheme_df.to_csv(r'/Users/tito_miniconda/JOYE_LAB_ANVIO_PROJECTS/SK_BACK
UP/p28_pangenomes/Marinobacter/Marinobacter_interpro_for_anvio.txt', ind
ex = False, sep='\t')
```

In [25]:
```
!anvi-import-misc-data Marinobacter_interpro_for_anvio.txt -p SPLIT_PAN
s/T_signals/PAN.db --target-data-table items --just-do-it
```

```
New data for 'items' in data group 'default'
================================================
Data key "INTERPROSCAN" .....................: Predicted type: str


NEW DATA
================================================
Database ....................................: pan
Data group ..................................: default
Data table ..................................: items
New data keys ...............................: INTERPROSCAN.
```

Layer of text for later analysis

In [27]:
```
DE_for_anvio_df = pd.read_csv(r'/Users/tito_miniconda/JOYE_LAB_ANVIO_PRO
JECTS/SK_BACKUP/p28_pangenomes/Marinobacter/Marinobacter_DE_for_anvio.tx
t', sep = '\t')
```

In [29]:
```
#load each output file and process it here
location = r'/Users/tito_miniconda/JOYE_LAB_ANVIO_PROJECTS/SK_BACKUP/p28
_pangenomes/Marinobacter/95_interproscan/ref_genome_intersproscan.tsv'
my_int_df = pd.read_csv(location, sep='\t', header=None, names=['protein
_id', 'md5', 'seq_len', 'source', 'accession', 'description', 'start',
'end', 'evalue', 'status', 'date', 'ipr_accession', 'ipr_description'])
```

In [30]:
```
#Remove NA descriptions
mask1 = isNaN(my_int_df.description)
mask2 = isNaN(my_int_df.ipr_description)
mask3 = mask1 & mask2
mask4 = [not bool for bool in mask3]
my_int_df = my_int_df[mask4]
my_int_df = my_int_df[my_int_df.evalue != '-']
my_int_df['evalue'] = my_int_df['evalue'].astype(float)
my_int_df = my_int_df[my_int_df.evalue < my_evalue]
```

In [31]:
```
genes_w_interproscan = my_int_df.protein_id.unique()
```

In [32]:
```
location = r'/Users/tito_miniconda/JOYE_LAB_ANVIO_PROJECTS/SK_BACKUP/p28
_pangenomes/Marinobacter/Marinobacter_DE_for_anvio.txt'
scheme_df = pd.read_csv(location, sep='\t')
scheme_df = scheme_df[['gene_cluster_id','DE_d!A']]
scheme_df.columns = ['gene_cluster_id','INTERPROSCAN_TEXT']
scheme_df['INTERPROSCAN_TEXT'] = scheme_df['INTERPROSCAN_TEXT'].astype(s
tr)
for index,row in scheme_df.iterrows():
    scheme_df.iloc[index,1]=''
```

In [43]:
```
pan_split_geneclusters_df = pd.read_csv(r'/Users/tito_miniconda/JOYE_LAB_ANVIO_PROJECTS/SK_BACKUP/p28_pangenomes/Marinobacter/SPLIT_PANs/pan_split_gene_clusters.csv')
pan_split_geneclusters_df = pan_split_geneclusters_df[pan_split_geneclusters_df.genome_name == 'M_sp__C18_GCF_001924925']
```

In [45]:
```
sum_col = DE_for_anvio_df['DE_d!A'] + DE_for_anvio_df['DE_d!B'] + DE_for_anvio_df['DE_o!A'] + DE_for_anvio_df['DE_o!B'] + DE_for_anvio_df['DE_od!A'] + DE_for_anvio_df['DE_od!B'] + DE_for_anvio_df['DE_odn!A'] + DE_for_anvio_df['DE_odn!B']
mask5 = sum_col != 0
gene_clusters_w_DE = DE_for_anvio_df[mask5].gene_cluster_id
```

In [46]:
```
for index,row in pan_split_geneclusters_df.iterrows():
    if row[2] in gene_clusters_w_DE.tolist():
        temp = my_int_df[my_int_df.protein_id == row[1]]
        mask1_t = isNaN(temp.description)
        mask2_t = [not bool for bool in mask1_t]
        temp = temp[mask2_t]
        if not temp.empty:
            if isNaN(temp.iat[0,12]):
                my_test = str(temp.iat[0,0]) + '__' + temp.iat[0,5]
            else:
                my_test = str(temp.iat[0,0]) + '__' + temp.iat[0,12]
            my_index = scheme_df[scheme_df['gene_cluster_id'] == row[2]].index
            scheme_df.iloc[my_index[0],1] = my_test
```

In [48]:
```
scheme_df.to_csv(r'/Users/tito_miniconda/JOYE_LAB_ANVIO_PROJECTS/SK_BACKUP/p28_pangenomes/Marinobacter/Marinobacter_interpro_labels_for_anvio.txt', index = False, sep='\t')
```

In [49]:
```
!anvi-import-misc-data Marinobacter_interpro_labels_for_anvio.txt -p SPL
IT_PANs/T_signals/PAN.db --target-data-table items --just-do-it
```

```
New data for 'items' in data group 'default'
===============================================
Data key "INTERPROSCAN_TEXT" .................: Predicted type: str


WARNING
===============================================
The following keys in your data dict will replace the ones that are alr
eady in
your pan database items table and default data group: INTERPROSCAN_TEX
T.


WARNING
===============================================
Data from the table 'items' for the following data keys in data group
 'default'
removed from the database: 'INTERPROSCAN_TEXT'. #SAD.


NEW DATA
===============================================
Database .....................................: pan
Data group ...................................: default
Data table ...................................: items
New data keys ................................: INTERPROSCAN_TEXT.
```

The last thing I would like to add to this diagram is

1- Numbers around the circle pointing to critical genes to discuss in the paper.

2- Empty space would be used to show the an expanded tree

To visualize

```
anvi-display-pan -p SPLIT_PANs/T_signals/PAN.db -g MARINOBACTER_GENOMES.db
```

In [50]:
```
#The following file may be usefull while working in the last version of
 the figure.
!anvi-export-functions -c 02_CONTIGS/Marinobacter_sp__C18_GCF_001924925_
1-contigs.db -o ref_genome_cogs.txt
```

```
Annotation sources ...........................: COG_CATEGORY, COG_FUNCT
ION.                         [0m
Number of annotations reported ...............: 7,318
[0m
Output file ..................................: ref_genome_cogs.txt
```

Selecting and splitting only GC containing DE genes.

Anvio diagram is getting too difficult to label names. So we are going to select and split bins that only contain DE genes. Selection was done using the search box and now we proceed to split again.

In [51]:
```
!anvi-split -p SPLIT_PANs/T_signals/PAN.db -g MARINOBACTER_GENOMES.db -C
DE_Genes -o SPLIT_DE_Genes
```

```
Genomes storage ........................................: Initiali
zed (storage hash: hashcbee1777)8;5;0m ETA: 0s
Num genomes in storage .................................: 113
Num genomes will be used ...............................: 113
Pan DB .................................................: Initiali
zed: SPLIT_PANs/T_signals/PAN.db (v. 13)
Gene cluster homogeneity estimates .....................: Function
al: [YES]; Geometric: [YES]; Combined: [YES]

[0m
* Gene clusters are initialized for all 3924 gene clusters in the datab
ase.


[0m
WARNING
============================================================
Anvi'o is about to start splitting your bins into individual, self-cont
ained
anvi'o profiles. This is quite a tricky operation, and even if it finis
hes
successfully, you must double check everyting in the resulting profiles
to make
sure things worked as expected. Although we are doing our best to test
 all
these, variation between projects make it impossible to be 100% sure.

Collections ............................: The collection "DEFAUL
T" that describes 342 splits and 1 bins has been successfully added to
 the database at "SPLIT_DE_Genes/DE_Genes/PAN.db". Here is a full list
 of the bin names in this collection: ALL_SPLITS.
New items order ........................: "frequency:euclidean:wa
rd" (type newick) has been added to the database...

[0m
WARNING
================================================
Clustering for "frequency:euclidean:ward" is already in the database. I
t will be
replaced with the new content.

New items order ........................: "frequency:euclidean:wa
rd" (type newick) has been added to the database...
New items order ........................: "presence-absence:eucli
dean:ward" (type newick) has been added to the database...
Num bins processed .....................................: 1
Output directory .......................................: /Users/t
ito_miniconda/JOYE_LAB_ANVIO_PROJECTS/SK_BACKUP/p28_pangenomes/Marinoba
cter/SPLIT_DE_Genes
```

```
anvi-display-pan -p SPLIT_DE_Genes/DE_genes/PAN.db -g MARINOBACTER_GENOMES.db
```

KEGG Annotation of DE Genes

```
anvi-get-sequences-for-gene-clusters -p SPLIT_DE_Genes/DE_Genes/PAN.db -g MARINOBACTER_GENOMES.db -o DE_Genes_Split_AA.fa
```

Then we filter only for M. sp. C18

```
cat DE_Genes_Split_AA.fa | tr '\n' '%' | sed -e 's/%>/\'$'\n>/g' | grep --color 'C18' | sed -e 's/%/\'$'\n/g' > DE_Genes_Split_AA_C18.fa
```

KEGG annotation was run in

https://www.genome.jp/kaas-bin/kaas_main (https://www.genome.jp/kaas-bin/kaas_main)

choosing GHOSTZ and Representative for Procaryotes options.

Static HTML output

Now our metapangenome is ready to generate an static HTML output.

```
anvi-summarize -p SPLIT_DE_Genes/DE_genes/PAN.db -g MARINOBACTER_GENOMES.db -o SPLIT_DE_Genes-SUMMARY -C core_v2
```

Supplemetary Data 2

In [3]:
```
sup_data_df_p1 = pd.read_csv(r'/Users/tito_miniconda/JOYE_LAB_ANVIO_PROJECTS/SK_BACKUP/p28_pangenomes/Marinobacter/Marinobacter_DE_w_description.txt', sep = '\t')
```

In [4]:
```
sup_data_df_p1
```

Out[4]:

	entry_id	gene_caller_id	gene_cluster_id	genome_name	alignment_summ
0	156	2789	GC_00000001	M_sp__C18_GCF_001924925	-|140|26|1|11|1|
1	156	2789	GC_00000001	M_sp__C18_GCF_001924925	-|140|26|1|11|1|
2	457	544	GC_00000001	M_sp__C18_GCF_001924925	|6|84|2|20|8|14|1|31|1|11|1|
3	497	2331	GC_00000002	M_sp__C18_GCF_001924925	-|65|34|65|23|65|328|8|1
4	909	4198	GC_00000003	M_sp__C18_GCF_001924925	-|2|11|7|132|2|2|2|3
...	...	...	...	...	
565	417097	3173	GC_00014438	M_sp__C18_GCF_001924925	.|130|38
566	421022	2516	GC_00016207	M_sp__C18_GCF_001924925	.
567	424030	1667	GC_00017711	M_sp__C18_GCF_001924925	.|2
568	425329	2700	GC_00018360	M_sp__C18_GCF_001924925	.
569	441939	384	GC_00032784	M_sp__C18_GCF_001924925	I

570 rows × 16 columns

In [5]:
```
sup_data_df_p2 = pd.read_csv(r'/Users/tito_miniconda/JOYE_LAB_ANVIO_PROJECTS/SK_BACKUP/p28_pangenomes/Marinobacter/97_FIGURES/NameReferences_v2.txt', sep = '\t')
```

In [6]:
```
sup_data_df = pd.merge(sup_data_df_p1, sup_data_df_p2, on='gene_caller_id')
```

In [7]:
```
sup_data_df
```

Out[7]:

	entry_id	gene_caller_id	gene_cluster_id	genome_name	alignment_summ
0	156	2789	GC_00000001	M_sp__C18_GCF_001924925	-|140|26|1|11|1|
1	156	2789	GC_00000001	M_sp__C18_GCF_001924925	-|140|26|1|11|1|
2	457	544	GC_00000001	M_sp__C18_GCF_001924925	|6|84|2|20|8|14|1|31|1|11|1|
3	497	2331	GC_00000002	M_sp__C18_GCF_001924925	-|65|34|65|23|65|328|8|1
4	909	4198	GC_00000003	M_sp__C18_GCF_001924925	-|2|11|7|132|2|2|2|3
...	...	...	...	...	
492	408091	2512	GC_00011507	M_sp__C18_GCF_001924925	.|89|248|5|1|12|4|7|4|41
493	417097	3173	GC_00014438	M_sp__C18_GCF_001924925	.|130|38
494	417097	3173	GC_00014438	M_sp__C18_GCF_001924925	.|130|38
495	424030	1667	GC_00017711	M_sp__C18_GCF_001924925	.|2
496	441939	384	GC_00032784	M_sp__C18_GCF_001924925	I

497 rows × 22 columns

In [8]:
```
sup_data_df.to_csv(r'/Users/tito_miniconda/JOYE_LAB_ANVIO_PROJECTS/SK_BACKUP/p28_pangenomes/Marinobacter/97_FIGURES/Marinobacter_supplementary_data_2.txt', index = False, sep='\t')
```

In [ ]:
```

```