Additional Reproducibility Efforts

Importing necessary library for data analysis for .csv and .tsv files; `pandas` & `DataFrame` All the supplementary tables (STable*) dowloaded from the Zenedo repository [1] of the original article [2].

> [1] A. Gavriilidou, "Compendium of specialized metabolite biosynthetic diversity encoded in bacterial genomes". Zenodo, Apr. 15, 2022. (https://doi.org/10.5281/zenodo.5159210).

> [2] Gavriilidou, A., Kautsar, S.A., Zaburannyi, N. et al. Compendium of specialized metabolite biosynthetic diversity encoded in bacterial genomes. Nat Microbiol 7, 726–735 (2022). (https://doi.org/10.1038/s41564-022-01110-2)

In [1]:
```
import pandas as pd
from pandas import DataFrame
```

In [2]:
```
STable1: DataFrame = pd.read_csv("STable1_all_genomes_info.tsv", sep="|")
STable2: DataFrame = pd.read_csv("STable2_BiG-SLICE_t0.4_GCF_assignment.csv"
STable3: DataFrame = pd.read_csv("STable3_BiG-SLICE_t0.5_GCF_assignment.csv"
STable4: DataFrame = pd.read_csv("STable4_BiG-SLICE_t0.6_GCF_assignment.csv"
STable5: DataFrame = pd.read_csv("STable5_BiG-SLICE_t0.7_GCF_assignment.csv"
```

In [3]:
```
STable1 = STable1.rename(columns={"bgc_ids":"bgc_id"})

STable1["bgc_id"] =  STable1.bgc_id.str.split(",")

STable1 = STable1.explode(column="bgc_id").reset_index(drop=True)

STable1["bgc_id"] = STable1.bgc_id.astype("int64")
```

For every threshhold `pd.merge` applied. Taxonomy information is selected from taxonomy column of each dataframe. The streptomyces genus level is displayed as an output, the following question is asked;

- **From the figure 1_A the streptomyces value counts are known (see Figure_1.ipynb), if the foloowing filtering algorithm utilized do we get the same results?**

Figure 1_A data;

	Genus	*Number of GCFs*	*Threshold (T)*
0	*Streptomyces*	*7294*	*t=0.4*
9	*Streptomyces*	*5720*	*t=0.5*

	Genus	Number of GCFs	Threshold (T)
18	Streptomyces	4360	t=0.6
27	Streptomyces	2889	t=0.7

```
In [4]: thresholds_4_gcfs = pd.merge(STable1, STable2, on='bgc_id')

        taxonomy_4 = thresholds_4_gcfs.taxonomy.str.split(",",expand=True).fillna(""
        taxonomy_4.name = 'taxonomy_4'
        streptomyces_index_4 = taxonomy_4[taxonomy_4[5] == "Streptomyces"].index

        thresholds_4_gcf_count = thresholds_4_gcfs.loc[streptomyces_index_4].gcf_id.
```

```
In [5]: thresholds_5_gcfs = pd.merge(STable1, STable3, on='bgc_id')

        taxonomy_5 = thresholds_5_gcfs.taxonomy.str.split(",",expand=True).fillna(""
        taxonomy_5.name = 'taxonomy_5'
        streptomyces_index_5 = taxonomy_5[taxonomy_5[5] == "Streptomyces"].index

        thresholds_5_gcf_count = thresholds_5_gcfs.loc[streptomyces_index_5].gcf_id.
```

```
In [6]: thresholds_6_gcfs = pd.merge(STable1, STable4, on='bgc_id')

        taxonomy_6 = thresholds_6_gcfs.taxonomy.str.split(",",expand=True).fillna(""
        taxonomy_6.name = 'taxonomy_6'
        streptomyces_index_6 = taxonomy_6[taxonomy_6[5] == "Streptomyces"].index

        thresholds_6_gcf_count = thresholds_6_gcfs.loc[streptomyces_index_6].gcf_id.
```

```
In [7]: thresholds_7_gcfs = pd.merge(STable1, STable5, on='bgc_id')

        taxonomy_7 = thresholds_7_gcfs.taxonomy.str.split(",",expand=True).fillna(""
        taxonomy_7.name = 'taxonomy_7'
        streptomyces_index_7 = taxonomy_7[taxonomy_7[5] == "Streptomyces"].index

        thresholds_7_gcf_count = thresholds_7_gcfs.loc[streptomyces_index_7].gcf_id.
```

```
In [8]: len(thresholds_4_gcf_count),len(thresholds_5_gcf_count),len(thresholds_6_gcf

Out[8]: (8703, 6798, 5136, 3363)
```

When inspected;

Figure 1_A data from the applied algorithm;

Original Data

	Genus	Number of GCFs	Threshold (T)
0	Streptomyces	7294	t=0.4
9	Streptomyces	5720	t=0.5

Reproduced Data

	Genus	Number of GCFs	Threshold (T)
0	Streptomyces	8703	t=0.4
9	Streptomyces	6798	t=0.5

	Genus	Number of GCFs	Threshold (T)
18	Streptomyces	4360	t=0.6
27	Streptomyces	2889	t=0.7

	Genus	Number of GCFs	Threshold (T)
18	Streptomyces	5136	t=0.6
27	Streptomyces	3363	t=0.7

```
In [9]: len(thresholds_4_gcf_count)-7294,len(thresholds_5_gcf_count)-5720,len(thresh
```

```
Out[9]: (1409, 1078, 776, 474)
```

```
In [10]: len(thresholds_4_gcf_count)-len(thresholds_5_gcf_count),len(thresholds_5_gcf
```

```
Out[10]: (1905, 1662, 1773)
```

```
In [11]: 1409-1078, 1078-776, 776-474
```

```
Out[11]: (331, 302, 302)
```

```
In [12]: len(STable2)-len(thresholds_4_gcfs),len(STable3)-len(thresholds_5_gcfs),len(
```

```
Out[12]: (764, 37, 37, 37)
```

When inspected;

Figure 1_A data from the applied algorithm;

Original Data vs Reproduced Data

	Genus	Number of GCFs Differ	Threshold (T)
$diff_1$	Streptomyces	1409	t=0.4
$diff_2$	Streptomyces	1078	t=0.5
$diff_3$	Streptomyces	776	t=0.6
$diff_4$	Streptomyces	474	t=0.7

Extra Operations

Operation	Valu of tl outp
$t_{0.4} - t_{0.5}, t_{0.5} - t_{0.6}, t_{0.6} - t_{0.7}$	(190 166 177
$diff_1 - diff_2, diff_2 - diff_3, diff_3 - diff_4$	(33 302 302
Data Lost From INNER MERGE OPERATION {t=0.4,0.5,0.6,0.7}	(76 37, 3 37

Taxonomy resolution

```
In [27]: def taxonomy_resolution(taxonomy_file: DataFrame) -> DataFrame:

             phylum_taxa  = taxonomy_file[1].value_counts()
```

```
    class_taxa   = taxonomy_file[2].value_counts()
    order_taxa   = taxonomy_file[3].value_counts()
    family_taxa  = taxonomy_file[4].value_counts()
    genus_taxa   = taxonomy_file[5].value_counts()
    species_taxa = taxonomy_file[6].value_counts()

    max_list_number = min(len(phylum_taxa), len(class_taxa), len(order_taxa)

    phylum_taxa  = phylum_taxa[:max_list_number]
    class_taxa   = class_taxa[:max_list_number]
    order_taxa   = order_taxa[:max_list_number]
    family_taxa  = family_taxa[:max_list_number]
    genus_taxa   = genus_taxa[:max_list_number]
    species_taxa = species_taxa[:max_list_number]

    taxonomy_resolution = DataFrame({"phylum_taxa_name":list(phylum_taxa.inc
                          "phylum_taxa_count":list(phylum_taxa.values),
                          "class_taxa_name":list(class_taxa.index),
                          "class_taxa_count":list(class_taxa.values),
                          "order_taxa_name":list(order_taxa.index),
                          "order_taxa_count":list(order_taxa.values),
                          "family_taxa_name":list(family_taxa.index),
                          "family_taxa_count":list(family_taxa.values),
                          "genus_taxa_name":list(genus_taxa.index),
                          "genus_taxa_count":list(genus_taxa.values),
                          "species_taxa_name":list(species_taxa.index),
                          "species_taxa_count":list(species_taxa.values),
                         }
                        )
    taxonomy_resolution.to_csv(f"taxonomy_resolution_{taxonomy_file.name}.cs
    return taxonomy_resolution
```

In [28]:
```
taxonomy_resolution_4 = taxonomy_resolution(taxonomy_file=taxonomy_4)

taxonomy_resolution_5 = taxonomy_resolution(taxonomy_file=taxonomy_5)

taxonomy_resolution_6 = taxonomy_resolution(taxonomy_file=taxonomy_6)

taxonomy_resolution_7 = taxonomy_resolution(taxonomy_file=taxonomy_7)
```

In [29]:
```
taxonomy_resolution_4.head()
```

Out[29]:

	phylum_taxa_name	phylum_taxa_count	class_taxa_name	class_taxa_count	or
0	Proteobacteria	565024	Gammaproteobacteria	507067	I
1	Actinobacteriota	264466	Actinobacteria	262184	
2	Firmicutes	229209	Bacilli	228404	Ps
3	Bacteroidota	32879	Alphaproteobacteria	57906	
4	Firmicutes_A	28999	Bacteroidia	30909	

In [30]:
```
taxonomy_resolution_5.head()
```

Out[30]:

	phylum_taxa_name	phylum_taxa_count	class_taxa_name	class_taxa_count	or
0	Proteobacteria	521947	Gammaproteobacteria	474642	I
1	Actinobacteriota	256165	Actinobacteria	255633	
2	Firmicutes	224467	Bacilli	224467	Ps
3	Bacteroidota	13018	Alphaproteobacteria	47278	
4	Firmicutes_A	11780	Bacteroidia	12755	

In [31]:
```
taxonomy_resolution_6.head()
```

Out[31]:

	phylum_taxa_name	phylum_taxa_count	class_taxa_name	class_taxa_count	or
0	Proteobacteria	521947	Gammaproteobacteria	474642	I
1	Actinobacteriota	256165	Actinobacteria	255633	
2	Firmicutes	224467	Bacilli	224467	Ps
3	Bacteroidota	13018	Alphaproteobacteria	47278	
4	Firmicutes_A	11780	Bacteroidia	12755	

In [32]:
```
taxonomy_resolution_7.head()
```

Out[32]:

	phylum_taxa_name	phylum_taxa_count	class_taxa_name	class_taxa_count	or
0	Proteobacteria	521947	Gammaproteobacteria	474642	E
1	Actinobacteriota	256165	Actinobacteria	255633	
2	Firmicutes	224467	Bacilli	224467	P
3	Bacteroidota	13018	Alphaproteobacteria	47278	
4	Firmicutes_A	11780	Bacteroidia	12755	

As it can be seen in the above outputs when dealing with `MAG`'s it is likely to get low taxonomic resolution resolution.

The first dataset has `RefSeq + MAG` datasets, others have just `RefSeq` sequences. This influences the taxonomic resolution.

In `species level` resolution merged dataset has `115,269.00` missing information in the other dataset it is just `52,225.00`.

In [16]:
```
def get_unique_gcf_count(taxonomy_file: DataFrame, gcfs: DataFrame) -> DataF
    genus_resolution = taxonomy_file[5].value_counts().index
    results = {}

    for genus in genus_resolution:
        genus_gcfs = gcfs.loc[taxonomy_file[taxonomy_file[5] == genus].index
        results[f"{genus}"] = [len(genus_gcfs), genus_gcfs.index, genus_gcfs

    df: DataFrame = pd.DataFrame.from_dict(results)
    df_t = df.transpose()

    df_t = df_t.rename({0:"gcf_count_unique", 1:"gcf_index", 2:"gcf_count"},
    df_t.to_csv(f"reproducibility_results_{taxonomy_file.name}.csv", index_l

    df_t_sorted = df_t.sort_values("gcf_count_unique", ascending=False)
    df_t_sorted.to_csv(f"reproducibility_results_{taxonomy_file.name}_sorted

    return df_t_sorted
```

In [17]:
```
taxonomy_4_results = get_unique_gcf_count(taxonomy_file=taxonomy_4, gcfs=thr

taxonomy_5_results = get_unique_gcf_count(taxonomy_file=taxonomy_5, gcfs=thr

taxonomy_6_results = get_unique_gcf_count(taxonomy_file=taxonomy_6, gcfs=thr

taxonomy_7_results = get_unique_gcf_count(taxonomy_file=taxonomy_7, gcfs=thr
```

In [18]:
```
taxonomy_4_results.head()
```

Out[18]:

	gcf_count_unique	gcf_index	gcf_count
	10820	Index([0, 9257, 17269, 49394, 53101, 1575...	[1254, 662, 249, 215, 211, 209, 202, 201, 182,...
Streptomyces	8703	Index([57228, 49265, 45969, 42802, 883, 925...	[1061, 766, 682, 654, 591, 534, 507, 491, 443,...
Pseudomonas_E	1517	Index([25095, 22671, 37850, 58987, 9257, 2787...	[4282, 2073, 2048, 1498, 1445, 1395, 1330, 120...
Nocardia	1421	Index([51953, 9257, 32744, 17269, 57228, 5000...	[173, 128, 76, 62, 60, 55, 54, 53, 48, 47, 46,...
Micromonospora	1089	Index([49696, 57228, 40767, 12206, 20197, 2830...	[215, 182, 152, 108, 88, 87, 86, 85, 81, 77, 7...

In [19]:
```
taxonomy_5_results.head()
```

Out[19]:

	gcf_count_unique	gcf_index	gcf_count
Streptomyces	6798	Index([40575, 32610, 2288, 584, 12007, 1522...	[2218, 1359, 658, 595, 586, 582, 498, 432, 378...
	3971	Index([12007, 40575, 3610, 2139, 10008, 1365...	[214, 171, 151, 125, 98, 82, 80, 73, 71, 66, 5...
Nocardia	1146	Index([33239, 39769, 40575, 3610, 38810, 2403...	[145, 134, 123, 106, 98, 93, 89, 85, 80, 78, 7...
Pseudomonas_E	1101	Index([13655, 39771, 16179, 25911, 20231, 4248...	[4129, 3239, 2077, 2057, 1503, 1327, 1235, 112...
Amycolatopsis	859	Index([40575, 2027, 2982, 28062, 12537, 2965...	[118, 70, 69, 53, 51, 47, 37, 31, 30, 30, 29, ...

In [20]:
```
taxonomy_6_results.head()
```

Out[20]:

	gcf_count_unique	gcf_index	gcf_count
Streptomyces	5136	Index([31953, 22441, 2101, 18765, 1488, 422...	[1975, 1908, 1468, 1455, 1338, 759, 648, 605, ...
	3110	Index([1816, 22441, 22488, 10420, 1472, 2638...	[273, 262, 174, 169, 146, 145, 113, 112, 95, 8...
Nocardia	890	Index([26384, 22441, 9415, 2101, 31953, 2461...	[233, 171, 168, 168, 159, 157, 148, 142, 127, ...
Pseudomonas_E	881	Index([4697, 30608, 11409, 19537, 1816, 1434...	[4131, 3428, 2095, 2054, 1828, 1370, 1327, 114...
Amycolatopsis	700	Index([22441, 31953, 2101, 30456, 9415, 2052...	[146, 117, 98, 92, 70, 55, 54, 53, 46, 44, 35,...

In [21]:
```
taxonomy_7_results.head()
```

Out[21]:

	gcf_count_unique	gcf_index	gcf_count
Streptomyces	3363	Index([9710, 3703, 15483, 1951, 12708, 88...	[3939, 2396, 1600, 1481, 1455, 1370, 1364, 115...
	2253	Index([1128, 9710, 3703, 710, 15483, 1519...	[506, 394, 312, 236, 236, 191, 190, 184, 176, ...
Pseudomonas_E	694	Index([5326, 21034, 1128, 12881, 17004, 151...	[4129, 3434, 2899, 2045, 1874, 1517, 1483, 146...
Nocardia	669	Index([9710, 15483, 3703, 1128, 5284, 1121...	[602, 297, 267, 244, 175, 163, 158, 154, 133, ...
Amycolatopsis	487	Index([9710, 19820, 15483, 3703, 1951, 651...	[232, 163, 135, 129, 100, 90, 70, 65, 63, 54, ...

END OF ADDITIONAL EFFORTS

In [22]:
```
# thresholds_4_gcfs: np.ndarray = np.array(STable2.gcf_id.unique())
# thresholds_5_gcfs: np.ndarray = np.array(STable3.gcf_id.unique())
# thresholds_6_gcfs: np.ndarray = np.array(STable4.gcf_id.unique())
# thresholds_7_gcfs: np.ndarray = np.array(STable5.gcf_id.unique())
# bgc_ids: DataFrame = STable1.bgc_ids.str.split(",", expand=True)
# bgc_ids = (bgc_ids.apply(pd.to_numeric, downcast="unsigned"))
# bgc_ids[["dataset_name", "AccNo", "taxonomy"]] = STable1[["dataset_name",
# STable2["bgc_id"] = STable2["bgc_id"].astype(np.int64)
# STable2.index = STable2["bgc_id"]
# STable2["taxonomy"] = ""
# from pandarallel import pandarallel
# import os
# pandarallel.initialize(nb_workers=os.cpu_count())
# def add_taxonomy_to_table(gcfs_array: np.ndarray, stable: DataFrame) -> No
#     # print(type(gcfs_array))
#     # print(gcfs_array[5])
#     for gcf in gcfs_array:
#         # print(f"gcf{gcf}\n")
#         bgc_ids_of_gcf: np.ndarray = np.array(STable2[STable2.gcf_id == gc
#         for bgc_id in bgc_ids_of_gcf:
#             # print(f"bgc_id{bgc_id}\n")
#             # print(f"type(bgc_id{type(bgc_id)}\n")
#             bgc_tax = bgc_ids[bgc_ids.eq(bgc_id).any(axis="columns")]["tax
#             stable.loc[bgc_id, "taxonomy"] = bgc_tax
#add_taxonomy_to_table = np.vectorize(add_taxonomy_to_table)

#add_taxonomy_to_table(gcfs_array = thresholds_4_gcfs[-10:], stable = STable
```