

Figure_3_A

February 11, 2024

Figure 3_A We need pandas for reading the excel file and process the information it contains.

We need seaborn and matplotlib for plotting purposes, we choose color palette `n_colors=1` to make the figures exactly same color as the paper

```
[1]: import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import ptitprince as pt
pal = sns.color_palette(n_colors=1)
```

We selected `sheet_name="Fig3 panel a"` and we have all the data.

```
[2]: df = pd.read_excel("Table_3.xlsx", sheet_name="Fig3 panel a")
```

For each label (for Figure 3 A y ticks) the order and the names are like the following; * Phyla * Classes * Orders * Families * Genera * Species

This is why we changed the column names to the respective names and order them accordingly.

```
[3]: df = df.replace({"species": "Species", "genus": "Genera", "family": "Families", "order": "Orders", "class": "Classes", "phylum": "Phyla"})
```

This point is extremely complicated due to the order of the data points it determines the plotting the dots in the figure in order to figure out which order they used we produced in every combination (sorting by each column name) and all of the options do not correlate with the paper's figures. As a conclusion most similar estimate sorting with respect to **Variance** choosen.

```
[4]: df = df.sort_values("Variance").reset_index(drop=True)
```

As discussed in the paper titled;

Allen M, Poggiali D, Whitaker K et al. Raincloud plots: a multi-platform tool for robust data visualization [version 2; peer review: 2 approved]. Wellcome Open Res 2021, 4:63 (<https://doi.org/10.12688/wellcomeopenres.15191.2>)

They used seaborn library and basically using stripplot, half_violinplot and box plot with pointplot option they build the library. For the ease the usage first the seaborn methodology tried.

```
[5]: f, ax = plt.subplots(figsize=(15, 12), dpi=200)
```

```

dy="Rank"
dx="Variance"
ort="h"

kwpoint = dict(capsize = 0., errwidth = 0., zorder = 20)

order_fig = ["Species", "Genera", "Families", "Orders", "Classes", "Phyla"]

ax=sns.stripplot(x = dx,
                 y = dy,
                 data = df,
                 palette = pal,
                 edgecolor = "white",
                 size = 3,
                 jitter = 1,
                 zorder = 0,
                 orient = ort,
                 dodge = False,
                 order=order_fig[::-1])

ax=sns.boxplot( x = dx,
               y = dy,
               data = df,
               color = "black",
               width = .15,
               zorder = 10,\
               showcaps = True,
               boxprops = {'facecolor':'none', "zorder":10},\
               showfliers=True,
               whiskerprops = {'linewidth':2, "zorder":10},\
               saturation = 1,
               orient = ort,
               dodge = False,
               order=order_fig[::-1])

ax = sns.pointplot(x=dx, y=dy, data=df, color='red', ax=ax, dodge = False,
                  ↪orient = ort, **kwpoint, order=order_fig[::-1])

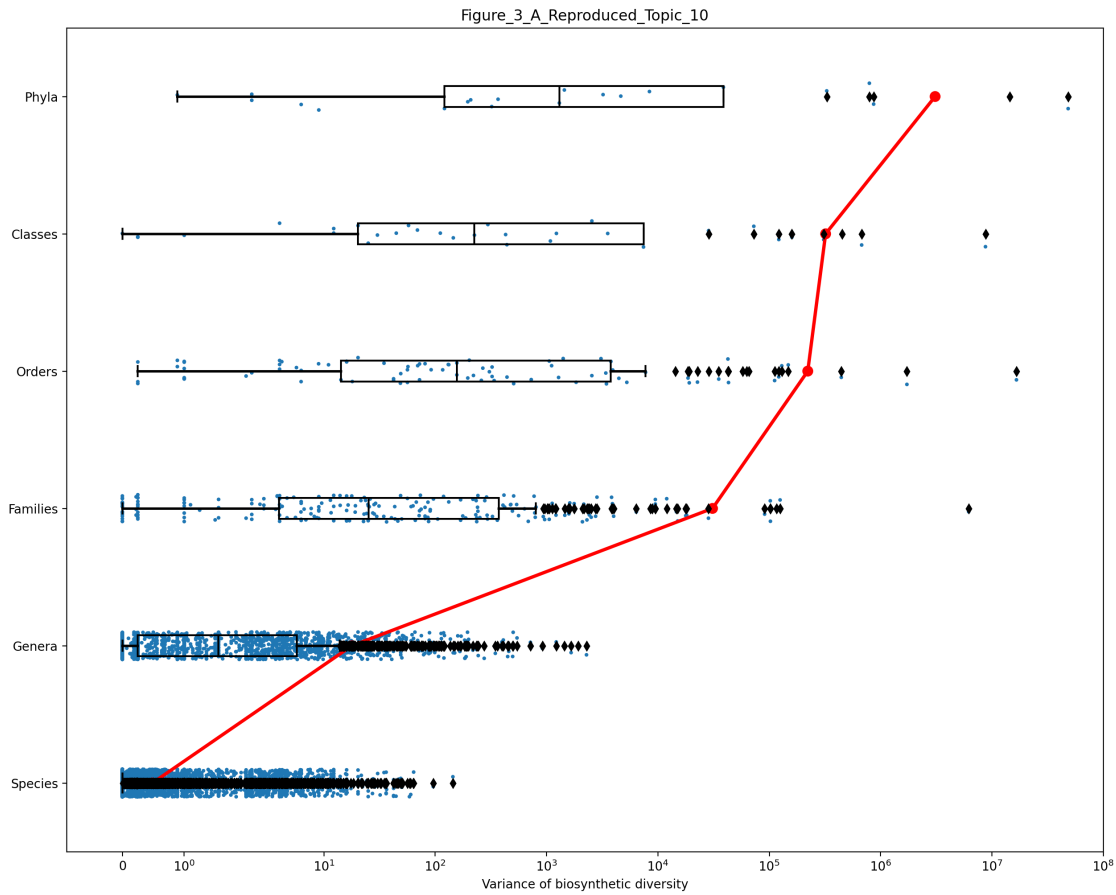
ax.set_xscale("symlog")
# 'linear', 'log', 'symlog', 'asinh', 'logit', 'function', 'functionlog'

ax.set_xlim([-0.9, 1e+8])

ax.set_ylabel(None)
ax.set_xlabel("Variance of biosynthetic diversity")

```

```
plt.title("Figure_3_A_Reproduced_Topic_10");
```



After using seaborn library we use `ptitprince` library and make it one line code with `.RainCloud()` functionality. For not drawing cloud plot;

```
# Draw cloud/half-violin
half_violinplot(x = x, y = y, hue = hue, data = data,
                order = order, hue_order = hue_order,
                orient = orient, width = width_viol,
                inner = None, palette = palette, bw = bw, linewidth = linewidth,
                cut = cut, scale = scale, split = split, offset = offset, ax = ax, **kwcloud)
```

Function Draw cloud/half-violin from the module has been commented out to get the desired result which is not to draw half violin plot. So from the module the following portion altered (743–748: lines);

```
# Draw cloud/half-violin
# half_violinplot(x = x, y = y, hue = hue, data = data,
#                 order = order, hue_order = hue_order,
```

```

#             orient = orient, width = width_viol,
#             inner = None, palette = palette, bw = bw, linewidth = linewidth,
#             cut = cut, scale = scale, split = split, offset = offset, ax = ax, **kwcloud.

```

```

[6]: f, ax = plt.subplots(figsize=(15, 12), dpi=200)

ax=pt.RainCloud(x = dy,
                y = dx,
                data = df,
                palette = pal,
                width_box = .21,
                ax = ax,
                orient = ort,
                pointplot = True,
                move=0.24,
                order=order_fig[::-1])

ax.set_xscale("symlog")

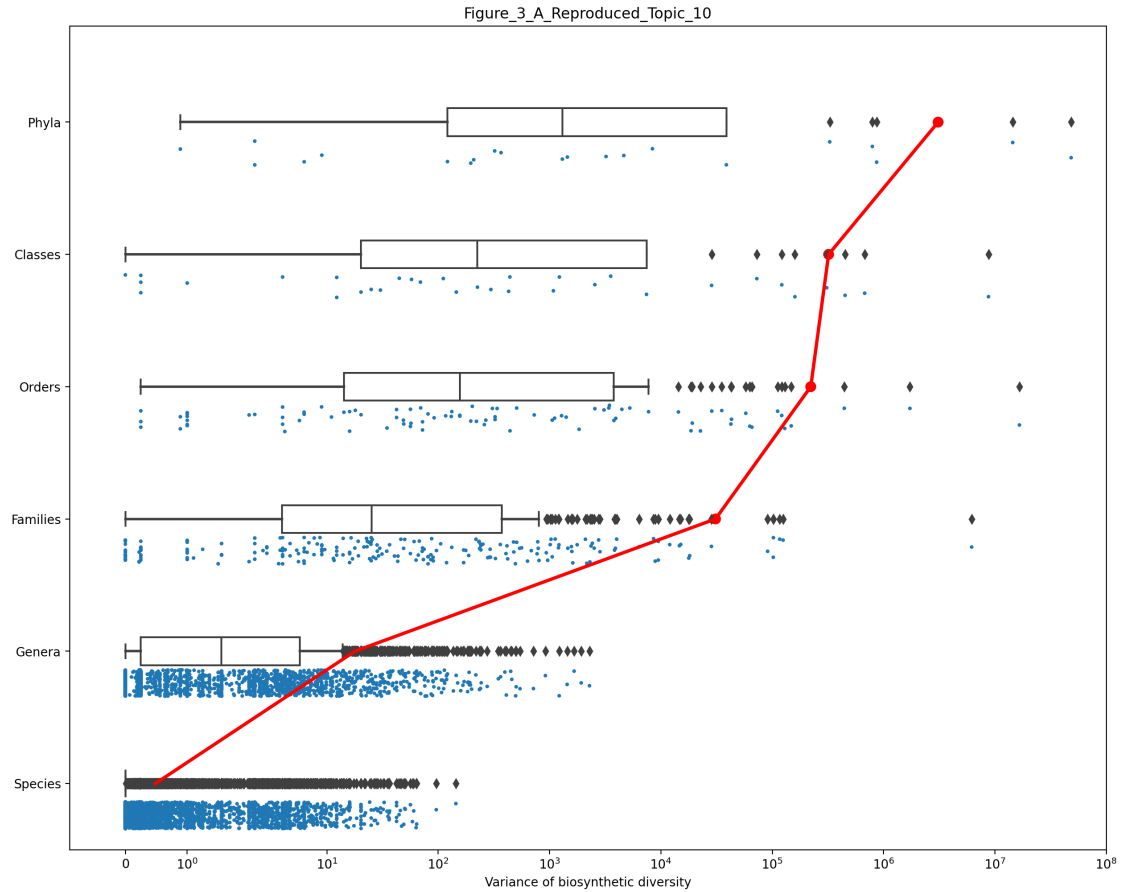
ax.set_xlim([-0.9, 1e+8])

ax.set_ylabel(None)
ax.set_xlabel("Variance of biosynthetic diversity")

plt.title("Figure_3_A_Reproduced_Topic_10");

plt.savefig("Figures_3_Original_Format/output_figure_3_A_Topic_10.png", dpi=300)

```



END OF REPRODUCING FIGURE 3 A. From here until the end are the efforts and tried steps that is not included and necessary for reproducing figure 3 A. (It is always better not to delete code rather commenting.)

```
[7]: # f, ax = plt.subplots(figsize=(12, 11))

# dx="Rank"; dy="Variance"; ort="h"

# ax=sns.stripplot(data=df,
#                  palette="Set2",
#                  edgecolor="white",
#                  size=2,
#                  orient=ort, \
#                  x=dx,
#                  y=dy,
#                  jitter=1,
#                  zorder=0)

# ax=sns.boxplot(data=df,
```

```

#             color="black",
#             orient=ort,
#             width=.15,
#             x=dx,
#             y=dy,
#             zorder=10,\
#             showcaps=True,
#             boxprops={'facecolor':'none', "zorder":10},\
#             showfliers=True,
#             whiskerprops={'linewidth':2, "zorder":10},
#             saturation=1)

# #ax = sns.pointplot(x=dx, y=dy, data=ddf,color='red')
# # Finalize the figure
# #ax.set(ylim=(3.5, -.7))
# sns.despine(left=True)

```

```

[8]: # #adding a red line connecting the groups' mean value (useful for longitudinal
      ↪data)
# sigma = .2
# f, ax = plt.subplots(figsize=(7, 5))

# ax=pt.RainCloud(x = dx, y = dy, data = df, palette = pal, bw = sigma,
#                 width_viol = .6, ax = ax, orient = ort,
#                 pointplot = True)

```

```

[9]: # f, ax = plt.subplots(figsize=(15, 9))

# pal = sns.color_palette(n_colors=1)

# dx = "Rank"
# dy = "Variance"
# ort = "h"

# ax = pt.RainCloud(x= df[dx],
#                   y= df[dy],
#                   palette = pal,
#                   width_viol = .1,
#                   width_box = 0.15,
#                   ax = ax,
#                   orient = ort,
#                   move = .18,
#                   pointplot = True,
#                   dodge=True,
#                   offset=1000)

```

```
# ax.set_xlabel("Variance of biosynthetic diversity")
# ax.set_ylabel(None)

# ax.set_xscale('log')

# plt.show()
```

```
[10]: # f, ax = plt.subplots(figsize=(15, 15), dpi=200)
#
# ax = sns.boxplot(y = df["Rank"], x = df["Variance"], data = df, orient='h')
#
# ax.set_xscale('log')
```

```
[11]: # f, ax = plt.subplots(figsize=(15, 9))
#
# ax=sns.boxplot(x = dx, y = dy, data = df, color = "black", width = .15,
↳zorder = 10,\
#
#         showcaps = True, boxprops = {'facecolor':'none', "zorder":10},\
#         showfliers=True, whiskerprops = {'linewidth':2, "zorder":10},\
#         saturation = 1, orient = ort)
#
# ax=sns.stripplot( x = dx, y = dy, data = df, palette = sns.
↳color_palette(n_colors=1), edgecolor = "white",
#
#         size = 3, jitter = 0, zorder = 0, orient = ort)
#
#
# ax.set_xscale('log')
```

```
[12]: # f, ax = plt.subplots(figsize=(15, 9))
#
# pal = sns.color_palette(n_colors=1)
#
# ax=sns.stripplot(x = dx, y = dy, data = df, palette = pal, edgecolor =
↳"white",
#
#         size = 3, jitter = 1, zorder = 0, orient = ort, ax = ax )
#
# ax=sns.boxplot(x = dx, y = dy, data = df, color = "black", width = .15,
↳zorder = 10,\
#
#         showcaps = True, boxprops = {'facecolor':'none', "zorder":10},\
#         showfliers=True, whiskerprops = {'linewidth':2, "zorder":10},\
#         saturation = 1, orient = ort,move = .2)
#
# ax.set_xscale('log')
```

```
[13]: # help(pt.RainCloud)
```

```
[14]: # df2["Variance"] = np.log(df2["Variance"])
```

```
[15]: # df2 = df2[df2["Variance"] != df2["Variance"].iloc[-1]]
```

```
[16]: # df2 = df2[df2.Variance != float('-inf')]
```

```
[17]: # df["Variance"].iloc[-1]
```

```
[18]: # help(pd.DataFrame.replace)
```