Rscript04: BB-score analaysis and microbial top winners

Angel Rain-Franco

2025-08-26

Contents

1 Load data

1.1 Load packages

```
rm(list=ls())
library(readxl)
library(ggplot2)
library(stringr)
library(dplyr)
library(ggrepel)
library(rstatix)
library(ggfortify)
library(cowplot)
library(tidyr)
library(scales)
library(ggbreak)
library(moments)
library(ggExtra)
library(purrr)
library(ggtext)
library(MASS)
library(tibble)
library(cooccur)
library(CooccurrenceAffinity)
library(pheatmap)
library(viridis)
library(ComplexHeatmap)
library(circlize)
library(grid)
library(stringr)
library(qgraph)
library(igraph)

cbp1 <- c("red","#E69F00","#0072B2","#009E73","#56B4E9","#F0E442","#100000")
```

1.2 Load rating-data

Loaded dataframe contains Elo-rating estimations (Category="Classic") BB-score estimations (Category="Corrected").

```
load(file = "../Scripts/Elo_MAPdata_from_Script02.RData")
str(df_MAP_Elo)
```

```
## 'data.frame':    1816294 obs. of  5 variables:
##  $ player_id  : chr  "90_1;96_1;97_1;98_1;99_1" "90_1;96_1;97_1;98_1;99_1" "90_1;96_1;97_1;98_1;99_1
##  $ biome      : chr  "animal.3" "plant.2" "freshwater.3" "freshwater.2" ...
##  $ Category   : chr  "Corrected" "Classic" "Corrected" "Classic" ...
##  $ mean_rating: num  802 1000 794 1000 1000 ...
##  $ sd_rating  : num  8.089 0.028 6.7699 0.0505 0.0473 ...
```

1.3 Load taxonomy

```r
# 1) Read the raw file (no header).  Replace "myfile.tsv" with your actual filename.
df_raw <- read.delim("../data/otus.info",
                     header    = T,
                     sep       = "\t",
                     stringsAsFactors = FALSE,
                     quote     = "")  # turn off any quoting behavior
str(df_raw)
```

```
## 'data.frame':    597954 obs. of  19 variables:
##  $ OTU          : chr  "90_17776;96_71281;97_92606;98_125911;99_193128" "90_17776;96_71281;97_92606;9
##  $ Tax          : chr  "Archaea" "Archaea" "Archaea" "Archaea" ...
##  $ SpeciesRep   : chr  "" "" "" "" ...
##  $ SeqCount     : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ GoldCount    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ GenomeCount  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ TypeStrains  : chr  "" "" "" "" ...
##  $ Strains      : chr  "" "" "" "" ...
##  $ Genomes      : chr  "" "" "" "" ...
##  $ GoldSeqs     : chr  "" "" "" "" ...
##  $ Aliases      : chr  "" "" "" "" ...
##  $ GoldHit      : chr  "NR_074195:1..1470" "NR_074195:1..1470" "NR_074195:1..1470" "NR_074195:1..147
##  $ GoldID       : int  725 725 725 725 725 569 569 569 569 569 ...
##  $ GoldScore    : num  0.754 0.754 0.754 0.754 0.754 ...
##  $ RepSpecies   : chr  "" "" "" "" ...
##  $ Taxaname     : chr  "Archaea" "Archaea" "Archaea" "Archaea" ...
##  $ OrigTax      : chr  "Archaea" "Archaea" "Archaea" "Archaea" ...
##  $ RepSequenceID: chr  "KC471280:1..1464" "KC471280:1..1464" "KC471280:1..1464" "KC471280:1..1464" .
##  $ RepSequence  : chr  "TACCCGTTGATCCTGCGGGAGGTCGCTGCTATCAGAATTCGACTTAACTAAGCTAGTTCTGGGGCTTCTTCGGAAG
```

```r
#set taxonomy
taxonomy.table<- df_raw[,c("OTU","Tax","RepSequence")]

taxonomy.table <- df_raw %>%
  dplyr::select(OTU, Tax, RepSequenceID) %>%
  tidyr::separate(
    col   = Tax,
    into  = c("Domain", "Phylum", "Class", "Order", "Family", "Genus", "Species"),
    sep   = ";",
    fill  = "right")

str(taxonomy.table)
```

```
## 'data.frame':    597954 obs. of  9 variables:
##  $ OTU          : chr  "90_17776;96_71281;97_92606;98_125911;99_193128" "90_17776;96_71281;97_92606;9
##  $ Domain       : chr  "Archaea" "Archaea" "Archaea" "Archaea" ...
##  $ Phylum       : chr  NA NA NA NA ...
##  $ Class        : chr  NA NA NA NA ...
##  $ Order        : chr  NA NA NA NA ...
##  $ Family       : chr  NA NA NA NA ...
##  $ Genus        : chr  NA NA NA NA ...
##  $ Species      : chr  NA NA NA NA ...
##  $ RepSequenceID: chr  "KC471280:1..1464" "KC471280:1..1464" "KC471280:1..1464" "KC471280:1..1464" .
```

Cheching results

```
length(unique(df_MAP_Elo$player_id))
```

```
## [1] 124772
```

1.4 Load metadata

```
 df.elo.diversity<-read_xlsx("../data/Supplementary_tables_ms.xlsx",
           sheet="Table S2")

tibble(df.elo.diversity)
```

```
## # A tibble: 24 x 10
##      ‘#‘ MAP_biomes   Life.Style      MAP_biomesFullName   n.samples.initial
##    <dbl> <chr>        <chr>           <chr>                            <dbl>
##  1     1 airborne.1   Free-living     airborne                          1595
##  2     2 animal.1     Host-associated animal-urogenital                29317
##  3     3 animal.2     Host-associated animal-proximalgut              180797
##  4     4 animal.3     Host-associated animal-distalgut                125143
##  5     5 animal.4     Host-associated animal-oral                      37433
##  6     6 animal.5     Host-associated animal-skin                      40488
##  7     7 animal.6     Host-associated animal-respiratory               10360
##  8     8 freshwater.1 Free-living     freshwater-sediments             19946
##  9     9 freshwater.2 Free-living     freshwater-water                 46327
## 10    10 freshwater.3 Free-living     freshwater-biofilm                5067
## # i 14 more rows
## # i 5 more variables: n.samples.1000reads.30TUS <dbl>, gamma.diversity <dbl>,
## #   Elo.coef <dbl>, Elo.coef.error <dbl>, Elo.pvalue <dbl>
```

```
df_MAP_Elo<-merge(df_MAP_Elo,df.elo.diversity[,c(2,3,4)],by.x="biome",by.y="MAP_biomes")
```

2 Microbial performance description

2.1 Classifying OTUs accoding performance

We operationally classified top performers as those OTUs which performance is greater that the biome species pool. We consider the top5% as top performers for overall winners description and top 1% for a shorted list of winners.

```
df_MAP_Elo.corr <- df_MAP_Elo %>%
  filter(Category == "Corrected") %>%
  group_by(biome) %>%
  mutate(
    # --- normalize only by the mean (centered) ---
```

```r
    mean_rating_centered = mean_rating - median(mean_rating, na.rm = TRUE),

    # --- empirical (5%) threshold + flag (on centered values) ---
    thresh_95_emp_centered = stats::quantile(mean_rating_centered, 0.95),
    exceed_95_emp_centered = mean_rating_centered > thresh_95_emp_centered,

    # --- empirical (1%) threshold + flag (on centered values) ---
    thresh_99_emp_centered = stats::quantile(mean_rating_centered, 0.99),
    exceed_99_emp_centered = mean_rating_centered > thresh_99_emp_centered
  ) %>%
  ungroup()
```

2.2 Species performance estimations based on BB-scores

For identification of successful taxa, we focused on OTUs in the top 5% of BB-scores in one or more biomes. For the quantification of the performance of these OTUs across the other biomes, we calculated average BB-scores for all biomes where they did not fall within the top 5% of BB-scores.

```r
# Step 1: Filter only player_ids
df_Elo_corr<-df_MAP_Elo.corr #Just to kepp the original
players_top <- df_Elo_corr%>%
  filter(exceed_95_emp_centered== TRUE) %>% # 5% top winners
  pull(player_id) %>%
  unique()

# Step 2: Filter df to include only those player_ids
filtered_df <- df_Elo_corr %>%
  filter(player_id %in% players_top)

# Step 4: Count and calculate percentages per player_id
df_top<- filtered_df %>%
  dplyr::select(player_id, biome, mean_rating_centered,exceed_95_emp_centered)

summary.tmp95<-df_top

#Step 5: Estimations
RP_summary_otus <- summary.tmp95 %>%
  group_by(player_id) %>%
  summarise(
    # In how many are greater than threshold
    count_BBgreater2 = sum(exceed_95_emp_centered==TRUE, na.rm = TRUE),
    # In how many RP <= threshold
    count_BBsmaller2 = sum(exceed_95_emp_centered==FALSE, na.rm = TRUE),
    # "d1_d9_count" = total biomes (24?) minus those greater than threshold
    biomes_rest = 24 - sum(exceed_95_emp_centered==TRUE, na.rm = TRUE),
    # sum of RP_centered non-top winners
    sum_BBsmaller2 = sum(mean_rating_centered[exceed_95_emp_centered==FALSE], na.rm = TRUE),
    sum_BBall = sum(mean_rating_centered, na.rm = TRUE),

    .groups = "drop")

# Estimating the interbiome-performance
```

```r
RP_summary_otus$AverageRP<-(RP_summary_otus$sum_BBsmaller2)/(RP_summary_otus$count_BBsmaller2+1)

#Biome occurrence
RP_summary_otus$biome.occ<-RP_summary_otus$count_BBgreater2+RP_summary_otus$count_BBsmaller2

##Set occurrence categories
RP_summary_otus <- RP_summary_otus %>%
  mutate(
    biome_occ_class = case_when(
      biome.occ == 24               ~ "Ubiquitous Top5%",  # exactly 24
      biome.occ <  24               ~ "Restricted",
      TRUE                           ~ NA_character_),biome_occ_class = factor(
      biome_occ_class,
      ordered = TRUE))

#Reorder biome levels co-occurrence
RP_summary_otus$biome_occ_class<-factor(RP_summary_otus$biome_occ_class,
                                                c("Ubiquitous Top5%","Restricted"))

#For the OTU that was top winner in 24, we manuall set it the max observed AverageRP
RP_summary_otus$AverageRP[RP_summary_otus$count_BBgreater2==24]<-max(RP_summary_otus$AverageRP)

# Estimate Spearman correlation between top5% occurrence and interbiome performance

res.correlation<-cor.test(~count_BBgreater2+AverageRP,data=RP_summary_otus[RP_summary_otus$biome_occ_cla

res.correlation
```

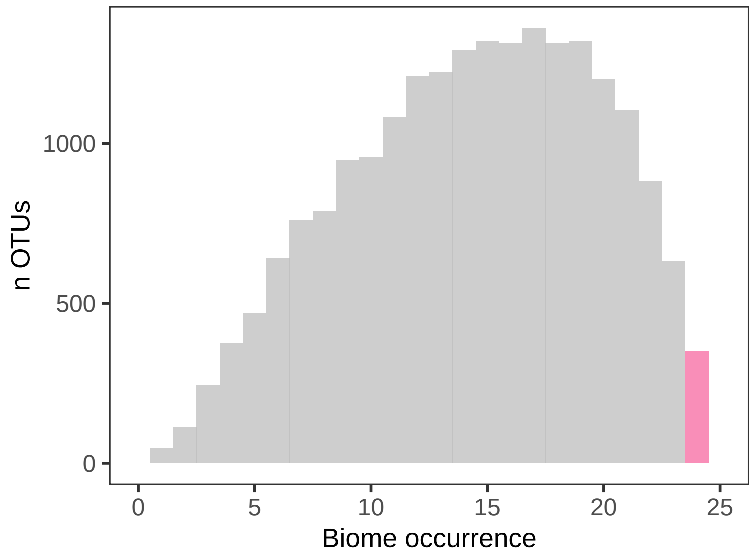
```
##
##  Spearman's rank correlation rho
##
## data:  count_BBgreater2 and AverageRP
## S = 2753960, p-value < 2.2e-16
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##       rho
## 0.6112806
```

```r
### Find the top 1% performers ###

# Step 1: Filter only player_ids
df_Elo_corr<-df_MAP_Elo.corr #Just to kepp the original
players_top <- df_Elo_corr%>%
  filter(exceed_99_emp_centered== TRUE) %>% # 1% top winners
  pull(player_id) %>%
  unique()

# Step 2: Filter df to include only those player_ids
filtered_df <- df_Elo_corr %>%
  filter(player_id %in% players_top)

# Step 4: Count and calculate percentages per player_id
df_top<- filtered_df %>%
  dplyr::select(player_id, biome, mean_rating_centered,exceed_99_emp_centered)

# Save results
summary.tmp99<-df_top

#Step 5: Estimations
RP_summary_otus_99 <- summary.tmp99 %>%
  group_by(player_id) %>%
  summarise(
    # In how many are greater than threshold
    count_BBgreater2 = sum(exceed_99_emp_centered==TRUE, na.rm = TRUE),
    # In how many RP <= threshold
    count_BBsmaller2 = sum(exceed_99_emp_centered==FALSE, na.rm = TRUE),
    # "d1_d9_count" = total biomes (24?) minus those greater than threshold
    biomes_rest = 24 - sum(exceed_99_emp_centered==TRUE, na.rm = TRUE),
    # sum of RP_centered non-top winners
    sum_BBsmaller2 = sum(mean_rating_centered[exceed_99_emp_centered==FALSE], na.rm = TRUE),
    .groups = "drop")
#change names
names(RP_summary_otus_99)<-c("player_id","count_BBgreater2.99","count_BBsmaller2.99","biomes_rest.99","

## Combine both top5% and top1% estimations
RP_summary_otus <- merge(RP_summary_otus,RP_summary_otus_99,by="player_id",all.x=T)
tibble(RP_summary_otus)
```

```
## # A tibble: 20,942 x 13
##    player_id        count_BBgreater2 count_BBsmaller2 biomes_rest sum_BBsmaller2
##    <chr>                       <int>            <int>       <dbl>          <dbl>
##  1 90_1;96_1;97_1;~                1               13          23          0.558
##  2 90_1;96_1;97_35~                3               19          21          3.22
##  3 90_1;96_1115;97~                3               19          21          1.25
##  4 90_1;96_11286;9~                1               12          23          0.310
##  5 90_1;96_11646;9~                2               16          22          2.01
##  6 90_1;96_11925;9~                1               14          23          0.971
##  7 90_1;96_1238;97~                1               19          23          3.33
##  8 90_1;96_1238;97~                1               17          23          1.25
##  9 90_1;96_13029;9~                2               10          22          0.598
## 10 90_1;96_13030;9~                1               20          23          2.32
## # i 20,932 more rows
## # i 8 more variables: sum_BBall <dbl>, AverageRP <dbl>, biome.occ <int>,
## #   biome_occ_class <ord>, count_BBgreater2.99 <int>,
## #   count_BBsmaller2.99 <int>, biomes_rest.99 <dbl>, sum_BBsmaller2.99 <dbl>
```

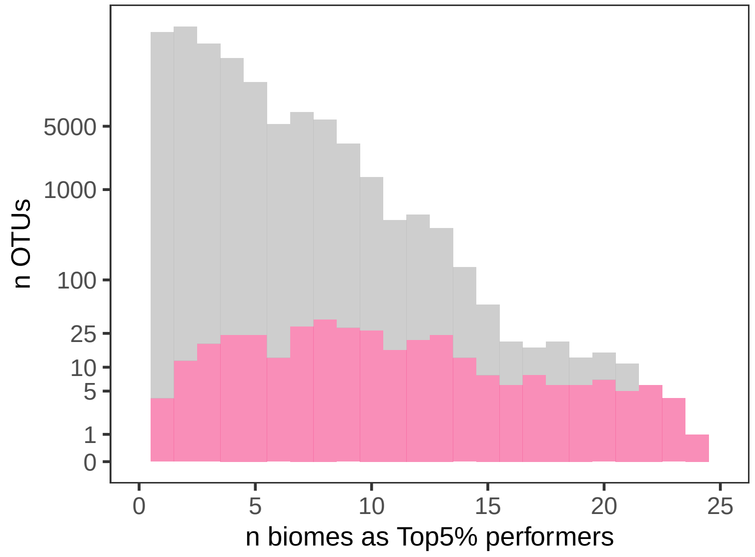
2.2.1 Biome occurrence of top 5% performers (Figure 4B)

```
# Top panel (Occupancy distribution)
df.agg.occ.biomes.V2<- aggregate(player_id~biome.occ ,data=RP_summary_otus, length)

df.agg.occ.biomes.V2<-df.agg.occ.biomes.V2%>%
 mutate(
    biome_occ_class = case_when(
      biome.occ == 24                 ~ "Ubiquitous Top5%",  # exactly 24
      biome.occ <  24                 ~ "Restricted",
      TRUE                             ~ NA_character_),biome_occ_class = factor(
      biome_occ_class,
      ordered = TRUE))

df.agg.occ.biomes.V2$biome_occ_class<-factor(df.agg.occ.biomes.V2$biome_occ_class,
                                              c("Ubiquitous Top5%","Restricted"))

# Plot
bar.plot.top.panel<-ggplot(df.agg.occ.biomes.V2, aes(x=biome.occ,y=player_id,fill=biome_occ_class)) +
  geom_bar(stat="identity",width = 1,alpha=0.75)+
 scale_fill_manual(values=c("#f768a1","grey"),name=NULL)+
  theme_bw()+theme(legend.position = "none",axis.title = element_text(size=10),
                   panel.grid.major = element_blank(),panel.grid.minor = element_blank())+labs(y="n OTUs
bar.plot.top.panel
```



2.2.2 Distribution top 5% performers based on BB-scores (Figure 4B)

```
# Bottom panel (Distribution top winners)
df.agg.occ.biomes.V3<- aggregate(player_id~count_BBgreater2+biome_occ_class ,data=RP_summary_otus, leng

df.agg.occ.biomes.V3$biome_occ_class<-factor(df.agg.occ.biomes.V3$biome_occ_class,
                                              c("Restricted","Ubiquitous Top5%"))

bar.plot.bottom.panel <- ggplot(df.agg.occ.biomes.V3, aes(x = count_BBgreater2, y = player_id+1,fill=bi
    geom_col(width = 1, alpha = 0.75) +
  scale_y_log10(
       breaks = c(0, 1, 5,10, 25, 100,1000, 5000) + 1,  # match shift
      labels = function(b) b - 1) +
    xlim(0, 25) +
    theme_bw() +
    theme(legend.position = "none",axis.title = element_text(size=10),
          panel.grid.major = element_blank(),
          panel.grid.minor = element_blank()) +
    labs(y = expression("n OTUs"), x = "n biomes as Top5% performers")+
   scale_fill_manual(values=c("grey","#f768a1"),name=NULL)
bar.plot.bottom.panel
```



2.3 Merge dataset with taxonomy

```
#set taxonomy
RP_summary_otus_tax<-merge(RP_summary_otus, taxonomy.table,by.x ="player_id",by.y="OTU",all.x=T)

RP_summary_otus_tax <- RP_summary_otus_tax %>%
  mutate(
    # turn empty strings into NA for consistency
    Family = na_if(Family, ""),
    Order  = na_if(Order,  ""),
    Class  = na_if(Class,  ""),

    # build a fallback label for each row:
    Genus = case_when(
      !is.na(Genus) & Genus != ""      ~ Genus,
      !is.na(Family)                   ~ paste0("unc_", Family),
      !is.na(Order)                    ~ paste0("unc_", Order),
      !is.na(Class)                    ~ paste0("unc_", Class),
      TRUE                              ~ NA_character_
    )
  )
```

2.3.1 Principal panel top5% winners versus average BB-score (Figure 4A)

Inter-biome performance (average normalized BB-score across all biomes) of OTUs that were in the top 5% of performers in 1 or more biomes. The subset of ubiquitous microbes detected in all 24 biomes are highlighted in pink. Six ubiquitous microbes were in the top 1% of performers in more than half of the biomes and their average BB-score was >99.9% of all OTUs (in green).

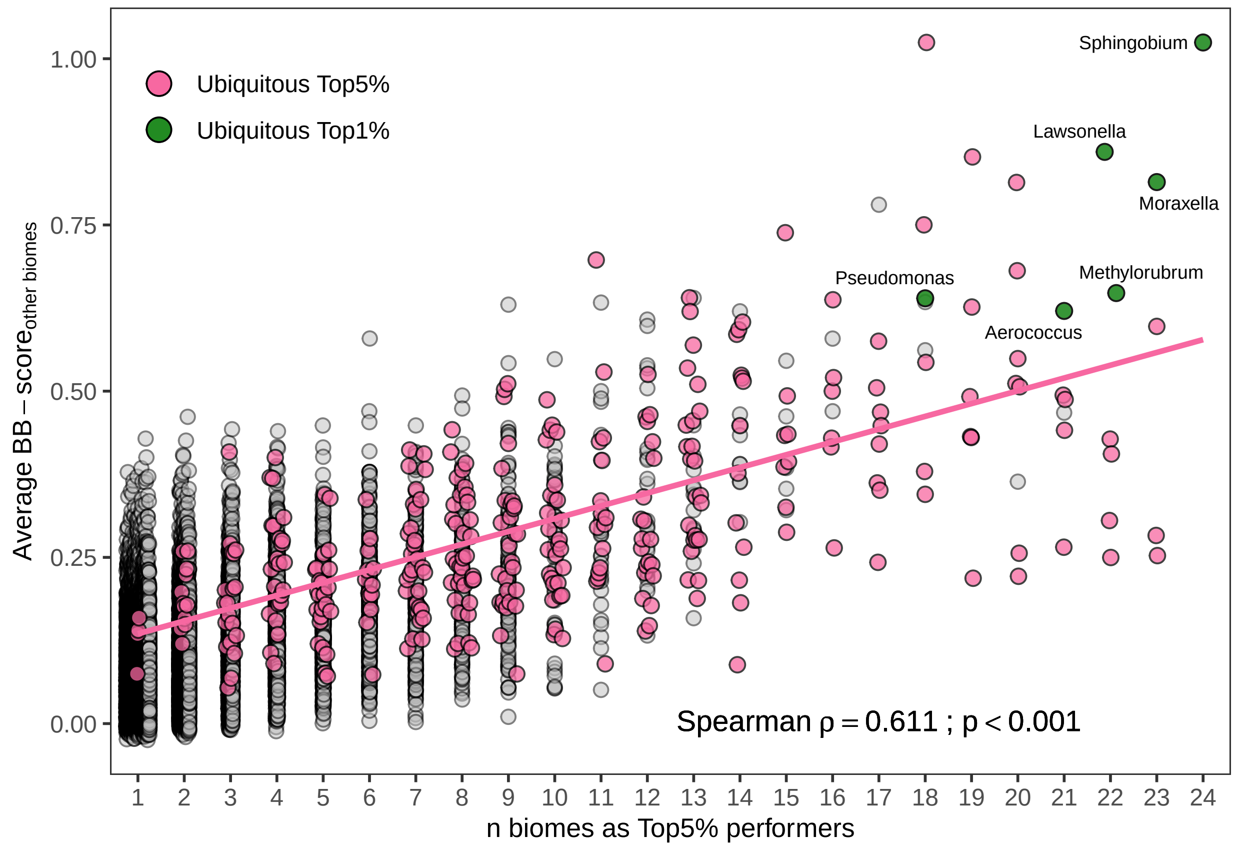
```
# add flag columns for conditions to highlight (Ubiquitous and high performers)
RP_summary_otus_tax2 <- RP_summary_otus_tax %>%
  mutate(
    ring = RP_summary_otus$count_BBgreater2.99 >= 12 & count_BBgreater2.99>=12 & AverageRP> quantile(RP_
    stroke_size = ifelse(ring, 1.1, 0.2)    # adjust as you like
  )


pos <- position_dodge2(width = 0.5, preserve = "single")

RP_summary_otus_tax2$flag.colour<-as.character(RP_summary_otus_tax2$biome_occ_class)
RP_summary_otus_tax2$flag.colour[RP_summary_otus_tax2$ring==TRUE]<-"Ubiquitous Top1%"
RP_summary_otus_tax2$flag.colour<-as.factor(RP_summary_otus_tax2$flag.colour)


##Check correlation between the 2 parameters (Spearman correlation)
res.correlation<-cor.test(~count_BBgreater2+AverageRP,RP_summary_otus_tax2[RP_summary_otus_tax2$biome_o
res.correlation
```

```
##
##  Spearman's rank correlation rho
##
## data:  count_BBgreater2 and AverageRP
## S = 2753960, p-value < 2.2e-16
## alternative hypothesis: true rho is not equal to 0
## sample estimates:
##       rho
## 0.6112806
```

```
principal.mappinng.OTUs.BB <- ggplot(
  RP_summary_otus_tax2,
  aes(x = factor(count_BBgreater2), y = AverageRP, fill = flag.colour)
) +
  # Restricted: plot but don't show in legend
  geom_point(
    data = dplyr::filter(RP_summary_otus_tax2, flag.colour == "Restricted"),
    shape = 21, size = 2.2, alpha = 0.5, color = "black",
    position = pos, show.legend = FALSE
  ) +
  # Ubiquitous
  geom_point(
    data = dplyr::filter(RP_summary_otus_tax2, flag.colour == "Ubiquitous Top5%"),
    shape = 21, size = 2.4, alpha = 0.75, color = "black", position = pos
  ) +
  # Ubiquitous 1%
  geom_point(
    data = dplyr::filter(RP_summary_otus_tax2, flag.colour == "Ubiquitous Top1%"),
    shape = 21, size = 2.5, alpha = 0.9, color = "black", position = pos
  ) +
  scale_fill_manual(
    name = NULL,
    values = c("Restricted" = "grey",
               "Ubiquitous Top5%" = "#f768a1",
               "Ubiquitous Top1%" = "#228B22"),
    breaks = c("Ubiquitous Top5%", "Ubiquitous Top1%")  # <- removes "Restricted" from legend
  ) +
  theme_bw() +
  scale_x_discrete(breaks = as.character(1:24)) +
  labs(x = "n biomes as Top5% performers", y = expression(Average~BB-score["other biomes"])) +
  ggrepel::geom_text_repel(
    data = dplyr::filter(RP_summary_otus_tax2, ring),
    aes(label = Genus), size = 2.5, max.overlaps = Inf,
    box.padding = 0.3, point.padding = 0.2, segment.size = 0.2
  ) +
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),axis.title = element_tex
      legend.position = c(0.01, 0.95), legend.justification = c(0, 1)) +
  geom_smooth(
    data = subset(RP_summary_otus_tax2, biome_occ_class == "Ubiquitous Top5%"),
    aes(count_BBgreater2, AverageRP,colour=biome_occ_class,fill=biome_occ_class),
    se = FALSE, method = "lm", colour = "#f768a1", show.legend = FALSE
  ) +
  guides(fill = guide_legend(override.aes = list(size = 4, alpha = 1)))+
  annotate("text", x=17, y=0.0,
        label=bquote(Spearman~rho==.(round(res.correlation$estimate,3))~"; p"<0.001))
```

```
## Warning: A numeric `legend.position` argument in `theme()` was deprecated in ggplot2
## 3.5.0.
## i Please use the `legend.position.inside` argument of `theme()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
principal.mappinng.OTUs.BB
```



2.4 Get names of the widespread and thos with high interbiome performance

```
names.taxa.top<-filter(RP_summary_otus_tax2, count_BBgreater2>=12& AverageRP>quantile(RP_summary_otus_ta
tibble(names.taxa.top[,1:5])
```

```
## # A tibble: 6 x 5
##   player_id          count_BBgreater2 count_BBsmaller2 biomes_rest sum_BBsmaller2
##   <chr>                         <int>            <int>       <dbl>          <dbl>
## 1 90_10;96_1541;97~                22                2           2           2.58
## 2 90_13;96_938;97_~                21                3           3           2.48
## 3 90_15368;96_177;~                22                2           2           1.94
## 4 90_20;96_774;97_~                24                0           0           0
## 5 90_23;96_61;97_8~                18                6           6           4.48
## 6 90_80;96_1834;97~                23                1           1           1.63
```

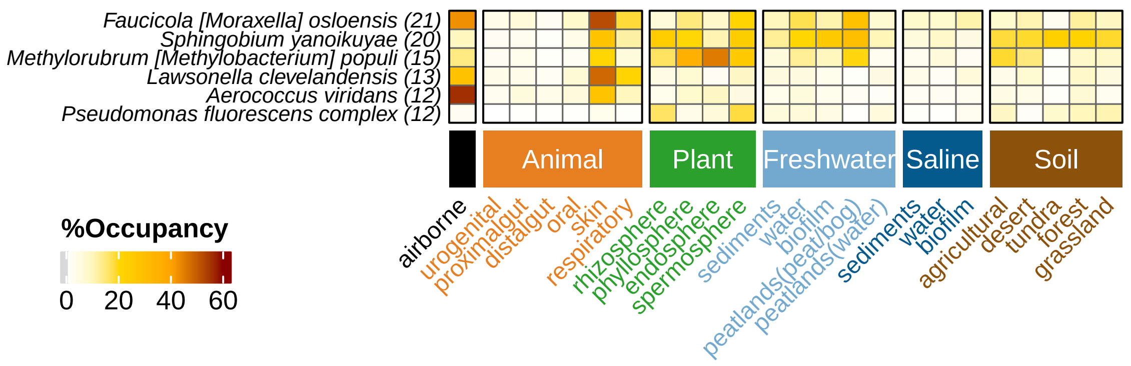
2.4.1 Distribution of the observed top winners across biomes (Figure 4D)

Heatmap of biome-level occupancy (% of samples per biome) of the six top-performing microbes. The numbers in parentheses indicate the number of biomes in which each lineage was ranked within the top 1% of performers. The color bar indicates relative occupancy.

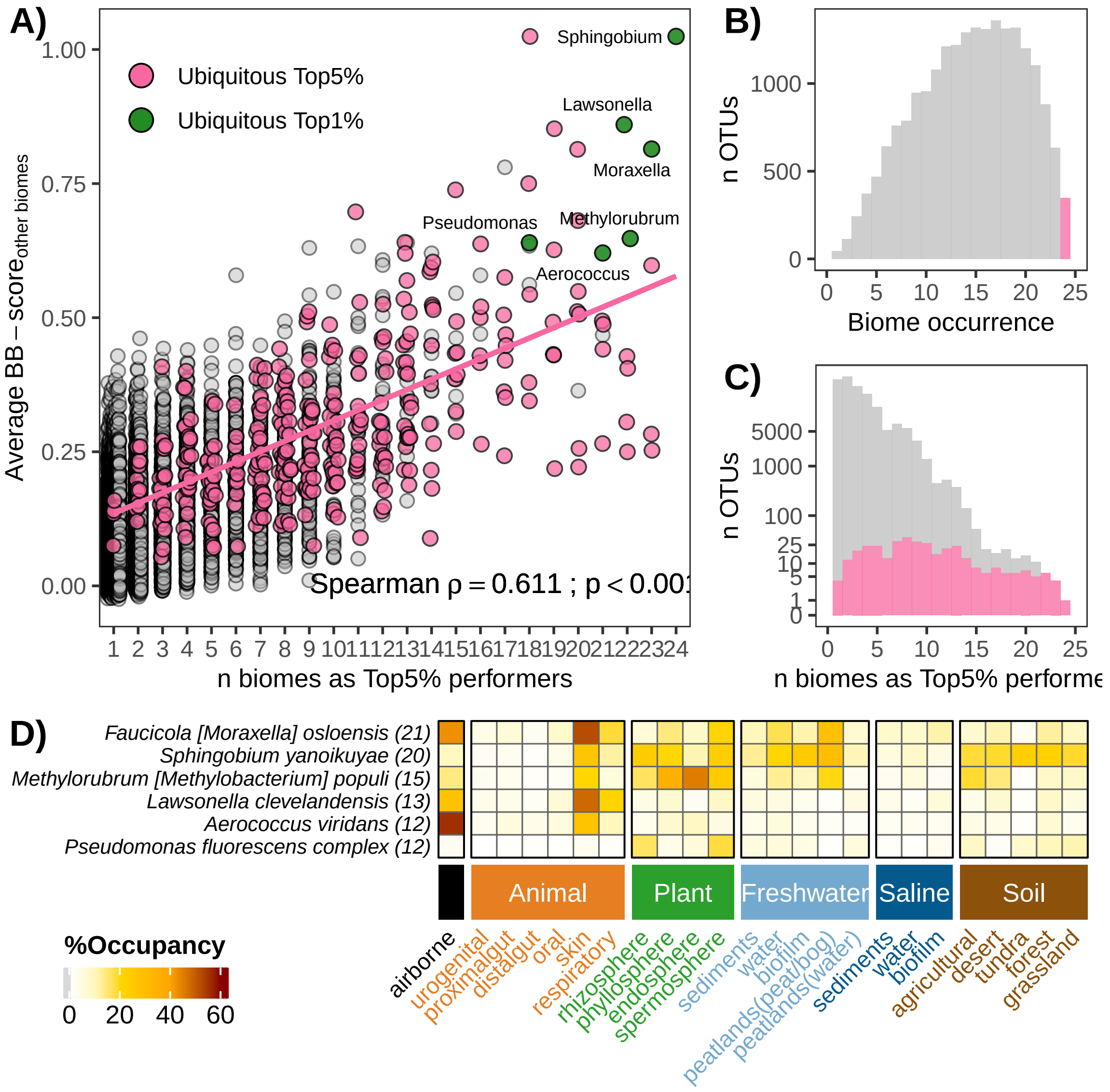
```r
winners.info<-data.frame(read.csv("../data/selected_OTUs_across_biomes.csv"))
#Intermediate step
#names.taxa.top<-filter(RP_summary_otus_tax2, count_BBgreater2>=12& AverageRP>quantile(RP_summary_otus_

## Add names to the winners
names.taxa.top$short.name<-paste0(names.taxa.top$Genus,".",str_split_fixed(names.taxa.top$RepSequenceID
#names$short.name<-str_replace_all(names$short.name,"[:]",".")

names.taxa.top<-merge(names.taxa.top[,c(1,25)],winners.info,by.x="player_id",by.y="otu",all.x=T)
names.taxa.top<-merge(names.taxa.top,df.elo.diversity[,c(2,4)],by.x="biome",by.y="MAP_biomes")

# Sort biomes
levels_sorted<-c("airborne","animal-urogenital","animal-proximalgut","animal-distalgut","animal-oral","
                    "plant-rhizosphere","plant-phyllosphere","plant-endosphere","plant-spermosphere",
                    "freshwater-sediments","freshwater-water","freshwater-biofilm",
"freshwater-peatlands(peat/bog)","freshwater-peatlands(water)",
"saline-sediments","saline-water","saline-biofilm",
"soil-agricultural","soil-desert","soil-tundra","soil-forest","soil-grassland")
names.taxa.top$MAP_biomesFullName<-as.factor(names.taxa.top$MAP_biomesFullName)
names.taxa.top$MAP_biomesFullName<-factor(names.taxa.top$MAP_biomesFullName,levels_sorted)

occ_mat <- names.taxa.top %>%
  dplyr::select(short.name = short.name, MAP_biomesFullName, occurrence_rate) %>%
  group_by(short.name, MAP_biomesFullName) %>%
  summarise(value = mean(occurrence_rate, na.rm = TRUE), .groups = "drop") %>%
  pivot_wider(names_from = MAP_biomesFullName, values_from = value, values_fill = 0) %>%
  column_to_rownames("short.name") %>%
  as.matrix()

row.names(occ_mat)<-c("Aerococcus viridans (12)", "Lawsonella clevelandensis (13)",
                      "Methylorubrum [Methylobacterium] populi (15)",
                      "Faucicola [Moraxella] osloensis (21)", "Pseudomonas fluorescens complex (12)",
                      "Sphingobium yanoikuyae (20)" )

occ_mat.sorted<-occ_mat[c(4,6,3,2,1,5),]

# Define breaks from 1 to 100
my_breaks <- seq(0, 57, length.out = 40)

mat_perc <- occ_mat.sorted * 100

# --- categories & colors ---
cat_levels <- c("airborne", "animal", "plant", "freshwater", "saline", "soil")
group_cols <- c(
  airborne   = "black",
  animal     = "#e67e22",   # orange
  plant      = "#2ca02c",   # green
  freshwater = "#74a9cf",   # light blue
  saline     = "#045a8d",   # dark blue
  soil       = "#8c510a"    # brown
)

# group assignment
```

```
biome_full  <- colnames(mat_perc)
biome_group <- sub("-.*$", "", biome_full)
biome_group <- dplyr::recode(biome_group, plants = "plant")   # unify naming
cat_fac     <- factor(biome_group, levels = cat_levels)

# shorten labels (after "-")
biome_short <- ifelse(grepl("-", biome_full),
                      sub("^[^-]+-", "", biome_full),
                      biome_full)

# block labels: capitalize, omit airborne
block_labels <- ifelse(cat_levels == "airborne", "", str_to_title(cat_levels))

# bottom annotation with group-colored boxes and white labels
ha_bottom <- HeatmapAnnotation(
  Category = anno_block(
    gp        = gpar(fill = group_cols[cat_levels], col = NA),
    labels    = block_labels,
    labels_gp = gpar(col = "white", fontsize = 10),
    which     = "column"
  )
)

col_fun <- circlize::colorRamp2(
  c(0, 0.001, 10, 20, 40, 60),
  c("grey85", "white", "#FFF7BC", "gold", "orange", "darkred"))
```

```
ht <- Heatmap(
  mat_perc,
  name = "%Occupancy",
  col  = col_fun,
  cluster_rows    = FALSE,
  cluster_columns = FALSE,
  column_split      = cat_fac,
  column_labels     = biome_short,
  column_names_side = "bottom",
  column_names_rot  = 45,
  column_names_gp   = gpar(col = group_cols[as.character(cat_fac)], fontsize = 9),
  row_names_side    = "left",
  row_names_gp      = gpar(fontsize = 8, fontface = "italic"),
  border  = TRUE,
  rect_gp = gpar(col = "grey40", lwd = 0.75),
  bottom_annotation = ha_bottom,
  column_title = NULL,
  show_heatmap_legend = FALSE,                # <-- important
  heatmap_legend_param = list(direction="horizontal", fontsize=9)
)

# 2) Your custom legend
lgd <- Legend(title = "%Occupancy", col_fun = col_fun, direction = "horizontal")

# 3) Capture EVERYTHING in a single grob; don't draw beforehand
combined_grob <- grid.grabExpr({
  draw(ht, newpage = FALSE)                    # draw once, inside grabExpr
  pushViewport(viewport(x = unit(-55, "mm"),   # adjust as needed
                        y = unit(-10, "mm"),
                        just = c("left", "bottom")))
  draw(lgd)
  upViewport()
})

ggdraw(combined_grob)
```



2.4.2 Figure 4



2.4.3 Screen for the pathogens and SKAPE bacteria (example, Table S7)

```
#Identification of the WHO bacterial priority pathogens list 2024
#1) Acinetobacter baumannii
# 90_8;96_11;97_11;98_13;99_14

#2) Pseudomonas aeruginosa
#"90_23;96_36;97_37;98_40;99_42"

#3)Neisseria gonorrhoeae subclade
# 90_15355;96_42;97_44;98_48;99_52

#4) Streptococcus pneumoniae
```

```
#90_4;96_4;97_4;98_4;99_4

#5) Haemophilus influenzae
# 90_18;96_30;97_31;98_34;99_36

#6) Shigella spp (KF080510)
# 90_11728;96_51054;97_65764;98_87812;99_131241

#7) Enterococcus faecium
# 90_13;96_19;97_19;98_22;99_23

#8) Staphylococcus aurerus
# 90_3;96_3;97_3;98_3;99_3

#9) Klebsiella pneumoniae
# 90_6;96_7;97_7;98_9;99_10


#Calculate percentiles per biome

biomes <- df_MAP_Elo.corr %>% distinct(biome)
stats <- df_MAP_Elo.corr %>%
  group_by(biome) %>%
  mutate(biome_percentile = percent_rank(mean_rating) * 100) %>%
  ungroup()

#Query
id <- "90_6;96_7;97_7;98_9;99_10" #Klebsiella pneumoniae

#Calculate players stats
player <- stats %>%
  filter(player_id == id) %>%
  group_by(biome) %>%
  summarise(player_mean_rating = round(mean(mean_rating), 2),
            biome_percentile = round(mean(biome_percentile), 2),
            .groups = "drop")
#summary
out <- biomes %>% left_join(player, by = "biome") %>% arrange(desc(biome_percentile))
t(out)
```

```
##                     [,1]       [,2]       [,3]      [,4]      [,5]
## biome               "animal.3" "animal.2" "plant.2" "plant.3" "animal.6"
## player_mean_rating "817.27"   "758.08"   "707.19"  "694.09"  "630.64"
## biome_percentile   "99.81"    "99.38"    "99.19"   "97.92"   "97.62"
##                     [,6]      [,7]           [,8]       [,9]
## biome               "plant.4" "freshwater.3" "animal.5" "animal.1"
## player_mean_rating "719.24"  "795.89"       "775.31"   "384.49"
## biome_percentile   "97.45"   "97.36"        "96.97"    "96.84"
##                     [,10]          [,11]     [,12]          [,13]      [,14]
## biome               "freshwater.2" "plant.1" "freshwater.1" "animal.4" "soil.1"
## player_mean_rating "812.43"       "903.40"  "870.04"       "793.29"   "902.94"
## biome_percentile   "96.70"        "96.63"   "95.59"        "94.47"    "93.79"
##                     [,15]    [,16]    [,17]      [,18]      [,19]      [,20]
## biome              "soil.5" "soil.4" "saline.1" "saline.2" "saline.3" "soil.2"
## player_mean_rating "924.21" "832.25" "868.75"   "846.52"   "843.90"   "862.35"
## biome_percentile   "90.02"  "88.93"  "88.26"    "82.98"    "79.52"    "61.99"
##                    [,21]        [,22]        [,23]        [,24]
## biome              "peatland.1" "airborne.1" "peatland.2" "soil.3"
## player_mean_rating "849.70"     "865.49"     "870.57"     NA
## biome_percentile   "51.77"      "51.59"      " 9.32"      NA
```

3 Network analysis

The biome connectivity network was constructed from the presence/absence of the top 5% BB-scores OTUs within each biome using the Jaccard dissimilarity. To assess whether similarity was higher than by change, we used the maximum likelihood estimation from the CooccurrenceAffinity R-package.

3.1 Prepare matrix presence-absences

3.2 Contruct Netowrk based on Jaccard dissimilarity.

The biome connectivity network used the presence/absence of the top 5% BB-scores OTUs within each biome using the Jaccard dissimilarity. To determine whether similarity was higher than by chance, we used the maximum likelihood estimation (alpha) from the CooccurrenceAffinity R-package.

```
# 2) Compute pairwise Jaccard distances
res <- affinity(data = presence_absence, row.or.col = "row",datatype = "binary",squarematrix = c("jacca

res.affinity<-res$all
res.affinity$p_value<-as.numeric(res.affinity$p_value)
res.affinity$padjust<-p.adjust(res.affinity$p_value,method="bonferroni")

#Save data for network visualization in Gephi
write.csv(file="../data/network.table.top5perc.csv",res.affinity,row.names = FALSE)
```

3.3 Network properties

Network analsysis and small-worldness index (SWI) were computed, and its significance was assessed by a Monte Carlo randomization test (Manly,2018. Randomization, bootstrap and Monte Carlo methods in biology) against 1000 degree-preserving null networks.

```
# Filter out a>0 and p-adjust <0.05 as in Gephi network
edges <- res.affinity %>%
  filter(alpha_mle > 0, padjust < 0.05) %>%
  dplyr::select(entity_1, entity_2, jaccard)

#Prepare adjance matrix from dataframe
nodes <- sort(unique(c(edges$entity_1, edges$entity_2)))
A <- matrix(0, length(nodes), length(nodes), dimnames = list(nodes, nodes))

for(i in 1:nrow(edges)){
  a <- edges$entity_1[i]; b <- edges$entity_2[i]
  A[a, b] <- A[b, a] <- 1
```

```r
}

# Network parameters
set.seed(1)
smallworldness(A, B = 1000)
```

```
##       smallworldness         trans_target averagelength_target
##            1.1847561            0.7179648            1.5688406
##          trans_rnd_M         trans_rnd_lo         trans_rnd_up
##            0.5897619            0.5615578            0.6218593
##  averagelength_rnd_M averagelength_rnd_lo averagelength_rnd_up
##            1.5267971            1.5108696            1.5471014
```

3.4 Smallworldness index (SWI) by Monte Carlo randomization test

```r
#1) Get observed index
set.seed(1)
sw_obs <- smallworldness(A, B = 0)
sw_obs[1]
```

```
## smallworldness
##       1.193143
```

```r
#2) Extract network from adjancent matrix
g_obs <- graph_from_adjacency_matrix(A, mode = "undirected", diag = FALSE)

#3) Estimate null distribution
set.seed(1)
B <- 1000
sw_null <- numeric(B)

for (b in seq_len(B)) {
  g_rand <- igraph::sample_degseq(igraph::degree(g_obs), method = "vl")   # preserves degree sequence
  tmp_sw<- smallworldness(g_rand, B = 0)
  sw_null[b] <-tmp_sw[["smallworldness"]]# Save the result from each iteration
}

#4) Estimate P-value by comparing the observed SWI vs the SWI-NULL distribution data
B <- length(sw_null)
p_one_sided <- (sum(sw_null >= sw_obs[1]) + 1) / (B + 1)

print (paste("Pvalue Monte Carlo test =",round(p_one_sided,3)))
```

```
## [1] "Pvalue Monte Carlo test = 0.001"
```