Figure4A

Emiliano Pereira

2025-08-06

Intro

Taxa differentially associated between semesters based on the IndVal analysis to reproduce the results of Figure 4A from the original publication Seasonal dynamics of the coastal microbiome and its association with environmental factors.

1. Set the environment

```
library(tidyverse)
library(vegan)
library(doParallel)
library(indicspecies)
source("../scripts/resources/rare_indval.R")
```

2. Load data

`asvs_non_rare_long.tsv` contains the non-rarefied ASV abundance profiles, in a long format. `date2season2community.tsv` is table mapping the date, season, and community columns. `asv_table_nbcandem_annot_clean` contains taxonomic annotation for each ASV as obtained by NBC.

```
ABUND <- read_tsv("../data/asvs_workable.tsv.gz", show_col_types = FALSE) %>%
         column_to_rownames("Date")
ASV_ABUND_TBL <- read_tsv("../data/asvs_non_rare_long.tsv", show_col_types = FALSE)
DATE2SEASON2COMMUITY <- read_tsv("../data/date2season2community.tsv", show_col_types = FALSE)
ASV_TAXA_TBL <- read_tsv("../data/asv_table_nbcandem_annot_clean_long.tsv.gz", col_names = T, show_col_
colnames(ASV_TAXA_TBL)[1] <- "asv_id"
```

3. Format abundance table ASV_ABUND_TBL

Remove all singletons and samples with less than 2500 reads.

```
ASV_ABUND_TBL_filt <- ASV_ABUND_TBL %>%
                      group_by(Date) %>%
                      mutate(sample_sum = sum(abund)) %>%
                      filter(asv_n >= 2 & sample_sum >= 2500)
```

4. Convert table to wide

```
ASV_ABUND_TBL_filt_wide <- ASV_ABUND_TBL_filt %>%
                           dplyr::select(Date, asv_id, abund) %>%
                           pivot_wider(names_from = asv_id,
                                       values_from = abund,
                                       values_fill = 0) %>%
                           arrange(Date) %>%
                           column_to_rownames(var = "Date")
```

**5. Define variables to run IndVal over iterated rarefactions

```
# Set the number of cores to use for parallel processing;
# this should be adjusted to the local computational capacitites.
n_cores <- 44
doParallel::registerDoParallel(n_cores)

sample_size <- rowSums(ASV_ABUND_TBL_filt_wide) %>% min()

semester <- DATE2SEASON2COMMUITY %>%
  dplyr::select(Date, Community) %>%
  column_to_rownames("Date")

semester_ord <- semester[rownames(ASV_ABUND_TBL_filt_wide),, drop = F]
# Sanity check
all(rownames(semester_ord) == rownames(ASV_ABUND_TBL_filt_wide))
```

```
## [1] TRUE
```

```
vector_group <- semester_ord$Community
```

6. Run IndVal over iterated rarefactions

To accelerate the execution of the following function, we will use the paramters nperm = 99. However, to reproduce the results from the original publication, this parameter should be set to 9999.

```
rare_indval_output <- rare_indval(iterations = 100,
                                  abundance_table = ASV_ABUND_TBL_filt_wide,
                                  sample_size = sample_size,
                                  vector_group = vector_group,
                                  nperm = 99,
                                  p_value = 1e-2)
```

```
## `summarise()` has grouped output by 'asv_id'. You can override using the `.groups` argument.
```

7. Select ASVs with a consistency score of 100

```
rare_indval_output_s1_100 <- rare_indval_output %>%
                             filter(consistency >= 100 & semester == "S1")

rare_indval_output_s2_100 <- rare_indval_output %>%
                              filter(consistency >= 100 & semester == "S2")

# Check dimensions
dim(rare_indval_output_s1_100)
```

```
## [1] 11  5
```

```
dim(rare_indval_output_s2_100)
```

```
## [1] 3 5
```

8. Create ASV to taxonomy table

```
asv2tax_filt <- ASV_TAXA_TBL %>%
                dplyr::select(asv_id,
                              starts_with("tax"),
                              starts_with("boot")) %>%
                mutate(asv_id = as.character(asv_id)) %>%
                mutate(tax.Genus = if_else(boot.Genus > 75, tax.Genus, NA)) %>%
                mutate(tax.Family = if_else(boot.Family > 75, tax.Family, NA)) %>%
                mutate(tax.Order = if_else(boot.Order > 75, tax.Order, NA)) %>%
                mutate(tax.Class = if_else(boot.Class > 75, tax.Class, NA)) %>%
                mutate(tax.Phylum = if_else(boot.Phylum > 75, tax.Phylum, NA)) %>%
                unique() %>%
                select(-starts_with("boot")) %>%
                mutate(tax.Family = if_else(tax.Order == "SAR11_clade", paste("SAR11_", tax.Family, sep
                mutate(tax.Genus = if_else(tax.Order == "SAR11_clade", paste("SAR11_", tax.Genus, sep =
```

9. Map taxonomy

```
indval_output_df_sig_s1_100_wtax <- left_join(x = rare_indval_output_s1_100,
                                              y = asv2tax_filt,
                                              by = "asv_id")

indval_output_df_sig_s2_100_wtax <- left_join(x = rare_indval_output_s2_100,
                                              y = asv2tax_filt,
                                              by = "asv_id")
```

10. Map abundance

```
samples_s1 <- semester_ord %>% filter(Community == "S1") %>% rownames()
samples_s2 <- semester_ord %>% filter(Community == "S2") %>% rownames()
ABUND_s1 <- ABUND[samples_s1,]
ABUND_s2 <- ABUND[samples_s2,]

ABUND_s1 <- ABUND_s1[, colSums(ABUND_s1) > 0]
ABUND_s2 <- ABUND_s2[, colSums(ABUND_s2) > 0]

# compute mean feature relative abundance in ABUND_s1 and ABUND_s2
mean_abund_s1 <- mean(colSums(ABUND_s1)/sum(ABUND_s1))
mean_abund_s2 <- mean(colSums(ABUND_s2)/sum(ABUND_s2))

asvs_abund_s1 <- data.frame(asv_abund_rel = colSums(ABUND_s1)/sum(colSums(ABUND_s1)),
                         asv_id = colnames(ABUND_s1))
asvs_abund_s2 <- data.frame(asv_abund_rel = colSums(ABUND_s2)/sum(colSums(ABUND_s2)),
                         asv_id = colnames(ABUND_s2))

indval_output_df_sig_s1_100_wtax_wabund <- left_join(x = indval_output_df_sig_s1_100_wtax,
                                                     y = asvs_abund_s1,
                                                     by = "asv_id")
indval_output_df_sig_s2_100_wtax_wabund <- left_join(x = indval_output_df_sig_s2_100_wtax,
                                                     y = asvs_abund_s2,
                                                     by = "asv_id")
```

11. Count number of ASVs per family and phylum, and compute mean relative abundance

```
indval_s1_counts2abund <- indval_output_df_sig_s1_100_wtax_wabund %>%
                          group_by(tax.Family, tax.Phylum) %>%
                          summarize(n_asv = length(unique(asv_id)),
                                    mean_abund = mean(asv_abund_rel),
                                    sd_abund = sd(asv_abund_rel),
                                    abund_total = sum(asv_abund_rel),
                                    mean_stat = mean(stat_mean)) %>%
                         # mutate(abund_total = if_else(is.na(tax.Family) == T, NA, abund_total)) %>%
                          mutate(mean_stat = if_else(is.na(tax.Family) == T, NA, mean_stat))  %>%
                          mutate(tax.Family = if_else(is.na(tax.Family), "Unclassified", tax.Family))
```

```
## `summarise()` has grouped output by 'tax.Family'. You can override using the `.groups` argument.
```

```
indval_s2_counts2abund <- indval_output_df_sig_s2_100_wtax_wabund %>%
                          group_by(tax.Family, tax.Phylum) %>%
                          summarize(n_asv = length(unique(asv_id)),
                                    mean_abund = mean(asv_abund_rel),
                                    sd_abund = sd(asv_abund_rel),
                                    abund_total = sum(asv_abund_rel),
                                    mean_stat = mean(stat_mean)) %>%
                          # mutate(abund_total = if_else(is.na(tax.Family) == T, NA, abund_total))  %>%
                          mutate(mean_stat = if_else(is.na(tax.Family) == T, NA, mean_stat)) %>%
                          mutate(tax.Family = if_else(is.na(tax.Family), "Unclassified", tax.Family))
```

```
## `summarise()` has grouped output by 'tax.Family'. You can override using the `.groups` argument.
```

12. Define colors and levels of tax.Family

```
phyla_palette <- c(
  "#8B3E2F",  # earthy red (burnt sienna)
  "#C97E2A",  # clay orange (ochre)
  "#A9745A",  # canyon brown (raw umber)
  "#4A6C59",  # moss green (pine)
  "#5F9EA0", # Cadet Blue
  "#6A8BAA",   # muted blue (mineral water)
  "#3E5F9E",  # slate blue (dusty indigo)
  "#6D597A",  # dusty purple (dried lavender)
  "#483D8B" # Dark Slate Blue

)

names(phyla_palette) <- c(indval_s1_counts2abund$tax.Phylum, indval_s2_counts2abund$tax.Phylum) %>% uni
indval_s1_counts2abund$color <- phyla_palette[indval_s1_counts2abund$tax.Phylum]
indval_s2_counts2abund$color <- phyla_palette[indval_s2_counts2abund$tax.Phylum]

families <- rbind(indval_s1_counts2abund,
                  indval_s2_counts2abund) %>%
            filter(tax.Family != "Unclassified") %>%
            pull(tax.Family) %>% unique()
families_levels <- c(families[order(families)], "Unclassified")

indval_s1_counts2abund$tax.Family <- factor(indval_s1_counts2abund$tax.Family,
                                            levels = families_levels)
```

```r
indval_s2_counts2abund$tax.Family <- factor(indval_s2_counts2abund$tax.Family,
                                            levels = families_levels)
```

13. Create barplots: S1

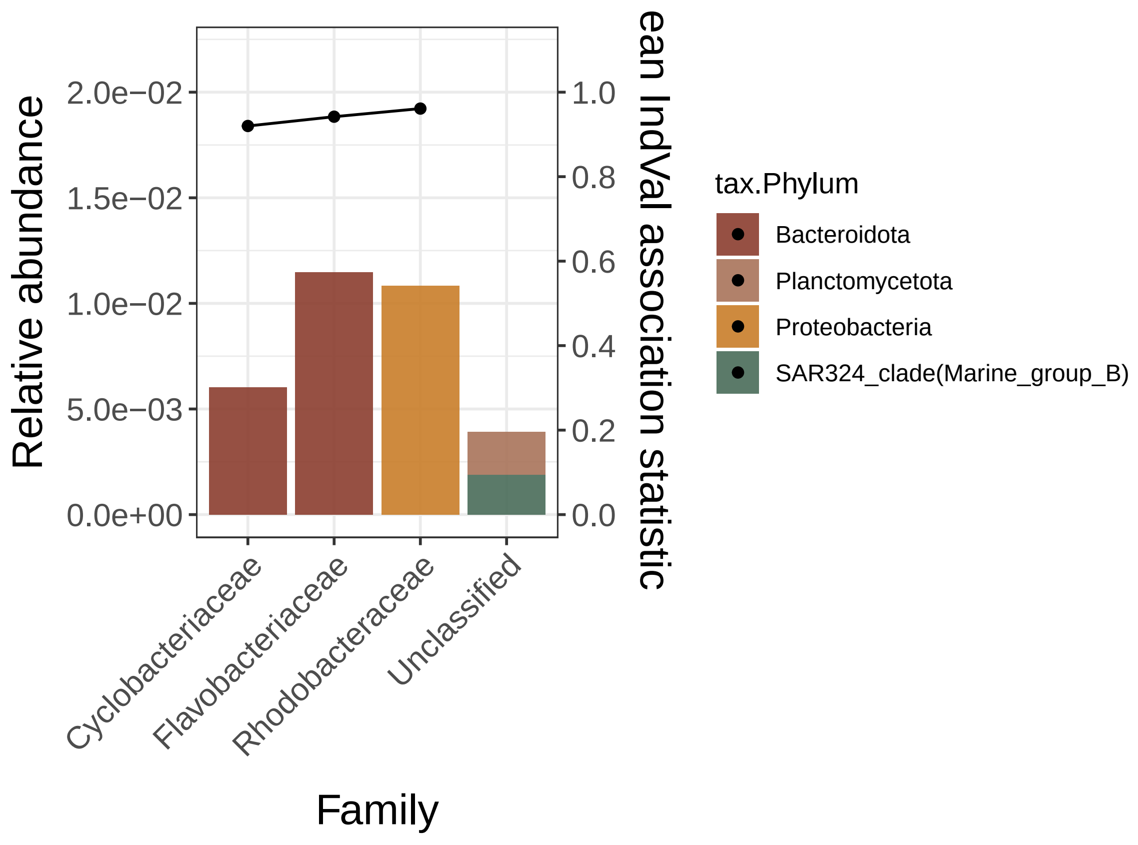
```r
scaling_factor_s1 <- 2e-2
text_size <- 14

barplot_asv_indval_s1 <- indval_s1_counts2abund %>%
                         ggplot(.,
                                aes(x = tax.Family,
                                    y = abund_total,
                                    fill = tax.Phylum)) +
                          geom_bar(stat = "identity", alpha = 0.9) +
                          scale_fill_manual(values = indval_s1_counts2abund$color) +
                          geom_line(aes(x = tax.Family,
                                        y = mean_stat*scaling_factor_s1,
                                        group = 1), color = "black") +
                          geom_point(aes(x = tax.Family,
                                         y = mean_stat*scaling_factor_s1,
                                         group = 1), color = "black") +
                          scale_y_continuous(
                                        limits  = c(0, scaling_factor_s1 + 2e-3),
                                        labels = function(x) format(x, scientific = TRUE),
                                        name = "Relative abundance",
                                        sec.axis = sec_axis(~ . / scaling_factor_s1,
                                                            name = "Mean IndVal association statistic",
                                                            breaks = seq(0, 1, 0.2))
                          ) +
                          # geom_errorbar(aes(ymin = (mean_abund*scaling_factor_s1 - sd_abund),
                          #                   ymax = (mean_abund*scaling_factor_s1 + sd_abund)),
                          #               linewidth = 0.3,  width = 0.2, color = "gray5") +
                          xlab("Family") +
                          theme_bw() +
                          theme(
                            axis.text.x = element_text(size = text_size-2, angle = 45, hjust = 1),
                            axis.text.y = element_text(size = text_size-2),
                            axis.title.x = element_text(size = text_size +2, margin = unit(c(4,0,0,0),
                            axis.title.y = element_text(size = text_size +2, margin = unit(c(0,2,0,0),
                            axis.title.y.right = element_text(size = text_size +2, margin = unit(c(0,0,
                            plot.title = element_text(size = text_size+4, hjust = 0.5),
                            strip.text = element_text(size = text_size+4),
                            strip.background = element_blank(),
                            plot.margin = margin(r = 1, l = 2, t= 0.5,b = 1, unit = "lines")
                          ) #  +
                           # theme(legend.position = "none")


barplot_asv_indval_s1
```

```
## Warning: Removed 2 rows containing missing values or values outside the scale range (`geom_line()`).

## Warning: Removed 2 rows containing missing values or values outside the scale range (`geom_point()`)
```

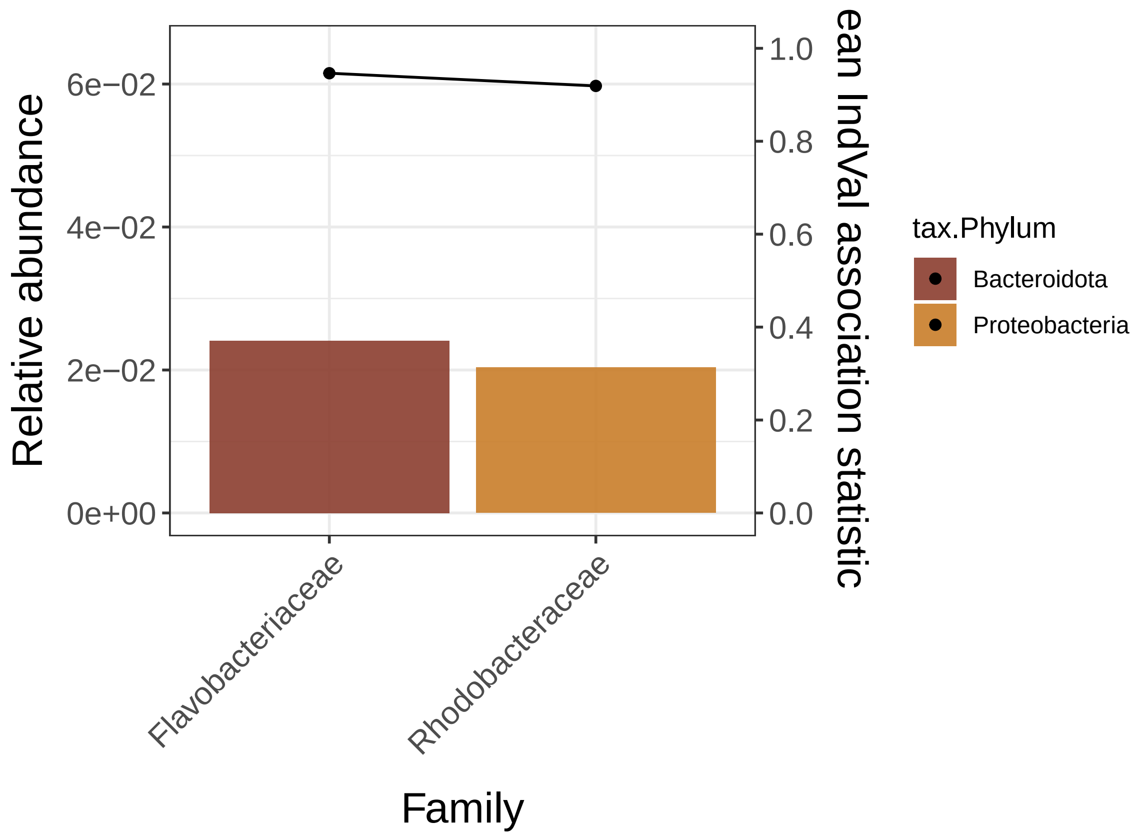


### '14. Create barplots: S2

```
scaling_factor_s2 <- 6.5e-2
text_size <- 14

barplot_asv_indval_s2 <- indval_s2_counts2abund %>%
                          ggplot(.,
                                 aes(x = tax.Family,
                                     y = abund_total,
                                     fill = tax.Phylum)) +
                          geom_bar(stat = "identity", alpha = 0.9) +
                          scale_fill_manual(values = indval_s2_counts2abund$color) +
                          geom_line(aes(x = tax.Family,
                                        y = mean_stat*scaling_factor_s2,
                                        group = 1), color = "black") +
                          geom_point(aes(x = tax.Family,
                                         y = mean_stat*scaling_factor_s2,
                                         group = 1), color = "black") +
                          scale_y_continuous(
                                      limits  = c(0, scaling_factor_s2),
                                      labels = function(x) format(x, scientific = TRUE),
                                      name = "Relative abundance",
                                      sec.axis = sec_axis(~ . / scaling_factor_s2,
                                                          name = "Mean IndVal association statisti
                                                          breaks = seq(0, 1, 0.2))
                          ) +
                          # geom_errorbar(aes(ymin = (mean_abund*scaling_factor - sd_abund),
                          #                   ymax = (mean_abund*scaling_factor + sd_abund)),
                          #               linewidth = 0.3,  width = 0.2, color = "gray5") +
                          xlab("Family") +
                          theme_bw() +
                          theme(
                            axis.text.x = element_text(size = text_size-2, angle = 45, hjust = 1),
                            axis.text.y = element_text(size = text_size-2),
                            axis.title.x = element_text(size = text_size +2, margin = unit(c(4,0,0,0),
                            axis.title.y = element_text(size = text_size +2, margin = unit(c(0,2,0,0),
                            axis.title.y.right = element_text(size = text_size +2, margin = unit(c(0,0,
                            plot.title = element_text(size = text_size+4, hjust = 0.5),
                            strip.text = element_text(size = text_size+4),
                            strip.background = element_blank(),
                            plot.margin = margin(r = 1, l = 2, t= 0.5,b = 1, unit = "lines")
                          )
                          # theme(legend.position = "none")


barplot_asv_indval_s2
```



15. Print session info

```
sessionInfo()
```

```
## R version 4.4.2 (2024-10-31)
## Platform: x86_64-pc-linux-gnu
## Running under: Ubuntu 20.04.6 LTS
##
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/blas/libblas.so.3.9.0
## LAPACK: /usr/lib/x86_64-linux-gnu/lapack/liblapack.so.3.9.0
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C               LC_TIME=en_US.UTF-8        LC_COLLATE=en_U
##  [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8    LC_PAPER=en_US.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C             LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATI
##
## time zone: Etc/UTC
## tzcode source: system (glibc)
##
## attached base packages:
## [1] parallel  stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] indicspecies_1.8.0 doParallel_1.0.17  iterators_1.0.14   foreach_1.5.2      vegan_2.6-8        l
##  [7] permute_0.9-7      lubridate_1.9.3    forcats_1.0.0      stringr_1.5.1      dplyr_1.1.4        p
## [13] readr_2.1.5        tidyr_1.3.1        tibble_3.2.1       ggplot2_3.5.1      tidyverse_2.0.0
##
## loaded via a namespace (and not attached):
##  [1] utf8_1.2.4        generics_0.1.3    stringi_1.8.4     digest_0.6.37     hms_1.1.3         magrit
##  [7] evaluate_1.0.1    grid_4.4.2        timechange_0.3.0  fastmap_1.2.0     Matrix_1.7-0      mgcv_
## [13] fansi_1.0.6       scales_1.3.0      codetools_0.2-20  cli_3.6.3         crayon_1.5.3      rlang
## [19] cowplot_1.1.3     bit64_4.5.2       munsell_0.5.1     splines_4.4.2     yaml_2.3.10       withr
## [25] tools_4.4.2       tzdb_0.4.0        colorspace_2.1-1  vctrs_0.6.5       R6_2.5.1          lifec
## [31] bit_4.5.0         vroom_1.6.5       MASS_7.3-61       cluster_2.1.6     pkgconfig_2.0.3   pilla
## [37] gtable_0.3.5      glue_1.8.0        highr_0.11        xfun_0.48         tidyselect_1.2.1  rstud
## [43] knitr_1.48        farver_2.1.2      htmltools_0.5.8.1 nlme_3.1-166      labeling_0.4.3    rmark
## [49] compiler_4.4.2
```