Script04: Community Patterns

Sara Beier

2024-11-06

Contents

1 Clean and setup working space

```
rm(list = ls())
loaded_packages <- setdiff(loadedNamespaces(), c("base", "compiler", "datasets",
    "graphics", "grDevices", "grid", "methods", "parallel", "splines", "stats",
    "stats4", "tcltk", "tools", "utils"))
for (pkg in loaded_packages) {
    try(detach(paste0("package:", pkg), unload = TRUE, character.only = TRUE),
        silent = TRUE)
}

# Load packages
library(tidyverse)
library(phyloseq)
library(PCAtest)
library(pairwiseAdonis)
library(reshape2)

# Define functions read files with PICRUSt predictions
read.data.pic <- function(i) {
    read.table(paste0(path.pic, files.pic[i]), header = T, sep = "\t", row.names = 1)[,
        1, drop = F] %>%
        filter(grepl("SV", rownames(.)))
}
## Community weighted means (CWMs)
f.CWM.pic <- function(i) {
    counts.s.rel.pic %>%
        rownames_to_column(var = "ASV") %>%
        inner_join(pic[, c(1, i)], by = c(ASV = "ASV")) %>%
        mutate_at(c(2:(dim(counts.s)[2] + 1)), .funs = funs(. * dplyr::select(cur_data_all(),
            dim(counts.s)[2] + 2))) %>%
        dplyr::select(-1, -(dim(counts.s)[2] + 2)) %>%
        summarise(across(everything(), sum))
}
```

2 Load community input data

(output file from script02)

```
load("/Users/sara/Documents/DFG/coalescense/dada-coal.img")

## Split initial and experimental samples
ps.rel_i <- prune_samples(sample_names(ps.rel)[c(1:3, 17:19)], ps.rel)  # initial samples
ps.rel_e <- prune_samples(sample_names(ps.rel)[c(4:16, 20:22)], ps.rel)  # experimental samples
counts_i <- data.frame(t(otu_table(ps.rel_i)))
counts_e <- data.frame(t(otu_table(ps.rel_e)))
schema_i <- data.frame(sample_data(ps.rel_i))
schema_e <- data.frame(sample_data(ps.rel_e))
```

3 Load and format input data for CWM calculations of PICRUSt predicted genomic traits (output script03_genomic_traits_estimation)

```
## Information about data transformations
pic.traits <- read.table("../Data/PICRUSt_predictions/trait_info_coal.tsv",
  header=T, sep ='\t')
pic.traits
```

```
##          var      var.min      var.max var.trans
## 1 genomesize 7.181850e+05 1.604067e+07      <NA>
## 2    TF_perc 9.794319e-02 7.102273e+00      <NA>
## 3    cub.dRg 1.401977e-01 4.668551e+03    log(x)
```

```
## Filenames of PICRUSt predictions (excluding RRN)
path.pic<-("../Data/PICRUSt_predictions/")
files.pic <-sort(list.files(path.pic, pattern=glob2rx("pic*_predicted")))
files.pic <- files.pic[!files.pic %in% c("pic.16S_predicted_rrnDB")] #remove pic.RRN_predicted.tsv
files.pic
```

```
## [1] "pic.d.gRodon_predicted"   "pic.genomesize_predicted"
## [3] "pic.TF_perc_predicted"
```

```
## NSTI values
NSTI <-   read.table(list.files(path.pic, pattern=glob2rx("pic*_predicted"),
                                full.names = TRUE)[1],
                     header=T, sep='\t', row.names = 1)[,2, drop=F] %>%
  filter(grepl("SV", rownames(.)))

## Read RRN predictions based on rrnDB
pic.16s <- read.table("../Data/PICRUSt_predictions/pic.16S_predicted_rrnDB",
  header=T)

## Concatenate files and retransform trait predictions
pic <-do.call(cbind,lapply(seq(1, length(files.pic)), FUN=read.data.pic)) %>%
  rownames_to_column(var="ASV") %>%
  tibble() %>%
  mutate (metadata_NSTI.pic = NSTI$metadata_NSTI) %>%
  relocate (metadata_NSTI.pic, .before = d.gRodon) %>%
  left_join(pic.16s, by=c("ASV"="sequence")) %>%
  dplyr::rename ("RRN"="X16S_rRNA_Count") %>%
  dplyr::rename("metadata_NSTI.rrn"= "metadata_NSTI") %>%
  relocate(metadata_NSTI.rrn, .before = d.gRodon) %>%

  mutate (d.gRodon = exp(1)^ (d.gRodon / 1000 * (log(pic.traits$var.max[3])
                                                 -log(pic.traits$var.min[3]))
                              + log(pic.traits$var.min[3]))) %>%
  mutate (umax = 1/d.gRodon) %>% #maximal growth rates from d.gRodon
  relocate (umax, .after = d.gRodon) %>%
  mutate (genome.size = ((genome.size / 1000 * (pic.traits$var.max[1]
                                                -pic.traits$var.min[1]))
                         + pic.traits$var.min[1])/1000000) %>% #divide for results in Mbp
  mutate (TF_perc =  (TF_perc / 1000 * (pic.traits$var.max[2]
                                        -pic.traits$var.min[2]))
          + pic.traits$var.min[2] )
pic
```

```
## # A tibble: 688 x 8
##    ASV   metadata_NSTI.pic metadata_NSTI.rrn d.gRodon   umax genome.size TF_perc
##    <chr>             <dbl>             <dbl>    <dbl>  <dbl>       <dbl>   <dbl>
##  1 SV_1~           0.286               0.653    3.19  0.314         3.29    1.68
##  2 SV_1~           0.0685              0.190   11.2   0.0890        2.45    1.56
##  3 SV_1~           0.206               0.191    4.45  0.225         4.01    3.17
##  4 SV_1~           0.0326              0.589    4.94  0.203         4.66    2.79
##  5 SV_1~           0.00784             0.196    3.25  0.307         3.80    2.02
##  6 SV_1~           0.0329              0.162    3.39  0.295         3.37    1.97
##  7 SV_1~           0.0213              0.221    3.46  0.289         3.09    2.75
##  8 SV_1~           0.188               0.234    2.99  0.334         4.38    2.53
##  9 SV_1~           0.0331              0.264    0.914 1.09          1.42    2.16
## 10 SV_1~           0.0298              0.690   10.3   0.0967        1.30    1.11
## # i 678 more rows
## # i 1 more variable: RRN <int>
```

4 CWMs genomic traits

```
## Check fraction of sequences that can be assigned to species NSTI cutoff
## 2 applied for RRN predictions based on rrnDB
counts.s <- counts_i[row.names(counts_i) %in% pic.16s[pic$metadata_NSTI.rrn <
    2, 1], ]
min(colSums(counts.s))  #sample with min number of retained reads
```

```
## [1] 0.9995454
```

```
counts.s.rel.rrn <- as.data.frame.matrix(prop.table(t(t(counts.s)), 2))  #re-normlize

## NSTI cutoff 1 applied for remaining predictions using PICRUSt default
## database
counts.s <- counts_i[row.names(counts_i) %in% pic[pic$metadata_NSTI.pic < 1,
    1, drop = T], ]
min(colSums(counts.s))  #sample with min number of retained reads
```

```
## [1] 0.9980302
```

```
counts.s.rel.pic <- as.data.frame.matrix(prop.table(t(t(counts.s)), 2))  #re-normlize
```

4.1 CWMs

```
## RRN predicted rrnDB
CWM.RRN <- counts.s.rel.rrn %>%
  rownames_to_column(var="ASV") %>%
  inner_join (pic[,c(1, dim(pic)[2])], by= c("ASV"="ASV")) %>%
  mutate_at(c(2:(dim(counts.s)[2]+1)),
            .funs = funs(. *   dplyr::select(cur_data_all(),dim(counts.s)[2]+2)   ) ) %>%
  dplyr::select (-1,-(dim(counts.s)[2]+2)) %>%
  summarise(across(everything(), sum))

## traits predicted via PICRUSt2 default database
CWM.pic <- as.data.frame.matrix(matrix(unlist(sapply ( seq(4,dim(pic)[2]), f.CWM.pic)),
                                       nrow=length(counts_i), byrow=FALSE )) %>%
```

```
  `colnames<-`(c(names (pic)[4:dim(pic)[2]])) %>%
  mutate(RRN=as.numeric(paste(CWM.RRN)) ) %>% # add rrnDB predictions
  mutate (sampleID = rownames(schema_i))
tibble(CWM.pic)
```

```
## # A tibble: 6 x 6
##   d.gRodon  umax genome.size TF_perc   RRN sampleID
##      <dbl> <dbl>       <dbl>   <dbl> <dbl> <chr>
## 1     3.51 0.327        3.73    2.41  2.60 CANET1
## 2     3.54 0.323        3.72    2.40  2.51 CANET2
## 3     3.56 0.326        3.74    2.41  2.58 CANET3
## 4     8.90 0.146        2.20    1.52  1.86 SOLA1
## 5     8.74 0.149        2.27    1.55  1.87 SOLA2
## 6     8.95 0.143        2.17    1.51  1.88 SOLA3
```

5 Alpha diversities initial samples

Input for PCA biplot (Fig. 3)

```
ASVrich_i <- apply(counts_i, 2, function(x) {
    length(x[x > 0])
})  # ASV richness

ASV.H_i <- vegan::diversity(counts_i, index = "shannon", MARGIN = 2)  # Shannon diversity
ASV.ev_i <- ASV.H_i/log(ASVrich_i)  # Pielou's eveness
data_frame(names(ASVrich_i), ASVrich = ASVrich_i, ASV.H = ASV.H_i, ASV.ev = ASV.ev_i)
```

```
## # A tibble: 6 x 4
##   `names(ASVrich_i)` ASVrich ASV.H ASV.ev
##   <chr>                <int> <dbl>  <dbl>
## 1 CANET1                 177  3.14  0.606
## 2 CANET2                 167  3.20  0.626
## 3 CANET3                 192  3.34  0.635
## 4 SOLA1                  295  4.22  0.742
## 5 SOLA2                  288  4.18  0.738
## 6 SOLA3                  286  4.16  0.736
```

6 Alpha diversities experimental samples

Input for ANOVA statistics (script04)

```
ASVrich_e <- apply(counts_e, 2, function(x) {
    length(x[x > 0])
})  # ASV richness
ASV.H_e <- vegan::diversity(counts_e, index = "shannon", MARGIN = 2)  # Shannon diversity
ASV.ev_e <- ASV.H_e/log(ASVrich_e)  # ASV Pielou's eveness

alphadiv_e <- data_frame(schema_e, ASVrich = ASVrich_e, ASV.H = ASV.H_e, ASV.ev = ASV.ev_e)
alphadiv_e
```

```
## # A tibble: 16 x 6
##    source DOM    treat     ASVrich ASV.H ASV.ev
##    <chr>  <chr>  <chr>       <int> <dbl>  <dbl>
##  1 C      SW-DOM C.SW-DOM      182  3.29  0.632
```

```
##  2 C      SW-DOM C.SW-DOM      199  3.26  0.616
##  3 C      SW-DOM C.SW-DOM      169  2.91  0.566
##  4 C      S-DOM  C.S-DOM       187  3.24  0.619
##  5 C      S-DOM  C.S-DOM       181  3.22  0.619
##  6 C      S-DOM  C.S-DOM       182  3.35  0.643
##  7 CS     SW-DOM CS.SW-DOM     240  3.20  0.583
##  8 CS     SW-DOM CS.SW-DOM     272  3.29  0.586
##  9 CS     S-DOM  CS.S-DOM      223  3.20  0.592
## 10 CS     S-DOM  CS.S-DOM      228  3.09  0.569
## 11 S      SW-DOM S.SW-DOM      241  2.52  0.460
## 12 S      SW-DOM S.SW-DOM      238  2.62  0.479
## 13 S      SW-DOM S.SW-DOM      228  2.62  0.483
## 14 S      S-DOM  S.S-DOM       158  1.07  0.211
## 15 S      S-DOM  S.S-DOM       221  3.22  0.596
## 16 S      S-DOM  S.S-DOM       217  2.89  0.538
```

7 Principal Component Analysis

```
## format input variables for PCA
vars <- CWM.pic %>%
    mutate(ASVrich = ASVrich_i) %>%
    mutate(ASV.H = ASV.H_i) %>%
    mutate(ASV.ev = ASV.ev_i) %>%
    select(-d.gRodon) %>%
    column_to_rownames(var = "sampleID")
colnames(vars) <- c("mumax", "Genome size", "%TF", "RRN", "richness", "Shannon",
    "eveness")

## Permutation based testing of statistical significance
PCAstats <- PCAtest(na.omit(vars), 100, 100, plot = F)
```

```
##
## Sampling bootstrap replicates... Please wait
##  1 of 100 bootstrap replicates 2 of 100 bootstrap replicates 3 of 100 bootstrap replicates 4 of 100 b
## Calculating confidence intervals of empirical statistics... Please wait
##
## Sampling random permutations... Please wait
##  1 of 100 random permutations                                                    2 of 100 random permu
## Comparing empirical statistics with their null distributions... Please wait
##
## ========================================================
## Test of PCA significance: 7 variables, 6 observations
## 100 bootstrap replicates, 100 random permutations
## ========================================================
##
## Empirical Psi = 39.3730, Max null Psi = 13.7022, Min null Psi = 2.4502, p-value = 0
## Empirical Phi = 0.9925, Max null Phi = 0.6114, Min null Phi = 0.3255, p-value = 0
##
## Empirical eigenvalue #1 = 6.955, Max null eigenvalue = 4.41079, p-value = 0
## Empirical eigenvalue #2 = 0.02881, Max null eigenvalue = 2.67399, p-value = 1
## Empirical eigenvalue #3 = 0.01511, Max null eigenvalue = 1.6802, p-value = 1
## Empirical eigenvalue #4 = 0.00094, Max null eigenvalue = 1.21642, p-value = 1
## Empirical eigenvalue #5 = 0.00015, Max null eigenvalue = 0.70403, p-value = 1
```

```
##
## PC 1 is significant and accounts for 99.4% (95%-CI:65.9-100) of the total variation
##
## Variables 1, 2, 3, 4, 5, 6, and 7 have significant loadings on PC 1
```

```
## Settings for PCA biplot
pca <- prcomp(na.omit(vars), center = TRUE, scale = TRUE)
prop.1 <- summary(pca)$importance[2] * 100
prop.2 <- summary(pca)$importance[5] * 100
prop.1
```

```
## [1] 99.357
```

```
prop.2
```

```
## [1] 0.412
```

```
pca.scores <- pca$x[, 1:2]
pca.loadings <- pca$rotation[, 1:2]

pca.scores[, 1] <- pca.scores[, 1] * (1/sqrt(sum(pca.scores[, 1]^2)))
pca.scores[, 2] <- pca.scores[, 2] * (1/sqrt(sum(pca.scores[, 2]^2)))
sc <- 0.1
unsigned.range <- function(x) c(-abs(min(x, na.rm = TRUE)), abs(max(x, na.rm = TRUE)))
x.scores = unsigned.range(pca.scores[, 1])
y.scores = unsigned.range(pca.scores[, 2])
x.loadings = unsigned.range(pca.loadings[, 1])
y.loadings = unsigned.range(pca.loadings[, 2])
xlim <- ylim <- x.scores <- x.loadings <- range(x.scores, x.loadings)
ratio <- max(y.scores/x.scores, y.loadings/x.loadings)/sc
```

7.1 Create PCA biplot

8 PERMANOVA

```
## prepare input data PERMANOVA
STR <- data.frame(sample_data(ps.rel_e))
bray.p <- phyloseq::distance(ps.rel_e, method = "bray")

## Test for homogeneity of multivariate dispersions
dispersion <- betadisper(bray.p, STR$treat)
anova(dispersion)  #>> p>0.05: homogeneity of variances can be assumed
```

```
## Analysis of Variance Table
##
## Response: Distances
##           Df  Sum Sq  Mean Sq F value Pr(>F)
## Groups     5 0.17112 0.034223  0.9664 0.4821
## Residuals 10 0.35414 0.035414
```

```
permutest(dispersion)  #>> p>0.05: homogeneity of variances can be assumed
```

```
##
## Permutation test for homogeneity of multivariate dispersions
## Permutation: free
## Number of permutations: 999
```

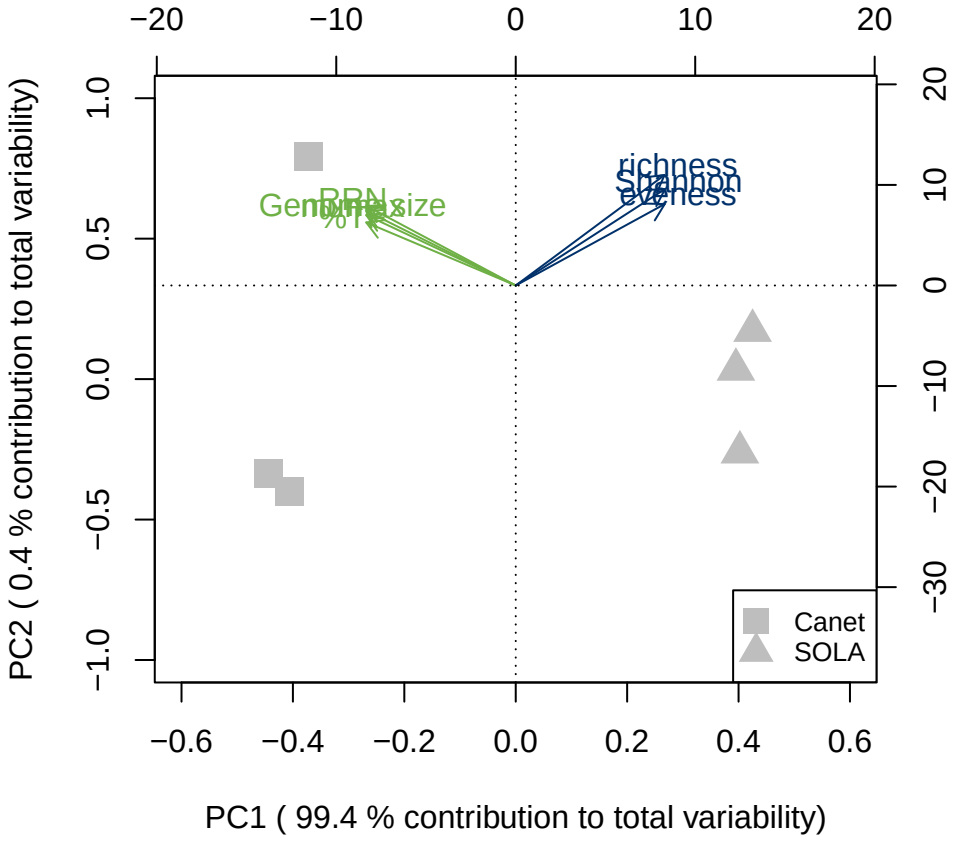

Figure 1: Principal component analysis (PCA) biplot illustrating variations of four genomic traits (community weighted means, green) and three alpha diversity measures (blue) in technical triplicates of the initial communities obtained from the Canet Lagoon and the SOLA field station.

```
##
## Response: Distances
##           Df  Sum Sq  Mean Sq      F N.Perm Pr(>F)
## Groups     5 0.17112 0.034223 0.9664    999  0.474
## Residuals 10 0.35414 0.035414
```

```
## run PERMANOVA
adonis2(bray.p ~ DOM * source, STR)  # permanova
```

```
## Permutation test for adonis under reduced model
## Terms added sequentially (first to last)
## Permutation: free
## Number of permutations: 999
##
## adonis2(formula = bray.p ~ DOM * source, data = STR)
##            Df SumOfSqs      R2       F Pr(>F)
## DOM         1   0.1449 0.03608  1.5014  0.140
## source      2   2.6711 0.66507 13.8376  0.001 ***
## DOM:source  2   0.2351 0.05854  1.2179  0.297
## Residual   10   0.9652 0.24031
## Total      15   4.0163 1.00000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
pairwise.adonis2(bray.p ~ source, STR)
```

```
## $parent_call
## [1] "bray.p ~ source , strata = Null , permutations 999"
##
## $C_vs_CS
##          Df SumOfSqs      R2      F Pr(>F)
## source    1 0.077752 0.27555 3.0429  0.013 *
## Residual  8 0.204417 0.72445
## Total     9 0.282169 1.00000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## $C_vs_S
##          Df SumOfSqs     R2      F Pr(>F)
## source    1   2.1339 0.6246 16.638  0.004 **
## Residual 10   1.2825 0.3754
## Total    11   3.4165 1.0000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## $CS_vs_S
##          Df SumOfSqs      R2      F Pr(>F)
## source    1   1.6354 0.57608 10.871  0.007 **
## Residual  8   1.2034 0.42392
## Total     9   2.8388 1.00000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## attr(,"class")
## [1] "pwadstrata" "list"
```

9 Ordinations

```
pcoa.bray <- ordinate(ps.rel, method = "PCoA", distance = "bray")

## create symbol and color vectors
col <- c(rep("gray", 3), rep("darkolivegreen3", 3), rep("#7F95AA", 3), rep("#CF5053",
    2), rep("#CF5053", 2), rep("darkolivegreen3", 3), rep("gray", 3), rep("#7F95AA",
    3))
sym <- c(rep(15, 3), rep(0, 6), rep(14, 4), rep(2, 3), rep(17, 3), rep(2, 3))
symcol <- data.frame(col, sym)
rownames(symcol) <- sample_names(ps.rel)
```

10 Community barplots

```
## Create input dataframe
df_bar <- data.frame(taxa.gt, data.frame(ASV = colnames(otu_table(ps.rel)),
    t(otu_table(ps.rel))))[, c(4, 9:30)] %>%
    drop_na() %>%
    group_by(Order) %>%
    summarise(across(everything(), sum))
order <- df_bar[, 1, drop = T]  #vector with orders
df_bar <- df_bar[, -1]  #remove column with orders and keep only abundance data
colSums(df_bar)  #test colSums to see if values close to 1 are reached
```

```
##    CANET1    CANET2    CANET3       CM1       CM2       CM3       CS1       CS2
## 0.9824987 0.9856807 0.9733313 0.9912115 0.9906053 0.9936359 0.9905296 0.9906053
##       CS3      SCM1      SCM2      SCS2      SCS3       SM1       SM2       SM3
## 0.9923479 0.9954542 0.9912872 0.9937116 0.9913630 0.9959845 0.9935601 0.9971210
##     SOLA1     SOLA2     SOLA3       SS1       SS2       SS3
## 0.8848398 0.9193878 0.8862035 0.9978029 0.9974998 0.9983332
```

```
## pool counts (mean) by treatment
agg = aggregate(t(df_bar), by = list(sample_data(ps.rel)$treat), FUN = mean) %>%
    column_to_rownames(var = "Group.1")

## change format to samples by column
agg <- t(agg)
agg <- as.data.frame(agg)
rownames(agg) <- order  #add order information
agg$Sum.agg <- rowSums(agg)  # column with counts at order-level

## select top 10 orders
agg10 <- agg[with(agg, order(-Sum.agg)), ][1:10, 1:8]
agg10$order <- rownames(agg10)

## convert to long format
agg10.long <- melt(agg10, id.vars = "order", variable.name = "treat")
agg10.long$value <- agg10.long$value * 100

## define sample order in barplot
positions <- c("S.org", "S.SW-DOM", "S.S-DOM", "CS.SW-DOM", "CS.S-DOM", "C.SW-DOM",
    "C.S-DOM", "C.org")
col1 <- c(rep(c("black"), 3), rep(c("#CF5053"), 2), rep(c("black"), 3))
```

```
col2 <- c("#999999", "#FFDB6D", "#E69F00", "#56B4E9", "#009E73", "#F0E442",
    "#0072B2", "#D55E00", "#CC79A7", "#293352")
```

10.1 Create barplot

```
plotB <- ggplot(agg10.long, aes(x = treat, y = value, fill = order)) + geom_bar(stat = "identity") +
    scale_fill_manual(values = col2) + ggtitle("B)") + labs(y = "abundance (%)") +
    scale_x_discrete(limits = positions, labels = c("initial", rep(c("SW-DOM",
        "S-DOM"), 3), "initial")) + theme_classic() + theme(axis.text.x = element_text(angle = 55,
    hjust = 1, size = 14, color = col1), axis.title.x = element_blank()) + theme(legend.text = element_
    legend.title = element_blank()) + theme(axis.text.y = element_text(size = 14),
    axis.title.y = element_text(size = 14)) + theme(plot.title = element_text(size = 18)) +
    theme(plot.title.position = "plot")  #mu title to left

plotB
```

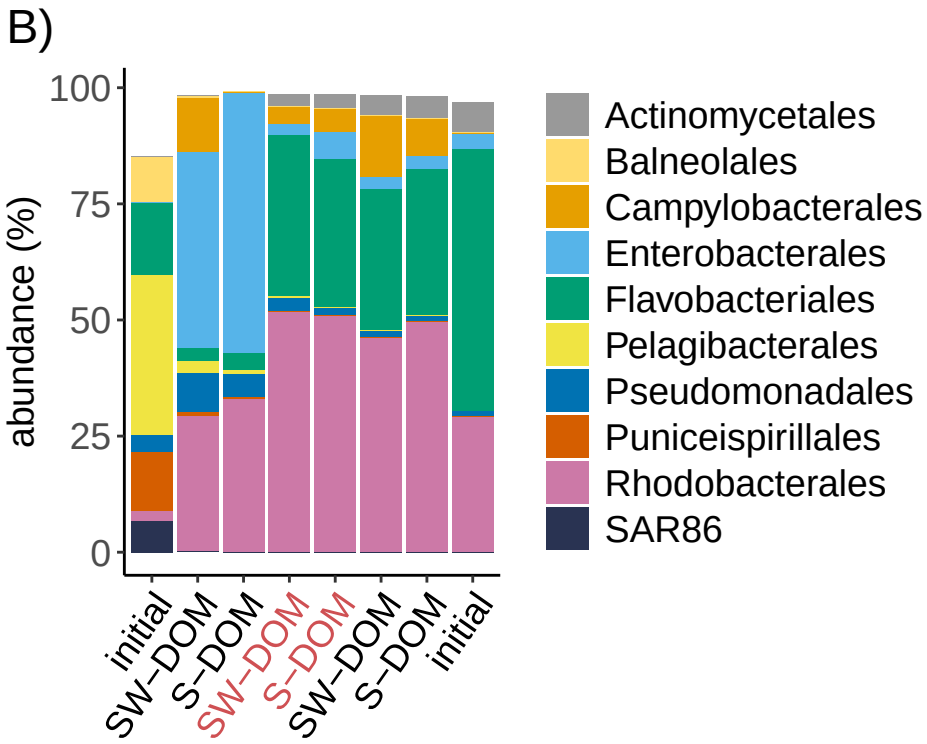

Figure 2: Barplot displaying relative abundances of the top 10 orders in the initial samples and incubation treatments (pooled replicates, hybrid incubations are labelled in red)

10.2 Create PCoA biplot

```
par(mfrow = c(1, 1))
par(mar = c(2, 2.2, 2, 0.8))
plot(pcoa.bray$vectors, pch = symcol$sym, col = as.character(symcol$col), cex = 3,
    xlab = paste("PCOA1 [", round(pcoa.bray$values[2][1, ] * 100, 0), "%]",
        sep = ""), ylab = paste("PCOA2 [", round(pcoa.bray$values[2][2, ] *
        100, 0), "%]", sep = ""))
title(main = "C)", adj = -0, font.main = 14, cex.main = 1.4)
```

```
legend(0, 0.25, c("SOLA", "SW-DOM", "S-DOM", "initial", "", "Hybrid", "SW/S-DOM",
    "", "Canet", "SW-DOM", "S-DOM", "intitial"), cex = 1, col = c("#F7F9FB",
    "darkolivegreen3", "#7F95AA", "gray", "#F7F9FB", "#F7F9FB", "#CF5053", "#F7F9FB",
    "#F7F9FB", "darkolivegreen3", "#7F95AA", "gray"), pch = c(2, 2, 2, 17, 0,
    0, 14, 0, 0, 0, 0, 15), pt.cex = 1.5)
```

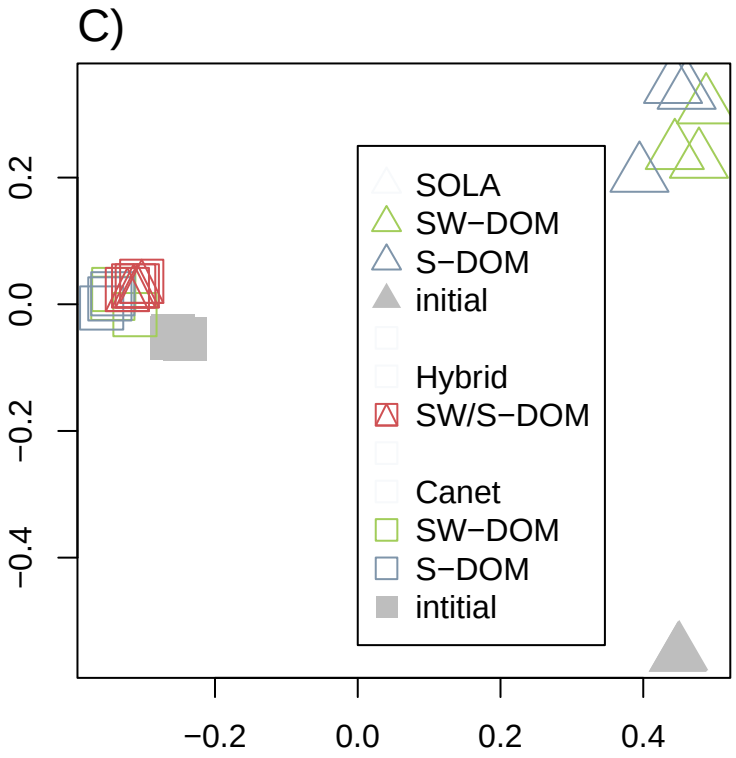

Figure 3: Principal coordinate analysis (PCoA) displaying distances in community ASV composition of initial samples and incubations (day 5) and using the Bray Curtis distance.