BRACE: Application to Oral Microbiome Data

Satabdi Saha

This R Markdown Notebook can be used to reproduce the supplementary and main figures for the real data analysis presented in the Section 4 of the paper "Flexible Aggregation of Compositional Predictors with Shared Effects."

Load the required libraries

```r
library(phyloseq)
library(dplyr)
library(salso)
library(ape)
library(ggtree)
library(pdfCluster)
library(dendextend)
library(ggplot2)
library(reshape2)
library(umap)
```

To illustrate the utility of BRACE, we applied it to data from the Oral Infections, Glucose Intolerance, and Insulin Resistance Study (ORIGINS), which investigated the correlation between periodontal microbiota and insulin resistance. Previous studies have established a significant association between periodontitis, a chronic inflammatory disease affecting the tissues supporting the teeth, and the risk of type 2 diabetes.

The cross-sectional ORIGINS study included 152 adults without diabetes (77% female), aged 20–55 years. We filtered the dataset to exclude samples lacking insulin level information and taxa with a total abundance lower than 30, resulting in 130 taxa and 111 samples. Our goal is to elucidate the relationship between the periodontal microbiome and observed insulin levels using BRACE

Load the phyloseq object: ORIGINS Study.

```r
pseq<-readRDS("/Users/satab/Documents/RevisedBRACElet_Results/Demmer/Demmer_phyloseq_processed.Rdata")
#pseq
```

For faster processing of outputs, we load the saved outputs from running BRACE on the real data example.

```r
load("/Users/satab/Documents/RevisedBRACElet_Results//Demmer/n111_p130DemmerWholeData.RData")
#number of samples after burning
nsamp = 5000
# number of burn samples
nburn = 5000
iterr = nsamp + nburn
n = 111 # Sammple size
p = 130 # Number of taxa/predictors
```

Compute the coefficient vector beta

```
beta_clust_labels<-apply(beta_clust_labels, 2, as.numeric)
#Reconstruct beta
new_beta<- rep(list(vector(mode = "logical", length = p)), times = iterr)
for(iter in 2 : (nsamp+nburn))
{
  for(j in 1: length(beta_clust_labels[,iter]))
  {
    if(beta_clust_labels[,iter][j] == 0)
    {
      new_beta[[iter]][j] = 0
    }
    else
      new_beta[[iter]][j] = theta[[iter]][beta_clust_labels[,iter][j]]
  }
}
sum_new_beta<-sapply(new_beta, sum)
#hist(sum_new_beta)
```

Calculate summary metrics for the beta vector and select the non zero beta coefficients

```
#Create the sum of beta vector
sum_new_beta = sapply(new_beta[(nburn + 1): (nsamp+nburn)], sum)
mean_sum_new_beta = mean(sum_new_beta)
sd_sum_new_beta = sd(sum_new_beta)
#hist(sum_new_beta, breaks = 100)
new_beta_mat = matrix(unlist(new_beta[nburn+1:iterr]), ncol = p, nrow = nsamp, byrow = TRUE)
colnames(new_beta_mat)<-taxa_names(pseq)
median_new_beta = apply(new_beta_mat, 2, function(x) quantile(x, probs = 0.5))
mean_new_beta = colMeans(new_beta_mat)
mean_new_beta = round(mean_new_beta,2)
sd_new_beta = apply(new_beta_mat, 2, sd)
quantile_new_beta = apply(new_beta_mat, 2, function(x) quantile(x, probs = c (0.025, 0.5,0.975)))
#t(quantile_new_beta)

quantile_new_beta<- data.frame(median = quantile_new_beta[2,],
                               lower.ci = quantile_new_beta[1,],
                               upper.ci = quantile_new_beta[3,])
quantile_new_beta <- quantile_new_beta %>%
  mutate(index = row_number())
selected_columns <- quantile_new_beta %>%
  filter(!(lower.ci < 0 & upper.ci > 0))%>%
  filter(!(lower.ci == 0 & upper.ci == 0)) %>%
  filter(!(lower.ci == 0 )) %>%
  filter(!(upper.ci == 0 )) %>%
  filter(!(median == 0))

# Display the selected columns
#dim(selected_columns)
selected_indices <- selected_columns$index
```

To obtain a final clustering from the sampled cluster labels, we ran SALSO for a range of possible cluster numbers.

Supplementary Figure 3: The expected VI loss obtained by varying the number of clusters in the SALSO algorithm, which was used to post-process the cluster labels estimated by BRACElet for the ORIGINS data. The red dot highlights the elbow point, representing the optimal balance between model complexity and fit.

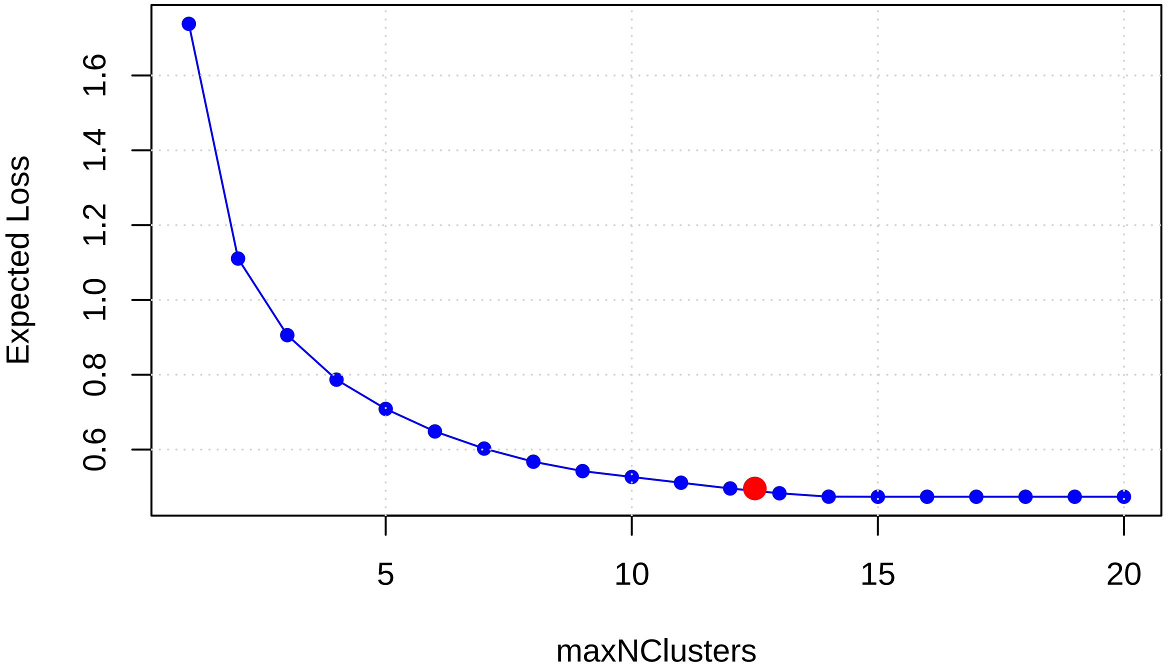
```r
load("/Users/satab/Documents/RevisedBRACElet_Results//Demmer/n111_p130DemmerWholeDataSalsoResults.Rdata
# Extract expectedLoss from each element in summary.results
expected_losses <- sapply(summary.results, function(res) attr(res$estimate, "info")$expectedLoss)

# Display the expected losses
expected_losses
```

```
##  [1] 1.7380163 1.1106268 0.9058393 0.7868692 0.7087107 0.6484237 0.6026735
##  [8] 0.5675410 0.5424812 0.5267524 0.5113678 0.4960591 0.4832468 0.4741022
## [15] 0.4739069 0.4739069 0.4739069 0.4739069 0.4739069 0.4739069
```

```r
# Define the range of maxNClusters
max_n_clusters <- 1:20

# Apply the Kneedle algorithm to find the elbow point
elbow <- inflection::findiplist(max_n_clusters, expected_losses,0)

# The function returns two inflection points, typically we choose the second one
elbow_point <- elbow[2, "chi"]

# Print the elbow point for maxNClusters
print(paste("The elbow point occurs at maxNClusters =", elbow_point))
```

```
## [1] "The elbow point occurs at maxNClusters = 12.5"
```

```r
# Plot the results with the identified elbow

plot(max_n_clusters, expected_losses, type = "o", col = "blue",
     xlab = "maxNClusters", ylab = "Expected Loss", main = "", pch = 16)
points(elbow_point, expected_losses[elbow_point], col = "red", pch = 19, cex = 1.5)
grid()
```



Applying the elbow method, we identified 12.5 as the optimal number of clusters. However, SALSO detected three singleton clusters when the number of clusters exceeded 7. Additionally, the expected loss was 0.50 for 12 clusters and 0.60 for 7 clusters. Given the small difference in expected loss, we opted for seven clusters to maintain model parsimony.

Calculate the final cluster labels using salso

```r
beta_labels_mat<- t(beta_clust_labels)
set.seed(889)
beta_salso<-salso(beta_labels_mat[(nburn+1):iterr,],
                  loss = VI(),
                  maxNClusters = 7,
                  nRuns = 16,
                  maxZealousAttempts = 10,
                  probSequentialAllocation = 0.5,
                  nCores = 3)
summ<-summary(beta_salso)
cluster_labels <-  summ$estimate[selected_indices]
```

To achieve variable selection, we filtered out features that contained zero within their 95% posterior credible intervals

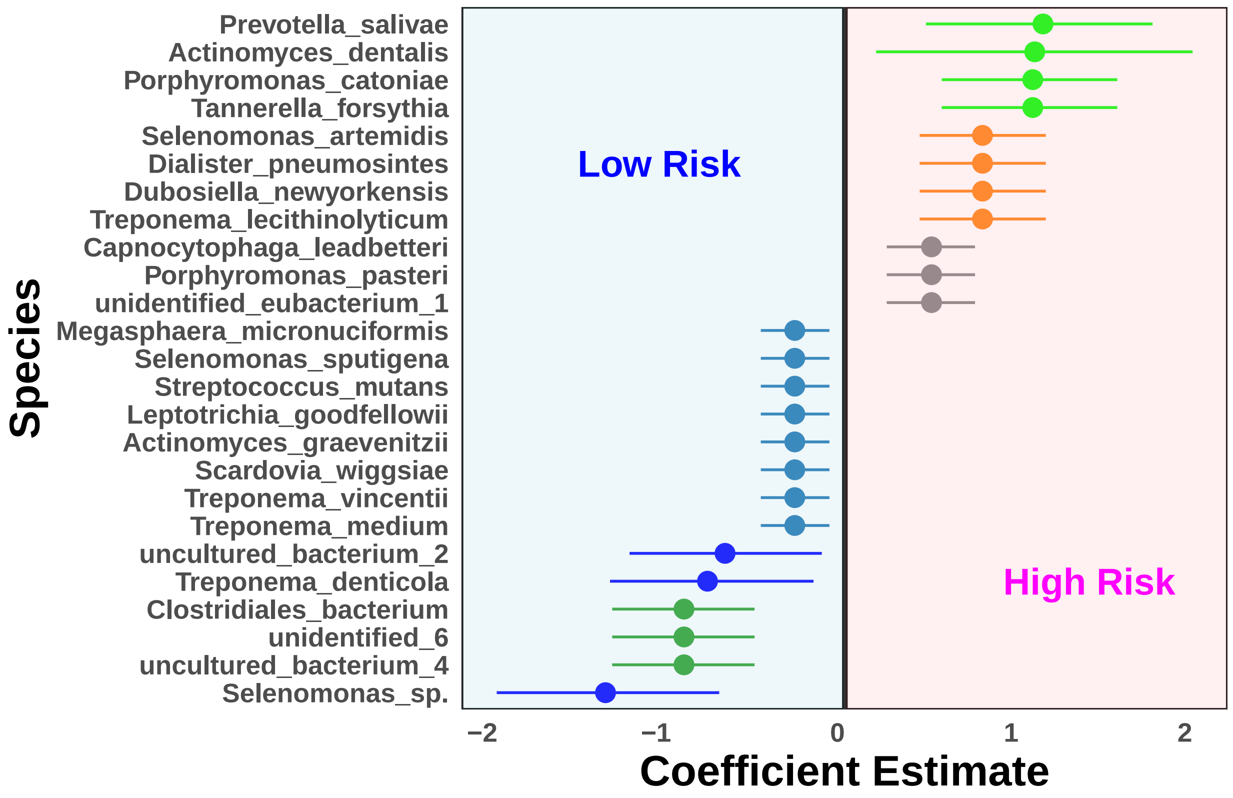
Main paper Figure 3: Parameter estimates and credible intervals for all features selected by BRACE.The color of the dots determines the cluster labels of the estimates.

```r
nonZero_clusterInd<- selected_indices
cluster_indicators <- summ$estimate[selected_indices]
```

```r
#substring(colnames(dist.mat.scaled.nonZero),4))
relabelled_labels <- as.integer(factor(cluster_indicators))
# Example data frame with estimates and credible intervals
data <- data.frame( parameter =  1:length(mean_new_beta),
                    quantile_new_beta[,2], quantile_new_beta[,1], quantile_new_beta[,3])

colnames(data) = c("parameter","lower.ci", "estimate", "upper.ci")
data<- data[selected_indices,]
data$Row_Names <- taxa_names(pseq)[selected_indices]
data$cluster_names<- as.factor(relabelled_labels)
data<- cbind(data, tax_table(pseq)[selected_indices,])
#data$Phylum <- tax_table(pseq)[, "Phylum"]
data$Phylum<- factor(data$Phylum)
#data<- data[order(data$cluster_names),]
palette <- c("#1f77b4", "#ff7f0e", "#2ca02c", "#d62728", "#9467bd",
             "#8c564b", "#e377c2", "#7f7f7f", "#bcbd22", "#17becf",
             "#aec7e8", "#ffbb78", "#98df8a", "#ff9896", "#c5b0d5")


# Create a column to determine the sign of the estimates
data <- data %>%
  mutate(sign = ifelse(estimate < 0, "Negative", "Positive"))

# Order the data: negatives first in descending order, positives in ascending order
data <- data %>%
  arrange(cluster_names, ifelse(sign == "Negative", -abs(estimate), abs(estimate)))

# Reverse the order of the "Low Risk" group while keeping the "High Risk" group as is
data <- data %>%
  mutate(order = ifelse(sign == "Negative", rank(-abs(estimate)), rank(abs(estimate)))) %>%
  arrange(sign, order)

# Set the factor levels for 'Row_Names' based on this order
data$Row_Names<-substring(data$Row_Names,4)
data$Row_Names <- factor(data$Row_Names, levels = data$Row_Names)

# Check the arranged data
#print(data)

# Create the plot
p <- ggplot(data, aes(x = Row_Names, y = estimate, color = cluster_names)) +
  geom_segment(aes(x = Row_Names, xend = Row_Names, y = lower.ci, yend = upper.ci),
               linetype = "solid", size = 0.5) +
  geom_point(size = 3) +
  geom_hline(yintercept = 0, linetype = "solid", color = "black", size = 1) +
  scale_color_manual(values = c(
    "1" = "blue",
    "2" = "green",
    "3" = "#7f7f7f",
    "4" = "#1f77b4",
    "5" = "#ff7f0e",
    "6" = "#2ca02c",
    "7" = "#d62728",
    "8" = "#9467bd",
    "9" = "#8c564b",
    "10" = "#e377c2",
    "11" = "#bcbd22",
    "12" = "#17becf",
    "13" = "#f7b6d2",
    "14" = "#c49c94"
  ))+
  theme_minimal() +
  theme(
    legend.position = "none",
    axis.text = element_text(angle = 0, hjust = 1, size = 10, face = "bold"),
    axis.title = element_text(size = 16, face = "bold"),
    panel.grid.major = element_blank(),
    panel.grid.minor = element_blank(),
    panel.background = element_rect(fill = "white", color = "black"),
    plot.background = element_blank()
  ) +
  labs(title = "",
       x = "Species", y = "Coefficient Estimate", color = "Cluster Labels") +
  annotate("rect", xmin = -Inf, xmax = Inf, ymin = -Inf, ymax = 0, alpha = 0.2, fill = "lightblue") +
  annotate("rect", xmin = -Inf, xmax = Inf, ymin = 0, ymax = Inf, alpha = 0.2, fill = "lightpink") +
  annotate("text", x = 5, y = max(data$upper.ci), label = "High Risk", fontface = "bold",hjust = 1.1, s
  annotate("text", x = 20, y = min(data$lower.ci)+ 1.5, label = "Low Risk", fontface = "bold",hjust = 1
  coord_flip()
```

```
## Warning: Using ‘size‘ aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use ‘linewidth‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

```
# Display the plot
print(p)
```



BRACE identified 25 features grouped into seven nonzero clusters. It selected species belonging to the phyla Firmicutes and Bacteroidota to be associated with insulin levels. At the genus level, BRACE identified a number of species belong to *Porphyromonas*, *Tanerella*, and *Treponema*.

Our next goal was to investigate whether these BRACE identified taxa clusters demonstrated phylogenetic similarity. To achieve that goal, we first constructed a phylogenetic tree based on the representative sequences for the observed taxa.

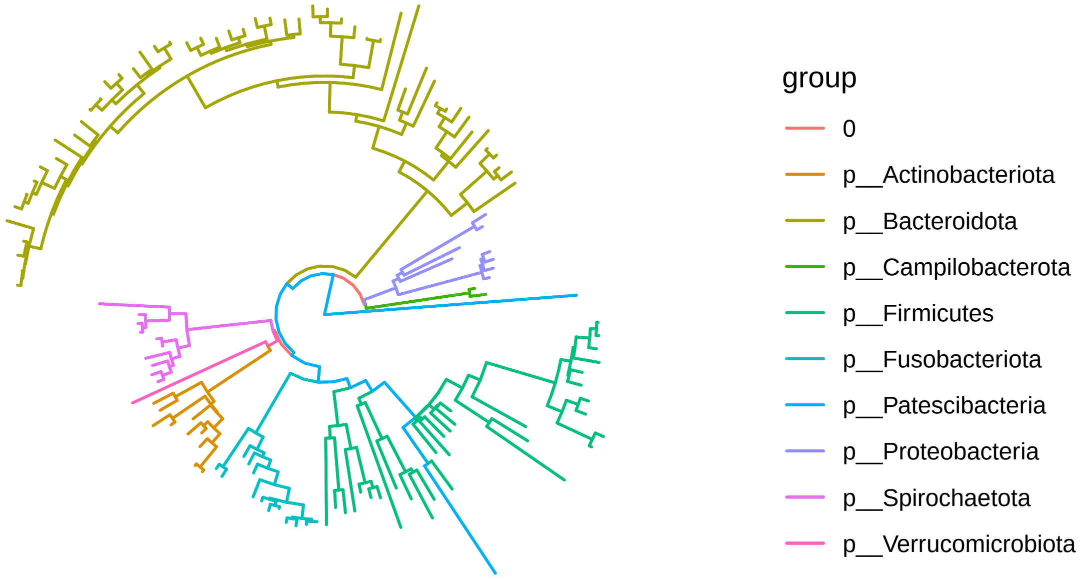
Supplementary Figure 4: The true phylogenetic tree generated from the ORIGINS data. The colors of the branches have been annotated to represent the nine phyla in the data.

```r
tree = phy_tree(pseq)
groupInfo <- split(tree$tip.label, tax_table(pseq)[,"Phylum"])
# Use groupOTU to add grouping information to the tree
tree_grouped <- groupOTU(tree, groupInfo)

# Plot the tree with ggtree and apply color by group
tree.phylo <- ggtree(tree_grouped, aes(color = group), layout = 'circular') +
  geom_tree(aes(width = 10)) #+
```

```
## Found more than one class "phylo" in cache; using the first, from namespace 'phyloseq'

## Also defined by 'tidytree'
```

```r
  #guides(color = "none") # Use color instead of fill for guide
```

```
# Display the plot
tree.phylo
```



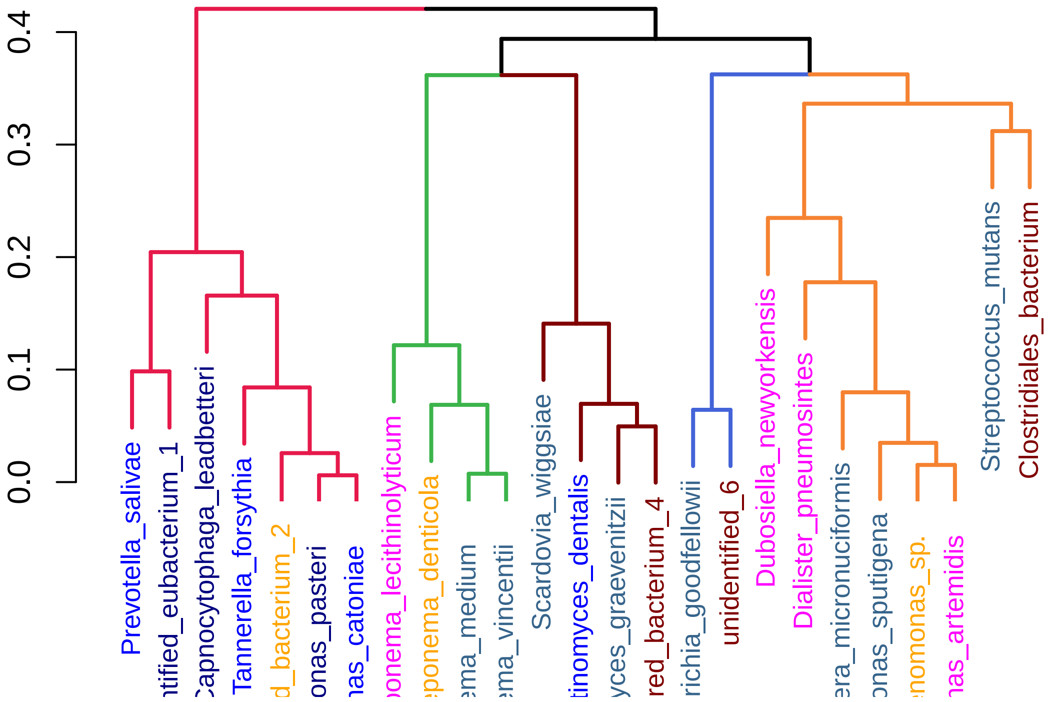
Next, we calculated the phylogenetic correlation matrix

```
dist.mat.scaled = vcv(tree, corr = TRUE)
#split(mean_new_beta[selected_indices],cluster_labels)
df1<-data.frame(species = mean_new_beta[selected_indices], labels = cluster_labels)
#round(as.vector(sapply(split(mean_new_beta[selected_indices],cluster_labels),mean)),3)
cluster_means<-sapply(split(mean_new_beta[selected_indices],cluster_labels), mean)
cluster_sd<-sapply(split(mean_new_beta[selected_indices],cluster_labels), sd)
cluster_medians<-sapply(split(mean_new_beta[selected_indices],cluster_labels), function(x) quantile(x, 
cluster_lower.ci<-sapply(split(mean_new_beta[selected_indices],cluster_labels),
                         function(x)  quantile(x, probs = 0.025))
cluster_upper.ci<-sapply(split(mean_new_beta[selected_indices],cluster_labels),
                         function(x)  quantile(x, probs = 0.975))


nonZero_clusterInd<- selected_indices
cluster_indicators <- summ$estimate[selected_indices]
dist.mat.scaled.nonZero<-dist.mat.scaled[nonZero_clusterInd,nonZero_clusterInd]
```

We then applied hierarchical clustering on the correlation matrix to understand the overlap between our estimated cluster labels and a clustering based on phylogenetic similarity.

Supplementary Figure 5: A hierarchical clustering dendrogram of the true phylogenetic correlation matrix for the features selected by BRACE. The colors of the labels represent the cluster labels estimated BRACE, whereas the colors of the branches are based on phylum annotations obtained from external reference information.

```
relabelled_labels <- as.integer(factor(cluster_indicators))
hc<-hclust(as.dist(1-dist.mat.scaled.nonZero), method = "complete")
#dend <- as.dendrogram(hclust(as.dist(1 - summ$psm), method = "complete"))
dend <- as.dendrogram(hclust(as.dist(1-dist.mat.scaled.nonZero), method = "complete"))
# Assign colors to the dendrogram based on cluster indicators
#num_colors = max(summ$estimate)
indicator_vector <-tax_table(pseq)[nonZero_clusterInd,"Phylum"]
color_palette = c('#e6194b', '#3cb44b', '#800000', '#4363d8', '#f58231', '#911eb4',
                  '#46f0f0', '#f032e6', '#bcf60c', '#fabebe', '#008080','#000000' ,
                  '#9a6324', "palevioletred4", '#ffe119', '#aaffc3', '#808000', '#ffd8b1',
                  '#000075', '#808080', '#ffffff', '#e6beff')
# Convert the factor vector to a character vector and use it to index the color palette
color_vector <-color_palette[as.numeric(as.factor(indicator_vector))]
num_colors = length(table(indicator_vector))
dend_colored <- dend %>%
  color_branches(k = num_colors, col = color_palette[1:num_colors]) #%>%
#hang.dendrogram(hang.height = 0.06)
#plot(dend_colored)
custom_palette <- c("#FFA500", "#0000FF", "#000080", "steelblue4",
                    "#FF00FF", "#800000", "#008000", "palevioletred4" , "#808000",
                    "#008080", "#800080", "#C0C0C0", "#FFA500")

# Example indicator vector (replace with your actual indicator vector)
#indicator_vector <-as.factor(tax_table(pseq)[,"Phylum"])
#labels(dend_colored)
num_colors = max(relabelled_labels)
# Plot the colored dendrogram
#plot(dend_colored, main = "Dendrogram with Colored Leaves",lwd = 5)
labels_colors(dend_colored) <- custom_palette[relabelled_labels[match(labels(dend_colored),rownames(df1
labels(dend_colored)<- substring(labels(dend_colored),4)

rescale_heights <- function(dend, scale_factor) {
  dend <- dendrapply(dend, function(node) {
    if (is.leaf(node)) return(node) # Don't modify leaves
    attr(node, "height") <- attr(node, "height") * scale_factor
    node
  })
  return(dend)
}

# Apply the scaling factor to the dendrogram heights
dend_colored <- rescale_heights(dend_colored, scale_factor = 0.5) # Compress the height by 50%

dend_colored <- hang.dendrogram(dend_colored, hang_height = 0.05)%>% set("labels_cex", 0.8) %>% set("bra
#pdf("/Users/satab/Documents/RevisedBRACElet_Results/Demmer/DPClustering&ClustersDendogramColoredGenusA
#par(mar = c(20, 0, 0, 0))
plot(dend_colored, sub = NULL)
```



```
#dev.off()
```

Figure demonstrates that there is a small degree of overlap between the BRACE-determined cluster labels and the phylum-level groupings in the oral microbiome. This suggests that the groupings learned from BRACE may offer additional insight on features with similar functional effects, and do not simply recapitulate known taxonomy.

Next, given the estimated cluster labels, we calculated the within-cluster and between-cluster mean phylogenetic correlations, the hypothesis being that the within-cluster phylogenetic correlations will be higher than the between-cluster ones in most cases. T

Main Paper Figure 4 - Table : Table showing the average phylogenetic correlations within and between the clusters for selected features.

```
correlation_matrix <- dist.mat.scaled.nonZero
relabelled_labels <- as.integer(factor(cluster_indicators))
cluster_labels <- relabelled_labels

correlation_matrix <- dist.mat.scaled
cluster_labels <- summ$estimate

# Calculate the average correlations within and between clusters
calculate_correlation_statistics <- function(correlation_matrix, cluster_labels) {
  num_clusters <- length(unique(cluster_labels))
  within_cluster_correlations <- numeric(num_clusters)
  between_cluster_correlations <- matrix(NA, nrow = num_clusters, ncol = num_clusters)
  # Compute within-cluster correlations
  for (i in 1:num_clusters) {
    within_indices <- which(cluster_labels == i)
    within_correlations <- correlation_matrix[within_indices, within_indices]
    within_cluster_correlations[i] <- mean(within_correlations, na.rm = TRUE)
  }

  # Compute between-cluster correlations
  for (i in 1:num_clusters) {
    for (j in 1:num_clusters) {
      if (i != j) {
        between_indices_i <- which(cluster_labels == i)
        between_indices_j <- which(cluster_labels == j)
        between_correlations <- correlation_matrix[between_indices_i, between_indices_j]
        between_cluster_correlations[i, j] <- mean(between_correlations, na.rm = TRUE)
      }
    }
  }

  list(within_cluster_correlations = within_cluster_correlations,
       between_cluster_correlations = between_cluster_correlations)
}

# Calculate statistics
statistics <- calculate_correlation_statistics(correlation_matrix, cluster_labels)
diag(statistics$between_cluster_correlations) <- statistics$within_cluster_correlations
#print(statistics$within_cluster_correlations)
print(round(statistics$between_cluster_correlations,2))
```

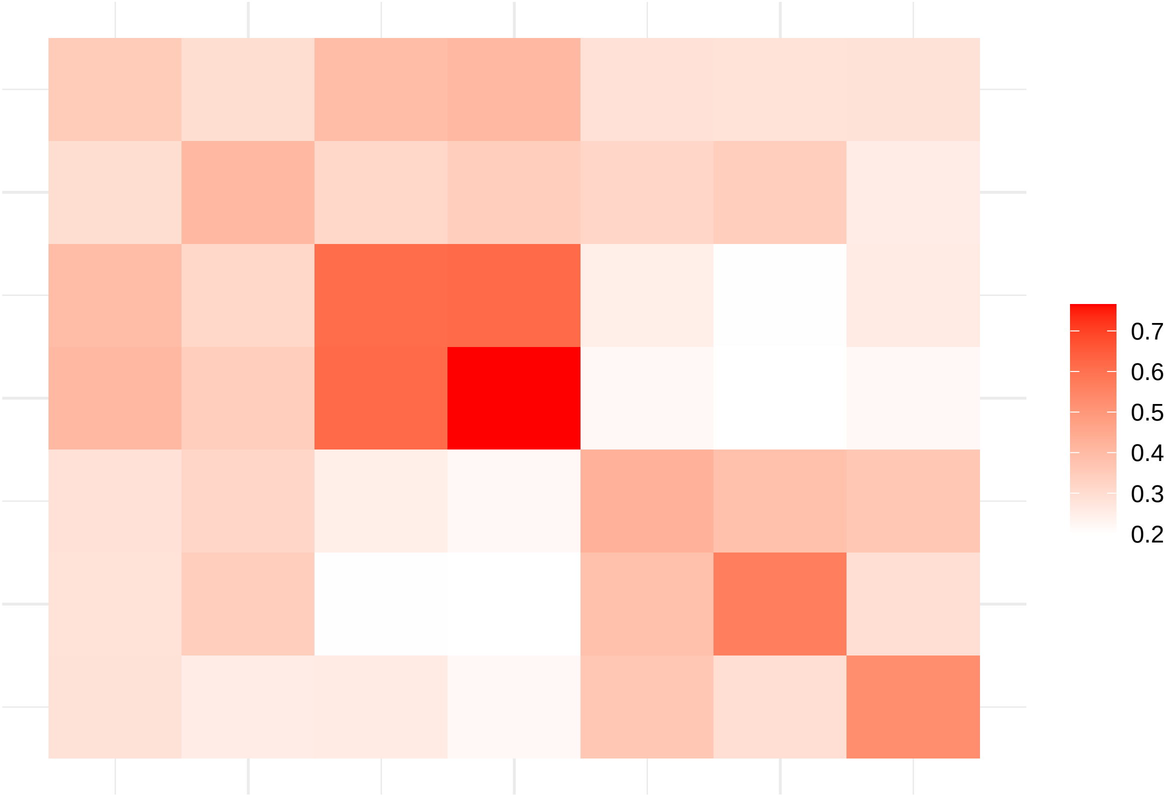
```
##      [,1] [,2] [,3] [,4] [,5] [,6] [,7]
## [1,] 0.35 0.30 0.40 0.41 0.29 0.28 0.29
## [2,] 0.30 0.41 0.32 0.35 0.32 0.34 0.25
## [3,] 0.40 0.32 0.61 0.62 0.25 0.20 0.26
## [4,] 0.41 0.35 0.62 0.77 0.22 0.20 0.22
## [5,] 0.29 0.32 0.25 0.22 0.43 0.38 0.36
## [6,] 0.28 0.34 0.20 0.20 0.38 0.57 0.30
## [7,] 0.29 0.25 0.26 0.22 0.36 0.30 0.53
```

The proportion of times the within-cluster correlations was greater than the between-cluster correlations, for all features, is 1.00, 1.00, 1.00, 1.00, 0.83, 1.00, 1.00, 0.67, 1.00, 0.83, 1.00, 1.00, 1.00, 1.00.

Generate Main Paper Figure 4 - Heatmap Figure 4 : Heatmap showing the average phylogenetic correlations within and between the clusters for selected features.

```
# Flip the matrix vertically
flipped_matrix <- statistics$between_cluster_correlations[nrow(statistics$between_cluster_correlations)

# Melt the matrix for ggplot
melted_matrix <- melt(flipped_matrix)

# Adjust row and column names
rownames(flipped_matrix) <- rev(rownames(statistics$between_cluster_correlations))
colnames(flipped_matrix) <- colnames(statistics$between_cluster_correlations)
```

```r
# Create the ggplot heatmap
p <- ggplot(melted_matrix, aes(Var2, Var1, fill = value)) +
  geom_tile() +
  scale_fill_gradient(low = "white", high = "red") +
  theme_minimal() +
  theme(
    axis.text.x = element_blank(),
    axis.text.y = element_blank()
  ) +
  labs(x = "", y = "", fill = "")

# Display the plot
print(p)
```



This result demonstrates that our model's clustering scheme successfully captures some phylogenetic similarity between the taxa, without the need to prespecify this information as a model input.

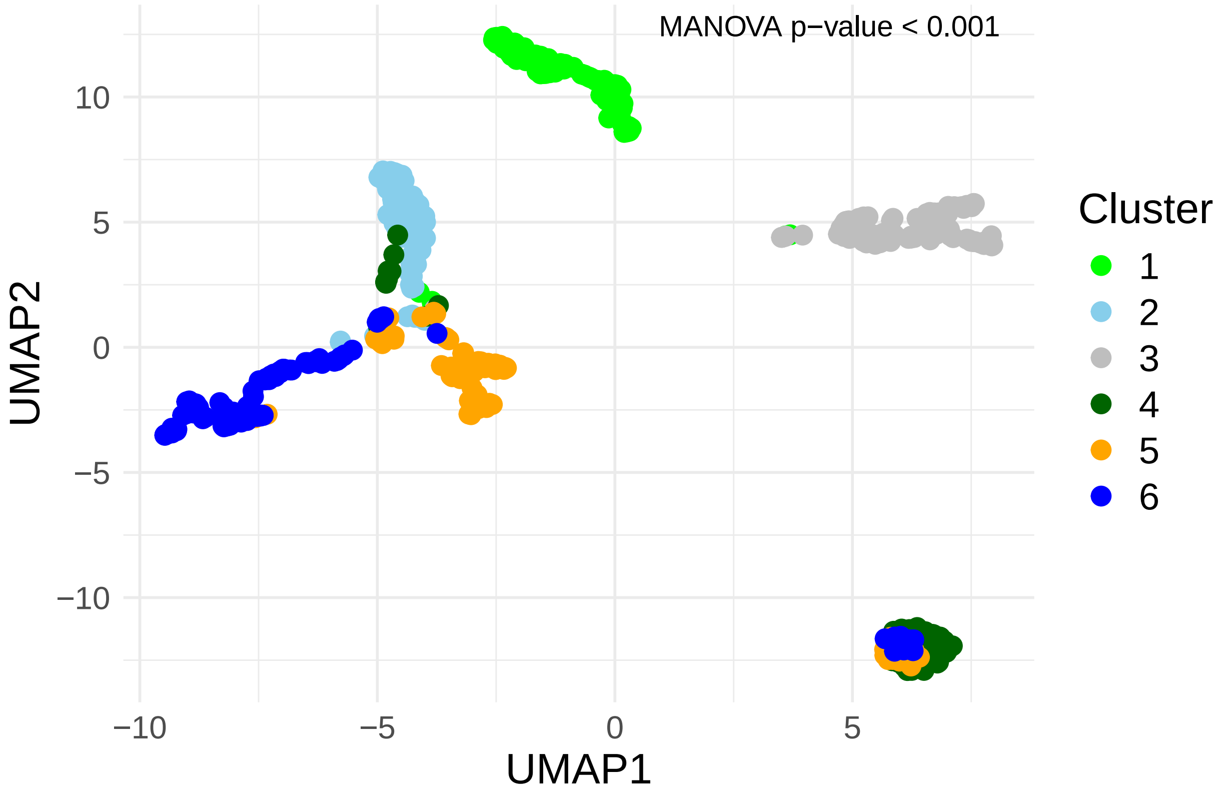
To explore whether the clusters identified by BRACE exhibited distinct functional roles, we performed UMAP of functional feature distributions across six clusters identified by BRACE and refined with SALSO partitioning.

Supplementary Figure 6 : UMAP visualization of functional feature distributions across six clusters identified by BRACE and refined with SALSO partitioning. A MANOVA analysis on the UMAP coordinates yielded a significant p-value ($p < 0.01$), indicating significant differ- ences in functional abundances across clusters.

```
#Load the count matrices for the functional variables corresponding to the selected species
load("/Users/satab/Documents/RevisedBRACElet_Results/Demmer/species_group.RData")
#names(species_grouped_list)
cluster_labels<- c(1,2,3,4,5,
                   5,2,2,1,3,
                   1,2,5,6,2,
                   2,1,6,5,2,
                   2, 6,4,4, 3)

color_vector<-c("green","skyblue","gray",
                "darkgreen","orange",
                "blue")

species_list_all <- lapply(species_grouped_list, function(mat) {
  match_ind <- which(colnames(mat) %in% colnames(otu_table(pseq)))
  species_list <- as.data.frame(mat[, match_ind])
  species_list <- as.data.frame(apply(species_list, 2, as.numeric))
  colnames(species_list) <- colnames(mat[, match_ind])
  return(species_list)
})


# Get unique cluster labels
unique_clusters <- unique(cluster_labels)

# Initialize a list to store the merged matrices for each cluster
merged_list <- list()

# Loop over each unique cluster
for (cluster in unique_clusters) {

  # Get the indices of matrices that belong to the current cluster
  cluster_indices <- which(cluster_labels == cluster)

  # Extract the matrices corresponding to this cluster
  matrices_in_cluster <- species_list_all[cluster_indices]

  # Ensure that the matrices can be merged (check for matching dimensions)
  if (length(matrices_in_cluster) > 1) {
    # Initialize an empty matrix to accumulate the sum, averaging them by summing and dividing
    merged_matrix <- Reduce("+", matrices_in_cluster) / length(matrices_in_cluster)
  } else {
    # If there's only one matrix in the cluster, just use that
    merged_matrix <- matrices_in_cluster[[1]]
  }

  # Store the averaged matrix in the merged list, using the cluster number as the key
  merged_list[[as.character(cluster)]] <- merged_matrix
}
# merged_list now contains one averaged matrix for each cluster

# Assuming:
# - merged_list: list of aggregated matrices (functions as rows, samples as columns)
# - outcome: a vector of continuous values (with the same length as the number of samples in each matri
match_ind_rev<-which(colnames(otu_table(pseq)) %in% colnames(species_grouped_list[[2]]))
insulin_new<-as.numeric(sample_data(pseq)$insulinv1[match_ind_rev])
outcome<-insulin_new
# Log-transform the matrices and handle zero values by adding a small constant (e.g., 1)
log_transformed_list <- lapply(merged_list, function(mat) {
  log(mat + 1)  # Log-transform and handle zeros
})

library(ggplot2)
# Combine merged matrices into a single data frame
# Each row is a function, and columns are samples from all clusters
combined_matrix <- do.call(cbind, log_transformed_list)
# Transpose combined_matrix to have samples as rows and features as columns
transposed_matrix <- t(combined_matrix)

# UMAP
set.seed(145)
umap_result <- umap(transposed_matrix)  # Run UMAP on the transposed matrix
cluster_labels<-rep(names(log_transformed_list), sapply(log_transformed_list, ncol))
umap_data <- data.frame(umap_result$layout, Cluster = cluster_labels)
colnames(umap_data) <- c("UMAP1", "UMAP2", "Cluster")


# Convert UMAP results to a data frame and add cluster labels
umap_data <- data.frame(UMAP1 = umap_result$layout[, 1],
                        UMAP2 = umap_result$layout[, 2],
                        Cluster = factor(cluster_labels))

# Define your color vector
color_vector <- c("green", "skyblue", "gray", "darkgreen", "orange", "blue")


# Perform MANOVA on both PC1 and PC2
manova_result <- manova(cbind(UMAP1, UMAP2) ~ Cluster, data = umap_data)
summary(manova_result)
```

```
##            Df Pillai approx F num Df den Df    Pr(>F)
## Cluster     5 1.4438   339.57     10   1308 < 2.2e-16 ***
## Residuals 654
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
```

```
# Plot with custom colors for clusters
umap_plot<-ggplot(umap_data, aes(x = UMAP1, y = UMAP2, color = Cluster)) +
  geom_point(size = 3) +
  labs(title = "") +
  scale_color_manual(values = color_vector) +
  theme_minimal()+
  theme(
    axis.text = element_text(size = 12),  # Adjusts size of axis tick labels
    axis.title = element_text(size = 16),  # Adjusts size of axis titles
    legend.title = element_text(size = 16),
    legend.text = element_text(size = 14)
  )+
  annotate("text", x = Inf, y = Inf, label = "MANOVA p-value < 0.001",
           hjust = 1.1, vjust = 1.5, size = 4, color = "black")  # Adjust position and size
umap_plot
```



```
# Perform ANOVA for UMAP1
anova_umap1 <- aov(UMAP1 ~ Cluster, data = umap_data)
summary(anova_umap1)
```

```
##              Df Sum Sq Mean Sq F value Pr(>F)    
## Cluster       5  12255  2451.1   214.6 <2e-16 ***
## Residuals   654   7469    11.4                   
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Perform ANOVA for UMAP2
anova_umap2 <- aov(UMAP2 ~ Cluster, data = umap_data)
summary(anova_umap2)
```

```
##              Df Sum Sq Mean Sq F value Pr(>F)    
## Cluster       5  28518    5704     318 <2e-16 ***
## Residuals   654  11729      18                   
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The significant MANOVA result indicates meaningful differences in functional abundances across the clusters, implying that BRACE captures underlying functional variability associated with the outcome. This differentiation supports the clustering property of BRACE as an effective approach that captures both phylogenetic and functional similarity. This outcome strengthens confidence in BRACE's ability to reveal biologically meaningful patterns within the data.