Scripts for figures for Plant-derived benzoxazinoids act as antibiotics and shape bacterial communities

Niklas Schandry

27 4 2021

Contents

About

This file contains code for the generation of the main Figures in Schandry et al. Plant-derived benzoxazinoids act as antibiotics and shape bacterial communities. Detailed scripts for the analysis of individuals and trees can be found in *Growth_trees.Rmd*. The read processing is documented in *Reads_in.Rmd* and the community analysis is documented in *Community_analysis.Rmd*.

Prepare session

Libraries and functions

```
library(tidyverse)
library(phyloseq)
library(patchwork)
library(magrittr)
library(ggnetwork)
library(correlation)
library(ggtree)
library(treeio)

source("functions/get_counts_and_overlay.R")
source("functions/phyloseq_corrs.R")
source("functions/overlay_corrnet.R")
source("functions/fix_node_placement.R")
source("functions/compare_conditions.R")
source("functions/read_fasta.R")
```

```
syncom_colors <- c("Random" = "#666699",
                   "Tolerant" = "#9BCB40",
                   "Mixed" = "#B6539F",
                   "Sensitive" = "#FEE377")
```

Figure 1

Data in

```
### Taxonomy
tax_info <- read_rds("files_publication/taxonomy.rds")


### read tree

fit_GTR <- read_rds("files_publication/Strain_tree_16S_gyrB_rhoB_recA_dnaJ_atpD_thrC.rds")

### read growth data
AUC_emmeans <- read_rds("files_publication/AUC_emmeans_pub.rds") %>%
  mutate(Family = case_when(Genus == "Rhizobacter" ~ "Burkholderiales",
                            TRUE ~ Family)) %>%
  left_join(tax_info %>%
              dplyr::select(Genus, Class, Order), by = "Genus") %>%
  filter(Strain %in% fit_GTR$tree$tip.label) ## Keep only those with sequences

tree_data <- AUC_emmeans %>%
  mutate(id = str_c(Genus,Strain ,sep ="_"))

fit_GTR$tree$tip.label %<>%
  as.data.frame() %>%
  set_colnames("Strain") %>%
  left_join(., ( AUC_emmeans %>% dplyr::select(Strain, Genus) %>% distinct)) %>%
  mutate(Strain = str_c(Genus,Strain, sep ="_")) %>%
```

```
  dplyr::select(-Genus) %>%
  as.vector() %>%
  unlist %>%
  unname
```

```
## Joining, by = "Strain"
```

```
syncoms <-  rbind(
                 read_rds("files_publication/syncom_reduced_mixed.rds") %>%
                  dplyr::select(Strain) %>%
                  mutate(Syncom = "Reduced Mixed"),
                read_rds("files_publication/syncom_reduced_tolerant.rds") %>%
                  dplyr::select(Strain) %>%
                  mutate(Syncom = "Reduced Tolerant"),
                read_rds("files_publication/syncom_reduced_sensitive.rds") %>%
                  dplyr::select(Strain) %>%
                  mutate(Syncom = "Reduced Sensitive"),
                read_rds("files_publication/syncom_full_random.rds") %>%
                  dplyr::select(Strain) %>%
                  mutate(Syncom = "Random")
                )
# Define colors for plotting
set.seed(42069)
family_colors <-  sample(c('#543005','#8c510a','#bf812d',
                           '#dfc27d','#f6e8c3','#4d9221',
                           '#c7eae5','#80cdc1','#35978f',
                           '#01665e','#003c30','#8e0152',
                           '#c51b7d','#de77ae','#f1b6da',
                           '#fde0ef','#000000','#e6f5d0',
                           '#b8e186','#7fbc41','#fb8072',
                           '#276419','#40004b','#bababa'),24)
names(family_colors) <- AUC_emmeans$Family %>% unique %>% sort
```

Figure 1A

Prepare data

```
### Below creates "Group Info", which is currently the Genus. Extracted from tiplabels, see above chunk
tree_structure <- fit_GTR$tree %>% as.phylo
groupInfo <- split(fit_GTR$tree$tip.label, gsub("_\\w+", "", fit_GTR$tree$tip.label))
tree_structure <- groupOTU(tree_structure, groupInfo)

tree_plot_data <- cbind(
           tree_data %>%
             filter(Chem == "APO") %>%
             mutate(Chem_conc = paste0(Chem,Conc)) %>%
             dplyr::select(id, Chem_conc, estimate) %>%
             distinct %>%
             spread(Chem_conc, estimate) %>%
             as.data.frame() %>%
             dplyr::select("id","APO1","APO5","APO10","APO50") %>%
             separate(id, into = c("Genus","Strain"), sep = "_", remove = FALSE) %>%
             left_join(., AUC_emmeans %>% dplyr::select(Strain, Family, Order, Class, Phylum), by = "Str
             distinct()  ,
```

```
          dummy1 = c(rep(NA,174)),
          dummy2 = c(rep(NA,174)),
         tree_data %>% filter(Chem == "BOA")  %>%
             mutate(Chem_conc = paste0(Chem,Conc)) %>%
             dplyr::select(id,Chem_conc, estimate) %>%
             distinct %>%
             spread(Chem_conc, estimate) %>%
             dplyr::select("BOA10","BOA50","BOA100") %>%
             as.data.frame())



tree_data_boa <-  tree_plot_data %>%
           dplyr::select("id","BOA10","BOA50","BOA100") %>%
           column_to_rownames("id") %>%
           set_colnames(c("10µM BOA", "50µM BOA", "100µM BOA"))
tree_data_apo <- tree_plot_data %>%
           dplyr::select("id","APO1","APO5","APO10","APO50") %>%
           column_to_rownames("id") %>%
           set_colnames(c("1µM APO", "5µM APO", "10µM APO", "50µM APO"))
# Make a wide table of memberships, since syncoms were rbinded, there is one entry for each strain/sync


syncom_memberships <- left_join(tree_plot_data %>% dplyr::select(Strain, id),
  syncoms %>%
    mutate(member = as.factor(case_when(
                               Syncom == "Reduced Mixed" ~ "Mixed",
                               Syncom == "Reduced Tolerant" ~ "Tolerant",
                               Syncom == "Reduced Sensitive" ~ "Sensitive",
                               Syncom == "Random" ~ "Random",
                               TRUE ~ "NA"))) %>%
             pivot_wider(names_from = Syncom, values_from = member),
  by = "Strain")
```

Figure 1A - Plot

```
fan_tree_family <- tree_structure %>%
  ggtree(branch.length = "none",
         layout = "fan",
         open.angle = 10) %<+% (tree_plot_data %>% mutate(Phylum_Family = paste(Phylum, Family, sep = "
```

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

```
## Also defined by 'tidytree'
```

```
## Scale for 'y' is already present. Adding another scale for 'y', which will
## replace the existing scale.
```

```
fan_tree_family <- fan_tree_family +
    geom_tippoint(aes(color =  Family,
                      shape = Class),
                  size = 1) +
  scale_color_manual("Family",
                     values = family_colors,
                     guide = "legend") +
```

```
  scale_shape(breaks = c("Alphaproteobacteria",
                         "Betaproteobacteria",
                         "Gammaproteobacteria",
                         "Actinomycetia",
                         "Bacteroidia",
                         "Bacilli"))
```

```
fan_tree_family <- fan_tree_family + guides(Family = guide_legend(ncol = 3, byrow = TRUE))
heat_boa_fam <- gheatmap(fan_tree_family,
         tree_data_boa,
         offset=0.3,
         width=0.2,
         colnames_angle = 90,
         colnames = T,
         hjust = 1,
         font.size = 3,
         legend_title = "Rel. growth")  +
  scale_fill_viridis_c(na.value = "white",
                       direction = -1,
                       name = "Rel. growth")
```

```
## Scale for 'fill' is already present. Adding another scale for 'fill', which
## will replace the existing scale.
```

```
heat_boa_apo_fam <- gheatmap(heat_boa_fam,
         tree_data_apo,
         offset=5.7,
         width=0.2,
         colnames_angle = 90,
         colnames = T,
         hjust = 1,
         font.size = 3,
         legend_title = "Rel. growth") +
  scale_fill_viridis_c(na.value = "white",
                       direction = -1,
                       name = "Rel. growth") +
  theme(text = element_text(family = "NimbusMon"))
```

```
## Scale for 'fill' is already present. Adding another scale for 'fill', which
## will replace the existing scale.
## Scale for 'fill' is already present. Adding another scale for 'fill', which
## will replace the existing scale.
```

```
heat_boa_apo_fam <- heat_boa_apo_fam + ggnewscale::new_scale_fill()
heat_boa_apo_fam_syncom <- gheatmap(heat_boa_apo_fam,
         syncom_memberships %>%
           dplyr::select("id","Random",
                         "Reduced Tolerant",
                         "Reduced Mixed",
                         "Reduced Sensitive") %>%
           column_to_rownames("id"),
         offset=11.5,
         width=0.2,
         colnames_angle = 90,
         hjust = 0.5,
```

```
        font.size = 0) +
  scale_fill_manual(na.value = "grey80",
                            name = "Syncom",
                            values = syncom_colors) +
  theme(text = element_text(family = "NimbusMon"),
        legend.position = "left")
```

```
## Scale for 'fill' is already present. Adding another scale for 'fill', which
## will replace the existing scale.
```

```
heat_boa_apo_fam_syncom + plot_annotation(title = "Fig 1") & theme(text = element_text(family = "Nimbusl
```

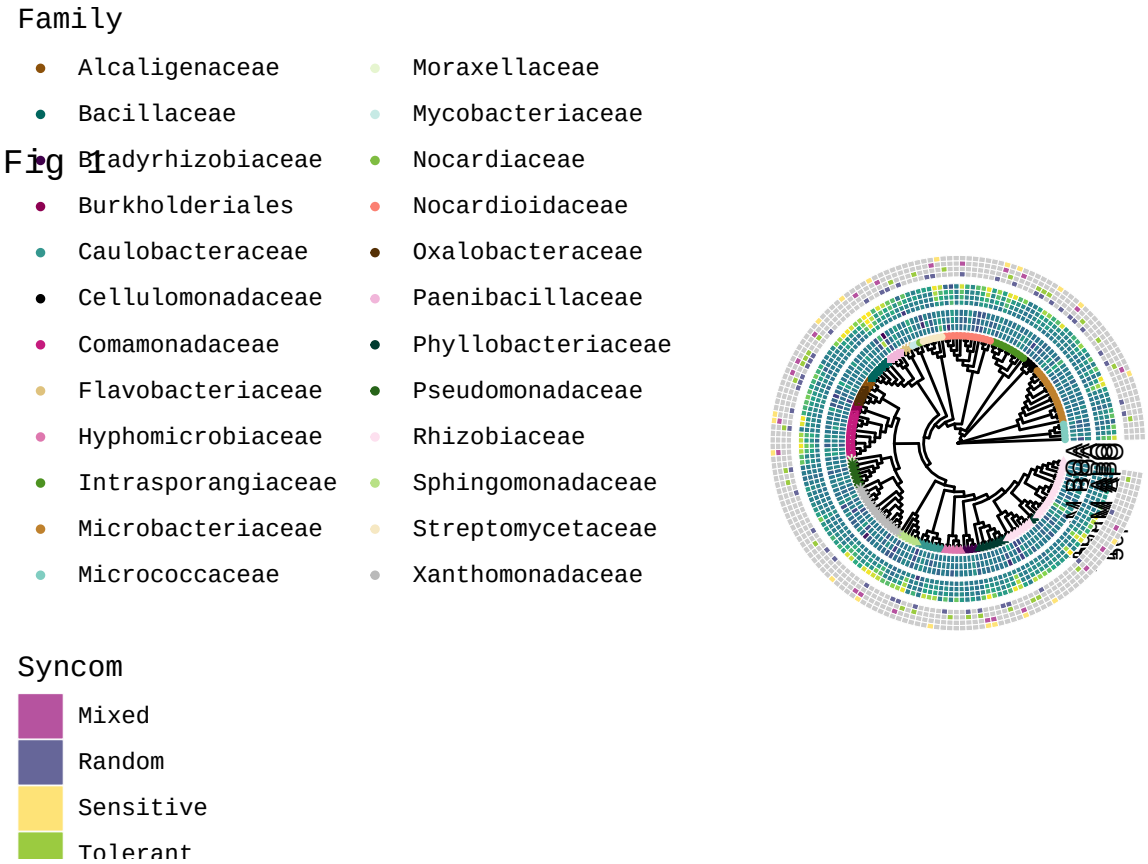


Fig 1

Figure 1B - Plot

```
AUC_emmeans %>%
  filter(Treat.x  == "50uM_APO") %>%
  ggplot() +
  geom_point(aes(y = Family,
                 x = estimate,
                 fill = Family,
                 group = Strain),
             pch =21,
             alpha=0.3,
             size = 3,
             stat = "summary") +
  ggridges::geom_density_ridges(aes(x=AUC_norm,
```

```
                                    y=Family,
                  color = Family,
                  fill = Family),
                  alpha = 0.4) +
  facet_wrap(~fct_relevel(Class,
                          "Alphaproteobacteria",
                         "Betaproteobacteria",
                         "Gammaproteobacteria",
                         "Actinomycetia",
                         "Bacteroidia",
                         "Bacilli"),
             scales = "free_y") +
  scale_color_manual(values = family_colors) +
  scale_fill_manual(values = family_colors) +
  theme_bw(base_family = "NimbusMon") +
  theme(strip.background = element_rect(fill = "white"),
        legend.position = "none",
        axis.text.y = element_blank(),
        axis.ticks.y = element_blank()) +
  labs(x = "AUC[APO] / AUC[Control]") +
  lims(x=c(-0.3,1.6)) +
  NULL
```

```
## No summary function supplied, defaulting to `mean_se()`
## No summary function supplied, defaulting to `mean_se()`
## No summary function supplied, defaulting to `mean_se()`
## No summary function supplied, defaulting to `mean_se()`
## No summary function supplied, defaulting to `mean_se()`
## No summary function supplied, defaulting to `mean_se()`
```

```
## Picking joint bandwidth of 0.115
```

```
## Picking joint bandwidth of 0.0988
```

```
## Picking joint bandwidth of 0.0724
```

```
## Picking joint bandwidth of 0.12
```

```
## Picking joint bandwidth of 0.128
```

```
## Picking joint bandwidth of 0.112
```

```
## Warning: Removed 9 rows containing non-finite values (stat_density_ridges).
```

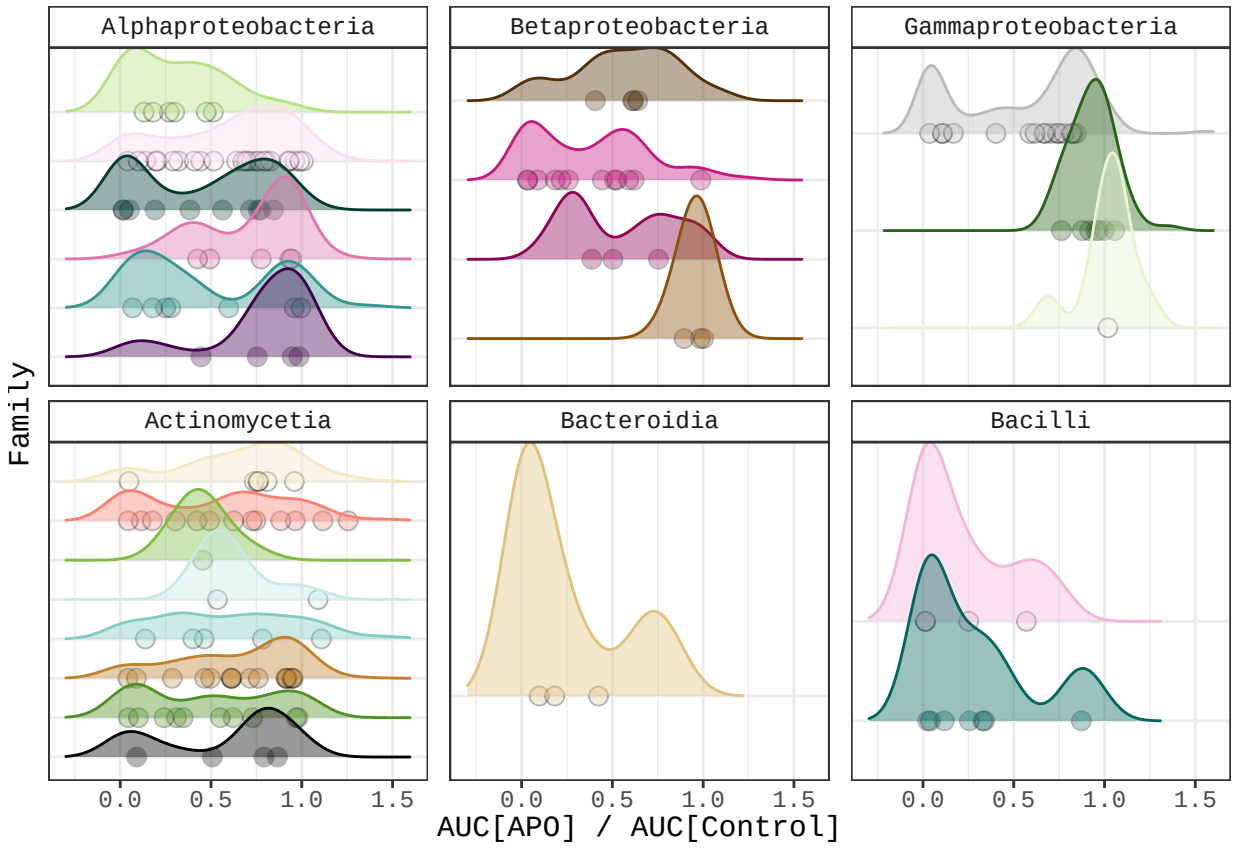

Alphaproteobacteria
Betaproteobacteria
Gammaproteobacteria
Actinomycetia
Bacteroidia
Bacilli
Family
0.0
0.5
1.0
1.5
AUC[APO] / AUC[Control]

Figure 2

Data in

Read the sequence tables etc (see *Reads_in.Rmd*)

Random

```
# Read sequence table
syncomRandom_noChim    <-  readRDS("files_publication/syncom_full_random_seqtable_nochim.rds")
syncomRandom_tax       <-  dada2::assignTaxonomy(seqs = syncomRandom_noChim,
                                    "files_publication/syncom_full_random_16SrRNA.fasta",
                                     taxLevels = c("Kingdom", "Phylum", "Family", "Genus", "Strain"))
```

```
## Warning in dada2::assignTaxonomy(seqs = syncomRandom_noChim, "files_publication/
## syncom_full_random_16SrRNA.fasta", : Some reference sequences were too short
## (<20nts) and were excluded.
```

```
syncomRandom_metadata  <-  readRDS("files_publication/Metadata_publication.rds") %>%
                              filter(Syncom == "Full_Random")

syncomRandom_strains <-  read_rds("files_publication/syncom_full_random.rds") %>%
  mutate(SynCom = "Random")
```

```
phyl_random <- phyloseq(otu_table(syncomRandom_noChim, taxa_are_rows =F),
                  sample_data(syncomRandom_metadata),
                  tax_table(syncomRandom_tax),
                  phy_tree(read_rds("files_publication/syncom_full_random_GTR_tree.rds")$tree))

phyl_random_glom_strain <- phyl_random %>%
  subset_samples(Rep != "Rep_1") %>%
  tax_glom("Strain")
phyl_random_rel_strain <- transform_sample_counts(phyl_random_glom_strain, function(x) x / sum(x) )
```

Tolerant

```
# Read sequence table
syncomTolerant_noChim   <-  readRDS("files_publication/syncom_reduced_tolerant_seqtable_nochim.rds")

syncomTolerant_tax      <- dada2::assignTaxonomy(seqs = syncomTolerant_noChim,
                                    "files_publication/syncom_reduced_tolerant_16SrRNA.fasta",
                                     taxLevels = c("Kingdom", "Phylum", "Family", "Genus", "Strain"))
```

```
## Warning in dada2::assignTaxonomy(seqs = syncomTolerant_noChim,
## "files_publication/syncom_reduced_tolerant_16SrRNA.fasta", : Some reference
## sequences were too short (<20nts) and were excluded.
```

```
syncomTolerant_metadata <-  readRDS("files_publication/Metadata_publication.rds") %>%
                          filter(Syncom == "Reduced_Tolerant")

syncomTolerant_strains <-  read_rds("files_publication/syncom_reduced_tolerant.rds") %>%
  mutate(SynCom = "Tolerant")
```

```
phyl_tolerant <- phyloseq(otu_table(syncomTolerant_noChim, taxa_are_rows =F),
                  sample_data(syncomTolerant_metadata),
                  tax_table(syncomTolerant_tax),
```

```
                  phy_tree(read_rds("files_publication/syncom_reduced_tolerant_GTR_tree.rds")$tree))

phyl_tolerant_glom_strain <- phyl_tolerant %>%
  subset_samples(Rep != "Rep_1") %>%
  tax_glom("Strain")
phyl_tolerant_rel_strain <- transform_sample_counts(phyl_tolerant_glom_strain, function(x) x / sum(x) )
```

Mixed

```
# Read sequence table
syncomMixed_noChim <-  readRDS("files_publication/syncom_reduced_mixed_seqtable_nochim.rds")
syncomMixed_tax <- dada2::assignTaxonomy(seqs = syncomTolerant_noChim,
                                         "files_publication/syncom_reduced_mixed_16SrRNA.fasta",
                                          taxLevels = c("Kingdom", "Phylum", "Family", "Genus", "Strain"))
syncomMixed_metadata        <-  readRDS("files_publication/Metadata_publication.rds") %>%

syncomMixed_strains <-  read_rds("files_publication/syncom_reduced_mixed.rds") %>%
  mutate(SynCom = "Mixed")
```

```
phyl_mixed <- phyloseq(otu_table(syncomMixed_noChim, taxa_are_rows =F),
                  sample_data(syncomMixed_metadata),
                  tax_table(syncomMixed_tax),
                  phy_tree(read_rds("files_publication/syncom_reduced_mixed_GTR_tree.rds")$tree))

phyl_mixed_glom_strain <- phyl_mixed %>%
  subset_samples(Rep != "Rep_1") %>%
  tax_glom("Strain")
phyl_mixed_rel_strain <- transform_sample_counts(phyl_mixed_glom_strain, function(x) x / sum(x) )
```

Sensitive

```
# Read sequence table
syncomSensitive_noChim <-  readRDS("files_publication/syncom_reduced_sensitive_seqtable_nochim.rds")
syncomSensitive_tax <- dada2::assignTaxonomy(seqs = syncomTolerant_noChim,
                                         "files_publication/syncom_reduced_sensitive_16SrRNA.fasta",
                                          taxLevels = c("Kingdom", "Phylum", "Family", "Genus", "Strain"))
syncomSensitive_metadata        <-  readRDS("files_publication/Metadata_publication.rds") %>%

syncomSensitive_strains <-  read_rds("files_publication/syncom_reduced_sensitive.rds") %>%
  mutate(SynCom = "Tolerant")
```

```
phyl_sensitive <- phyloseq(otu_table(syncomSensitive_noChim, taxa_are_rows =F),
                  sample_data(syncomSensitive_metadata),
                  tax_table(syncomSensitive_tax),
                  phy_tree(read_rds("files_publication/syncom_reduced_sensitive_GTR_tree.rds")$tree))

phyl_sensitive_glom_strain <- phyl_sensitive %>%
  subset_samples(Rep != "Rep_1") %>%
  tax_glom("Strain")
phyl_sensitive_rel_strain <- transform_sample_counts(phyl_sensitive_glom_strain, function(x) x / sum(x)
```

Plot

Individual panels

```
cca_jaccard_random_rel_strain <-  phyl_random_rel_strain %>%
  ordinate("CCA", "jaccard", formula = ~Treatment+Timepoint)
cca_jaccard_random_plot<- plot_ordination(phyl_random_rel_strain,
                cca_jaccard_random_rel_strain,
                color = "Treatment",
                axes = c(1,2)) +
  ggforce::geom_mark_ellipse(fill = "grey90") +
  geom_point() +
  facet_grid(~Timepoint) +
  geom_point(size = 3) +
  geom_point(size = 3, pch = 21, color = "black") +
  ggtitle(paste("Random Syncom: CCA")) +
  ggthemes::scale_color_few() +
  theme_bw() +
  theme(text = element_text(family = "NimbusMon"),
        strip.text = element_text(family = "NimbusMon",face = "bold"),
        strip.background = element_rect(fill = "white"))

cca_jaccard_tolerant_rel_strain <-  phyl_tolerant_rel_strain %>%
  ordinate("CCA", "jaccard", formula = ~Treatment+Timepoint)
cca_jaccard_tolerant_plot <- plot_ordination(phyl_tolerant_rel_strain,
                cca_jaccard_tolerant_rel_strain,
                color = "Treatment",
                axes = c(1,2)) +
  ggforce::geom_mark_ellipse(fill = "grey90") +
  geom_point() +
  facet_grid(~Timepoint) +
  geom_point(size = 3) +
  geom_point(size = 3, pch = 21, color = "black") +
  ggtitle(paste("Tolerant Syncom: CCA")) +
  ggthemes::scale_color_few() +
  theme_bw() +
  theme(text = element_text(family = "NimbusMon"),
        strip.text = element_text(family = "NimbusMon",face = "bold"),
        strip.background = element_rect(fill = "white"))

cca_jaccard_mixed_rel_strain <-  phyl_mixed_rel_strain %>%
  ordinate("CCA", "jaccard", formula = ~Treatment+Timepoint)
cca_jaccard_mixed_plot <- plot_ordination(phyl_mixed_rel_strain,
                cca_jaccard_mixed_rel_strain,
                color = "Treatment",
                axes = c(1,2)) +
  ggforce::geom_mark_ellipse(fill = "grey90") +
  geom_point() +
  facet_grid(~Timepoint) +
  geom_point(size = 3) +
  geom_point(size = 3, pch = 21, color = "black") +
  ggtitle(paste("Mixed Syncom: CCA")) +
  ggthemes::scale_color_few() +
  theme_bw() +
```

```
  theme(text = element_text(family = "NimbusMon"),
        strip.text = element_text(family = "NimbusMon",face = "bold"),
        strip.background = element_rect(fill = "white"))

cca_jaccard_sensitive_rel_strain <-  phyl_sensitive_rel_strain %>%
  ordinate("CCA", "jaccard", formula = ~Treatment+Timepoint)
cca_jaccard_sensitve_plot <- plot_ordination(phyl_sensitive_rel_strain,
                cca_jaccard_sensitive_rel_strain,
                color = "Treatment",
                axes = c(1,2)) +
  ggforce::geom_mark_ellipse(fill = "grey90") +
  geom_point() +
  facet_grid(~Timepoint) +
  geom_point(size = 3) +
  geom_point(size = 3, pch = 21, color = "black") +
  ggtitle(paste("Sensitive Syncom: CCA")) +
  ggthemes::scale_color_few() +
  theme_bw() +
  theme(text = element_text(family = "NimbusMon"),
        strip.text = element_text(family = "NimbusMon",face = "bold"),
        strip.background = element_rect(fill = "white"))
```

Figure

```
(cca_jaccard_random_plot | cca_jaccard_tolerant_plot) /
(cca_jaccard_mixed_plot | cca_jaccard_sensitve_plot) +
  plot_layout(guides = "collect",
              heights = c(1,1)) +
  plot_annotation(tag_levels = "A", caption = "Formula: Treatment + Timepoint") &
  theme(text=element_text(family = "NimbusMon"), strip.background = element_rect(fill = "white"))
```

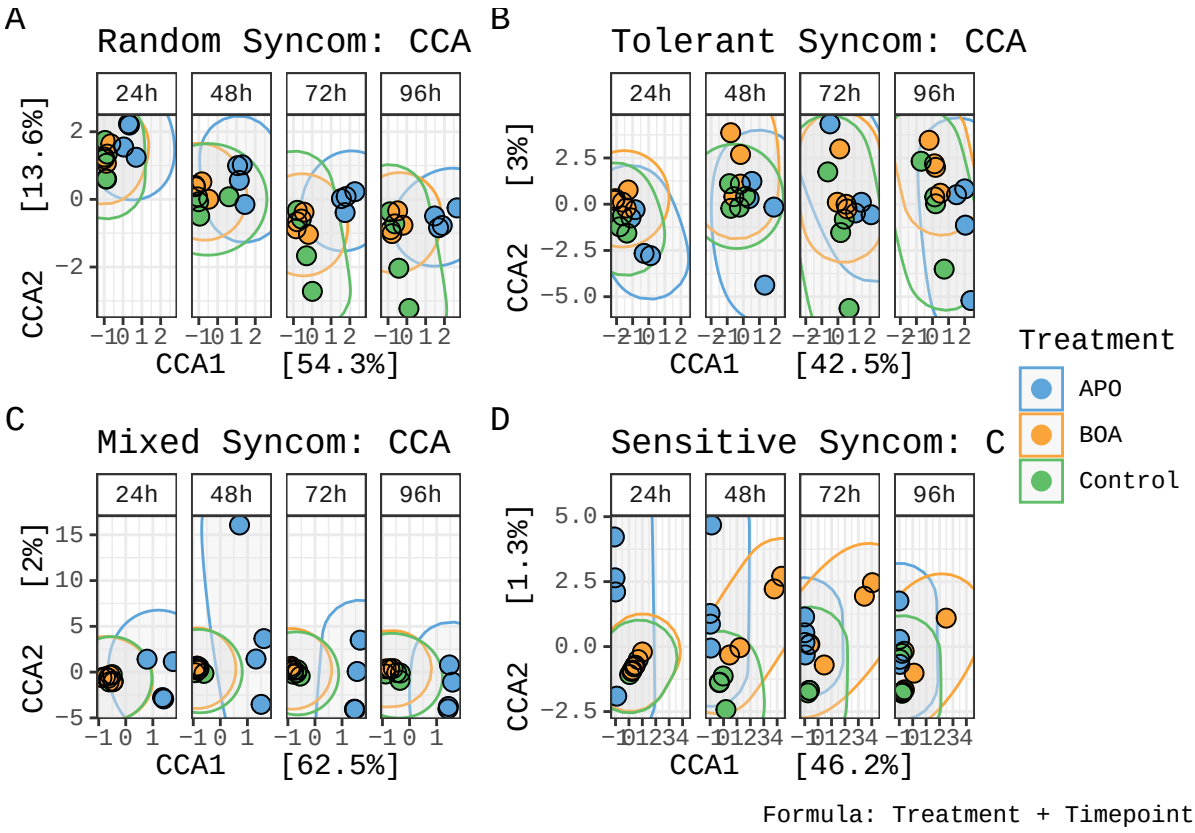

A
Random Syncom: CCA
B
Tolerant Syncom: CCA
C
Mixed Syncom: CCA
D
Sensitive Syncom: C
24h
48h
72h
96h
CCA1 [54.3%]
CCA2 [13.6%]
CCA1 [42.5%]
CCA2 [3%]
CCA1 [62.5%]
CCA2 [2%]
CCA1 [46.2%]
CCA2 [1.3%]
Treatment
APO
BOA
Control
Formula: Treatment + Timepoint

Figure 3

Calculate l2fc

```
all_lf2cs_no_time <- bind_rows(list(
phyl_mixed_glom_strain %>%
  compare_conditions(treatvar = "Treatment",
                     treat1 = "APO",
                     treat2 = "Control",
                     alpha = 0.05,
                     OTUcol = "Strain",
                     plot = F,
                     return_unfiltered = T) %>%
  mutate(Syncom = "Mixed", Treatment = "APO"),
phyl_mixed_glom_strain %>%
  compare_conditions(treatvar = "Treatment",
                     treat1 = "BOA",
                     treat2 = "Control",
                     alpha = 0.05,
                     OTUcol = "Strain",
                     plot = F,
                     return_unfiltered = T) %>%
  mutate(Syncom = "Mixed", Treatment = "BOA"),
phyl_sensitive_glom_strain %>%
  compare_conditions(treatvar = "Treatment",
                     treat1 = "APO",
                     treat2 = "Control",
                     alpha = 0.05,
                     OTUcol = "Strain",
                     plot = F,
                     return_unfiltered = T) %>%
  mutate(Syncom = "Sensitive", Treatment = "APO"),
phyl_sensitive_glom_strain %>%
  compare_conditions(treatvar = "Treatment",
                     treat1 = "BOA",
                     treat2 = "Control",
                     alpha = 0.05,
                     OTUcol = "Strain",
                     plot = F,
                     return_unfiltered = T) %>%
  mutate(Syncom = "Sensitive", Treatment = "BOA"),
phyl_random_glom_strain %>%
  compare_conditions(treatvar = "Treatment",
                     treat1 = "APO",
                     treat2 = "Control",
                     alpha = 0.05,
                     OTUcol = "Strain",
                     plot = F,
                     return_unfiltered = T) %>%
  mutate(Syncom = "Random", Treatment = "APO"),
phyl_random_glom_strain %>%
  compare_conditions(treatvar = "Treatment",
                     treat1 = "BOA",
                     treat2 = "Control",
```

```
                    alpha = 0.05,
                    OTUcol = "Strain",
                    plot = F,
                    return_unfiltered = T) %>%
  mutate(Syncom = "Random", Treatment = "BOA"),
phyl_tolerant_glom_strain %>%
  compare_conditions(treatvar = "Treatment",
                    treat1 = "APO",
                    treat2 = "Control",
                    alpha = 0.05,
                    OTUcol = "Strain",
                    plot = F,
                    return_unfiltered = T) %>%
  mutate(Syncom = "Tolerant", Treatment = "APO"),
phyl_tolerant_glom_strain %>%
  compare_conditions(treatvar = "Treatment",
                    treat1 = "BOA",
                    treat2 = "Control",
                    alpha = 0.05,
                    OTUcol = "Strain",
                    plot = F,
                    return_unfiltered = T) %>%
  mutate(Syncom = "Tolerant", Treatment = "BOA")))
```

```
## converting counts to integer mode
```

```
## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in
## design formula are characters, converting to factors
```

```
## estimating size factors
```

```
## estimating dispersions
```

```
## gene-wise dispersion estimates
```

```
## mean-dispersion relationship
```

```
## final dispersion estimates
```

```
## fitting model and testing
```

```
## -- replacing outliers and refitting for 3 genes
## -- DESeq argument 'minReplicatesForReplace' = 7
## -- original counts are preserved in counts(dds)
```

```
## estimating dispersions
```

```
## fitting model and testing
```

```
## converting counts to integer mode
```

```
## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in
## design formula are characters, converting to factors
```

```
## estimating size factors
```

```
## estimating dispersions
```

```
## gene-wise dispersion estimates
```

```
## mean-dispersion relationship
```

```
## final dispersion estimates

## fitting model and testing

## -- replacing outliers and refitting for 3 genes
## -- DESeq argument 'minReplicatesForReplace' = 7
## -- original counts are preserved in counts(dds)

## estimating dispersions

## fitting model and testing

## converting counts to integer mode

## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in
## design formula are characters, converting to factors

## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## -- note: fitType='parametric', but the dispersion trend was not well captured by the
##    function: y = a/x + b, and a local regression fit was automatically substituted.
##    specify fitType='local' or 'mean' to avoid this message next time.

## Warning in lfproc(x, y, weights = weights, cens = cens, base = base, geth =
## geth, : Estimated rdf < 1.0; not estimating variance

## final dispersion estimates

## fitting model and testing

## -- replacing outliers and refitting for 3 genes
## -- DESeq argument 'minReplicatesForReplace' = 7
## -- original counts are preserved in counts(dds)

## estimating dispersions

## fitting model and testing

## converting counts to integer mode

## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in
## design formula are characters, converting to factors

## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## -- note: fitType='parametric', but the dispersion trend was not well captured by the
##    function: y = a/x + b, and a local regression fit was automatically substituted.
##    specify fitType='local' or 'mean' to avoid this message next time.

## Warning in lfproc(x, y, weights = weights, cens = cens, base = base, geth =
## geth, : Estimated rdf < 1.0; not estimating variance

## final dispersion estimates

## fitting model and testing
```

```
## -- replacing outliers and refitting for 3 genes
## -- DESeq argument 'minReplicatesForReplace' = 7
## -- original counts are preserved in counts(dds)

## estimating dispersions

## fitting model and testing

## converting counts to integer mode

## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in
## design formula are characters, converting to factors

## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing

## -- replacing outliers and refitting for 2 genes
## -- DESeq argument 'minReplicatesForReplace' = 7
## -- original counts are preserved in counts(dds)

## estimating dispersions

## fitting model and testing

## converting counts to integer mode

## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in
## design formula are characters, converting to factors

## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing

## -- replacing outliers and refitting for 2 genes
## -- DESeq argument 'minReplicatesForReplace' = 7
## -- original counts are preserved in counts(dds)

## estimating dispersions

## fitting model and testing

## converting counts to integer mode

## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in
## design formula are characters, converting to factors

## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates
```

```
## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing

## -- replacing outliers and refitting for 2 genes
## -- DESeq argument 'minReplicatesForReplace' = 7
## -- original counts are preserved in counts(dds)

## estimating dispersions

## fitting model and testing

## converting counts to integer mode

## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in
## design formula are characters, converting to factors

## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing

## -- replacing outliers and refitting for 2 genes
## -- DESeq argument 'minReplicatesForReplace' = 7
## -- original counts are preserved in counts(dds)

## estimating dispersions

## fitting model and testing
```

```
strain_AUC <- AUC_emmeans %>%
  filter(Treat.x %in% c("Control","50uM_APO","100uM_BOA")) %>%
  mutate(Treatment = case_when(Treat.x == "Control" ~ Treat.x,
                               Treat.x == "50uM_APO" ~ "APO",
                               Treat.x == "100uM_BOA" ~ "BOA")) %>%
  dplyr::select(Strain, Genus ,Treatment, AUC, estimate) %>%
  group_by(Strain,Treatment) %>%
  summarize(mean_AUC = mean(AUC, na.rm = T),
            mean_rel_AUC = mean(estimate, na.rm = T)) %>%
  ungroup
```

```
## `summarise()` regrouping output by 'Strain' (override with `.groups` argument)
```

Plot l2FC community vs AUC ratio

```
growth_data_l2fc <- left_join(strain_AUC, all_lf2cs_no_time,  by = c("Strain", "Treatment")) %>%
  filter(!is.na(Treatment),
         !is.na(Syncom),
         !is.na(log2FoldChange),
         Treatment != "Control")
```

Figure 3A - Plot

```
growth_data_l2fc %>%
  ggplot(aes(x = mean_rel_AUC,
             y = log2FoldChange)) +
  geom_vline(aes(xintercept = 1), color = "darkgrey") +
  geom_hline(aes(yintercept = 0), color = "darkgrey") +
  geom_smooth(method = "lm", color = "black", se = T) +
  geom_point(colour = "black",
             size = 5,
             alpha = 0.7) +
  geom_point(aes(color = Syncom),
             size = 5,
             alpha = 0.7) +
  scale_color_manual(values = syncom_colors) +
  facet_wrap(~Treatment, scales = "free_x") +
  theme_bw(base_family = "NimbusMon") +
  xlab(paste("Effect on strain grown in isolation \n [Ratio area under the curve (Treatment / Control)]
  ylab(paste("Effect on strain grown in community \n [log2Foldchange (Treatment / Control)]")) +
  ggtitle("Growth and abundance changes by treatment") +
  theme(plot.title = element_text(family = "NimbusMon"),
        plot.subtitle =  element_text(family = "NimbusMon"),
        text = element_text(family = "NimbusMon"),
        strip.background = element_blank(),
        axis.title = element_text(family = "NimbusMon"),
        strip.text = element_text(family = "NimbusMon",
                                  face = "bold"),
        legend.position = "none")
```

```
## `geom_smooth()` using formula 'y ~ x'
```

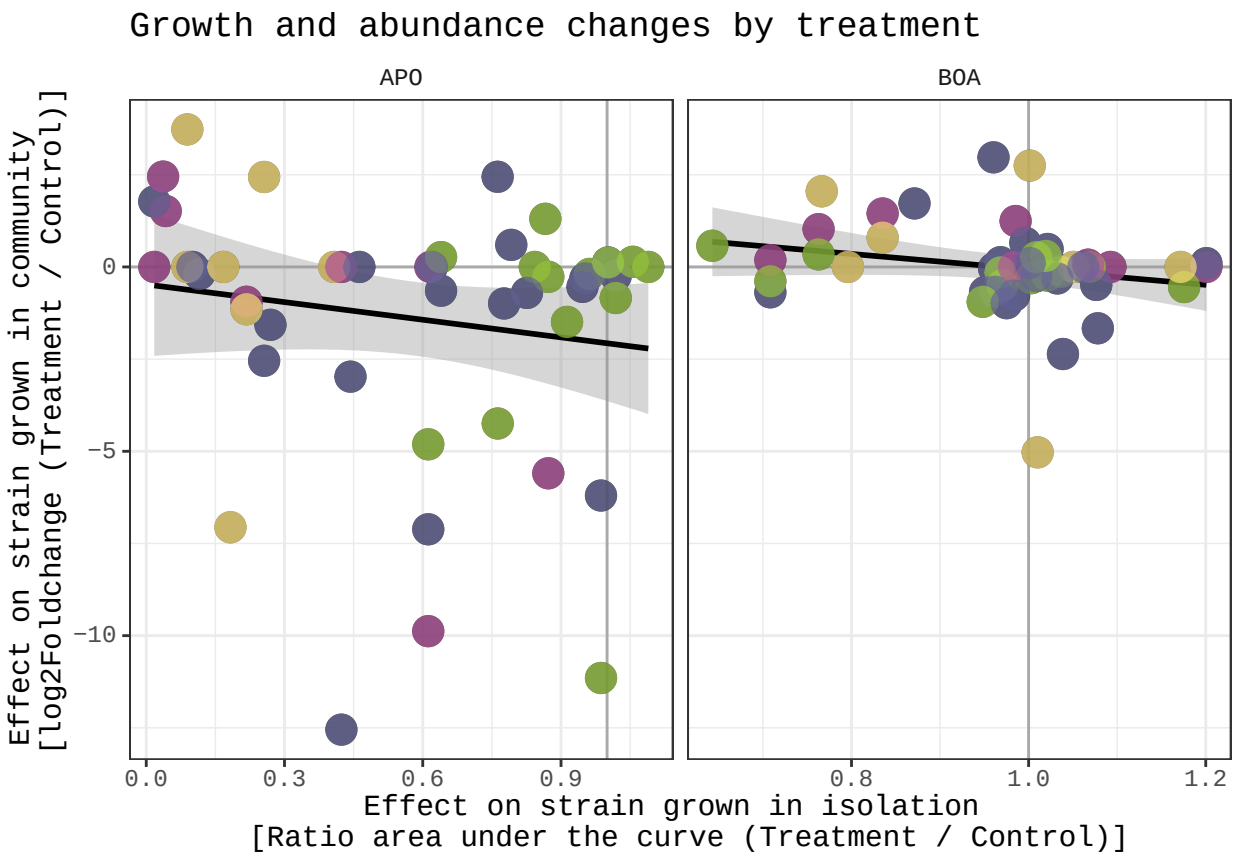


Figure 3B - Plot

```
all_lf2cs_no_time %>%
  left_join(strain_AUC, by = c("Strain", "Treatment")) %>%
  mutate(genus_strain = str_c(Genus,Strain, sep = "\n")) %>%
  group_by(Strain) %>%
  add_count(Treatment) %>%
  filter(n > 1) %>%
  ggplot(aes( y=log2FoldChange, x = Treatment, group = Syncom)) +
  facet_wrap(~genus_strain, ncol = 5, scales = "free_x") +
  geom_hline(aes(yintercept = 0),
             linetype = "dashed",
             color = "darkgrey" ) +
  geom_errorbar(aes(ymin = log2FoldChange-lfcSE,
                    ymax = log2FoldChange + lfcSE),
                position = position_dodge(width = 0.6),
                width = 0.2) +
  geom_point(size = 4,
             position = position_dodge(width = 0.6)) +
  geom_point(aes(color = Syncom),
             size = 3,
             position = position_dodge(width = 0.6)) +
  geom_vline(aes(xintercept = 1.5)) +
  scale_x_discrete(limits = rev) +
  coord_flip() +
  theme_bw() +
```

```
  theme(text = element_text(family = "NimbusMon"),
        strip.background = element_rect(fill = "white"),
        legend.position = "none") +
  scale_color_manual(values = syncom_colors) +
  labs(title = "Changes in abundance",
       sub = "Strains included in more than one community",
       y = "log2FC vs Control (across all timepoints)",
       x = "Treatment")
```

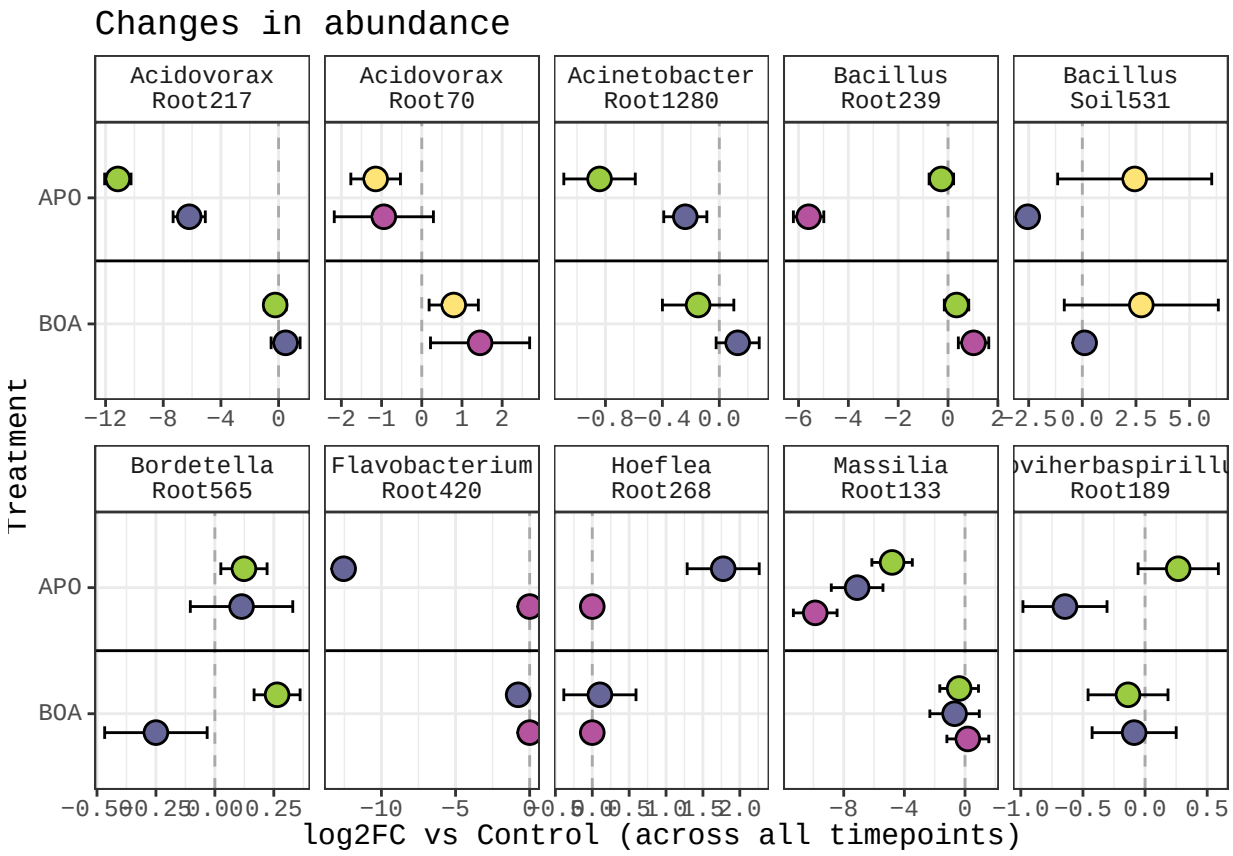


Figure 4

Figure 4A

Abundance table

```
abundances_tol_rand <- bind_rows(list(phyl_tolerant_rel_strain %>%
  psmelt() %>%
  dplyr::select(Genus, Strain, Treatment, Timepoint, Abundance) %>%
  mutate(Syncom = "Tolerant"),
  phyl_random_rel_strain %>%
  psmelt() %>%
  dplyr::select(Genus, Strain, Treatment, Timepoint, Abundance) %>%
  mutate(Syncom = "Random")))
abundances_tol_rand_summarized <- abundances_tol_rand %>%
  group_by(Syncom, Genus, Strain, Treatment) %>%
  summarize(mean = mean(Abundance, na.rm = T),
            sd = sd(Abundance, na.rm = T)) %>%
  ungroup
```

```
## `summarise()` regrouping output by 'Syncom', 'Genus', 'Strain' (override with `.groups` argument)
```

Overlap Random and Tolerant

```
tol_strains <- phyl_tolerant_rel_strain %>%
  psmelt() %>% .$Strain
rand_strains <- phyl_random_rel_strain %>%
  psmelt() %>% .$Strain
rand_tol_genusstrain <- rbind(phyl_tolerant_rel_strain %>%
  psmelt() %>%
  mutate(Genus_Strain = paste(Genus, Strain, sep = "\n")) %>%
  dplyr::select(Strain, Genus_Strain),
  phyl_random_rel_strain %>%
  psmelt() %>%
  mutate(Genus_Strain = paste(Genus, Strain, sep = "\n")) %>%
  dplyr::select(Strain, Genus_Strain)) %>%
  distinct()
tolerant_strains_genus <- phyl_tolerant_rel_strain %>%
  psmelt() %>%
  dplyr::select(Genus, Strain) %>%
  mutate(Syncom = "Tolerant") %>%
  distinct()
random_strains_genus <- phyl_random_rel_strain %>%
  psmelt() %>%
  dplyr::select(Genus, Strain) %>%
  mutate(Syncom = "Random") %>%
  distinct()
```

Figure 4A- Plot

```
genus_abundance_plot_random_tolerant <- abundances_tol_rand %>%
  mutate(Strain_Identity = case_when(Strain %in% intersect(tol_strains, rand_strains) ~ "Same strain",
                                     Strain %in% tol_strains ~ "Different strains",
                                     Strain %in% rand_strains ~ "Different strains",
```

```
                                TRUE ~"Neither")) %>%
  filter(Genus %in% (abundances_tol_rand_summarized %>%
                       filter(mean >= 0.01) %$%
                       Genus),
         !Genus %in% c("Rhizobium", "Hoeflea", "Flavobacterium")
         ) %>%
  ggplot(aes(y = Abundance,
             x = Syncom,
             color = fct_relevel(Treatment,"Control","BOA","APO"),
             group = fct_relevel(Treatment,"Control","BOA","APO"),
             shape = fct_relevel(Strain_Identity, "Same strain") )) +
  stat_summary(geom = "point", fun = "mean", position = position_dodge(width = 0.2), size = 4) +
  stat_summary(geom = "errorbar",
               fun.data = "mean_cl_boot",
               width = 0.2,
               position = position_dodge(width = 0.2),
               color = "black") +
  facet_grid(~Genus) +
  theme_bw() +
  theme(legend.position = "bottom",
        text = element_text(family = "NimbusMon"),
        strip.background = element_rect(fill = "white"),
        #strip.text.y = element_blank(),
        axis.text.x = element_text(angle = 30, hjust = 1),
        axis.title.x = element_blank()) +
  scale_shape_discrete(name = "Strain Identity") +
  scale_color_manual(name =" Treatment",values = rev(ggthemes::few_pal(palette = "Medium")(3)))
genus_abundance_plot_random_tolerant
```

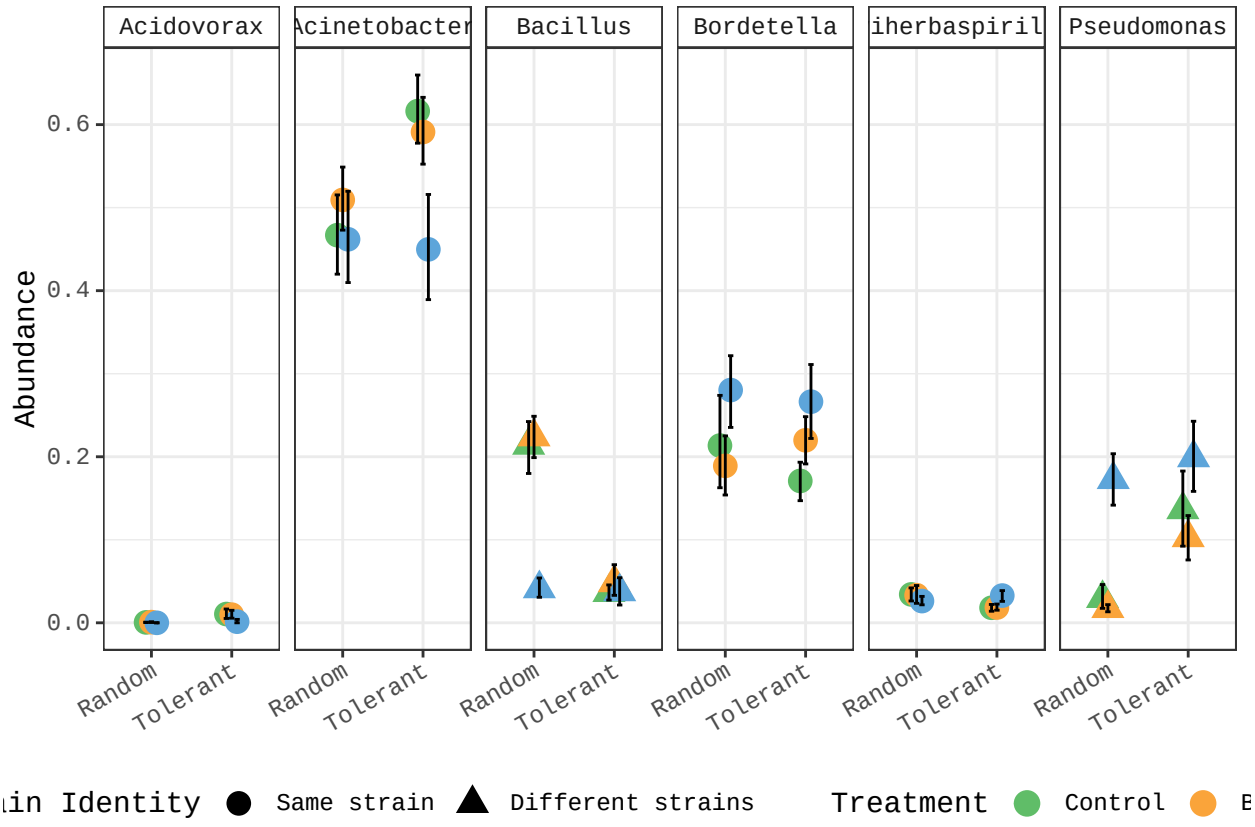


Figure 4B

Figure 4B

Partial Correlations

```
corr_meth <- "pearson"
adjust <- "fdr"

tol_apo_others_partial <- bind_rows(list(
  phyl_tolerant_glom_strain %>%
  phyloseq_corrs("Treatment",
                 "APO",
                 p.adj = adjust,
                 mincounts = 100,
                 method = corr_meth,
                 OTUcol = "Genus",
                 partial = TRUE) %>%
    mutate(Treatment = "APO",
           Syncom = "Tolerant"),
phyl_tolerant_glom_strain %>%
  phyloseq_corrs("Treatment",
                 c("BOA", "DMSO"),
                 p.adj = adjust,
                 mincounts = 100,
                 method = corr_meth,
                 OTUcol = "Genus",
                 partial = TRUE) %>%
```

```
  mutate(Treatment = "no APO",
         Syncom = "Tolerant")))
```

```
## converting counts to integer mode
```

```
## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in
## design formula are characters, converting to factors
```

```
## estimating size factors
```

```
## estimating dispersions
```

```
## gene-wise dispersion estimates
```

```
## mean-dispersion relationship
```

```
## final dispersion estimates
```

```
## fitting model and testing
```

```
## -- replacing outliers and refitting for 2 genes
## -- DESeq argument 'minReplicatesForReplace' = 7
## -- original counts are preserved in counts(dds)
```

```
## estimating dispersions
```

```
## fitting model and testing
```

```
## converting counts to integer mode
```

```
## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in
## design formula are characters, converting to factors
```

```
## estimating size factors
```

```
## estimating dispersions
```

```
## gene-wise dispersion estimates
```

```
## mean-dispersion relationship
```

```
## final dispersion estimates
```

```
## fitting model and testing
```

```
## -- replacing outliers and refitting for 2 genes
## -- DESeq argument 'minReplicatesForReplace' = 7
## -- original counts are preserved in counts(dds)
```

```
## estimating dispersions
```

```
## fitting model and testing
```

```
rand_apo_others_partial <- bind_rows(list(
  phyl_random_glom_strain %>%
  phyloseq_corrs("Treatment",
                 "APO",
                 p.adj = adjust,
                 mincounts = 100,
                 method = corr_meth,
                 OTUcol = "Genus",
                 partial = TRUE) %>%
    mutate(Treatment = "APO",
           Syncom = "Random") ,
  phyl_random_glom_strain %>%
```

```
  phyloseq_corrs("Treatment",
                 c("BOA", "DMSO"),
                 p.adj = adjust,
                 mincounts = 100,
                 method = corr_meth,
                 OTUcol = "Genus",
                 partial = TRUE) %>%
  mutate(Treatment = "no APO",
         Syncom = "Random")))
```

```
## converting counts to integer mode

## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in
## design formula are characters, converting to factors

## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing

## -- replacing outliers and refitting for 2 genes
## -- DESeq argument 'minReplicatesForReplace' = 7
## -- original counts are preserved in counts(dds)

## estimating dispersions

## fitting model and testing

## converting counts to integer mode

## Warning in DESeqDataSet(se, design = design, ignoreRank): some variables in
## design formula are characters, converting to factors

## estimating size factors

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

## final dispersion estimates

## fitting model and testing

## -- replacing outliers and refitting for 2 genes
## -- DESeq argument 'minReplicatesForReplace' = 7
## -- original counts are preserved in counts(dds)

## estimating dispersions

## fitting model and testing
```

```
tol_rand_apo_others_partial <- rbind(tol_apo_others_partial, rand_apo_others_partial)
tol_rand_apo_others_p_partial <- tol_rand_apo_others_partial %>%
  filter(p < 0.1)
```

Node information

```
abundances_tol_rand_summarized_apo_others <- abundances_tol_rand %>%
  mutate(Treatment = case_when(Treatment == "APO" ~ Treatment,
                               TRUE ~"no APO")) %>%
  group_by(Syncom, Genus, Strain, Treatment) %>%
  summarize(mean = mean(Abundance, na.rm = T),
            sd = sd(Abundance, na.rm = T)) %>%
  ungroup
```

```
## `summarise()` regrouping output by 'Syncom', 'Genus', 'Strain' (override with `.groups` argument)
```

Figure 4B - Plot

```
set.seed(5)
plot_partial_corrs_dat_apo_others <- tol_rand_apo_others_p_partial %>%
  igraph::graph_from_data_frame() %>%
  ggnetwork::ggnetwork(layout = igraph::with_fr(),
                       arrow.gap = F) %>%
  distinct() %>%
  fix_node_placement_no_time(Treatments = list("no APO","APO")) %>%
  mutate(Genus = name) %>%
  left_join(rbind(tolerant_strains_genus, random_strains_genus),
            by = c("Syncom","Genus")) %>%
  mutate(presence = case_when(Strain %in% intersect(tol_strains, rand_strains) ~ "Both",
                              Strain %in% tol_strains ~ "Only one",
                              Strain %in% rand_strains ~ "Only one",
                              TRUE ~"Neither")) %>%
  left_join(abundances_tol_rand_summarized_apo_others, by = c("Syncom","Genus","Strain","Treatment"))
```

```
## Warning in format_fortify(model = model, nodes = nodes, weights = "none", :
## duplicated edges detected
```

```
plot_partial_corrs_apo_others <- plot_partial_corrs_dat_apo_others %>%
  mutate(x = case_when(x == 1 ~ 0.7, TRUE ~ x ),
           xend = case_when(xend == 1 ~ 0.7, TRUE ~ xend)) %>%
  ggplot(aes(x = x, y = y, xend = xend, yend = yend)) +
  geom_edges(aes(color = r), size = 1.5, curvature = 0.2, alpha = 0.9) +
  geom_nodes(aes(size = mean + sd), color = "black") +
  geom_nodes(aes(size = mean - sd), color = "white") +
  geom_nodelabel_repel(aes(label = paste(Genus, Strain, sep = "\n")),  seed = 342, size= 3) +
  geom_nodelabel_repel(aes(label = paste(Genus, Strain, sep = "\n"),  fill = presence), alpha = 0.3, se
 # ggtitle("Partial correlations") +
  scale_color_viridis_c(option = "C") +
  ggthemes::scale_fill_canva(palette = 'Corporate and sleek') +
  facet_grid(Treatment~Syncom,
             scales = "free"
             ) +
  theme_blank() +
  theme(text = element_text(family = "NimbusMon"),
        strip.background = element_rect(fill = "white"),
        panel.border = element_rect(fill = NA)) +
  # lims(x = c(-0.1,1.1),
  #      y = c(-0.1,1.1)) +
  NULL
```

```
plot_partial_corrs_apo_others
```

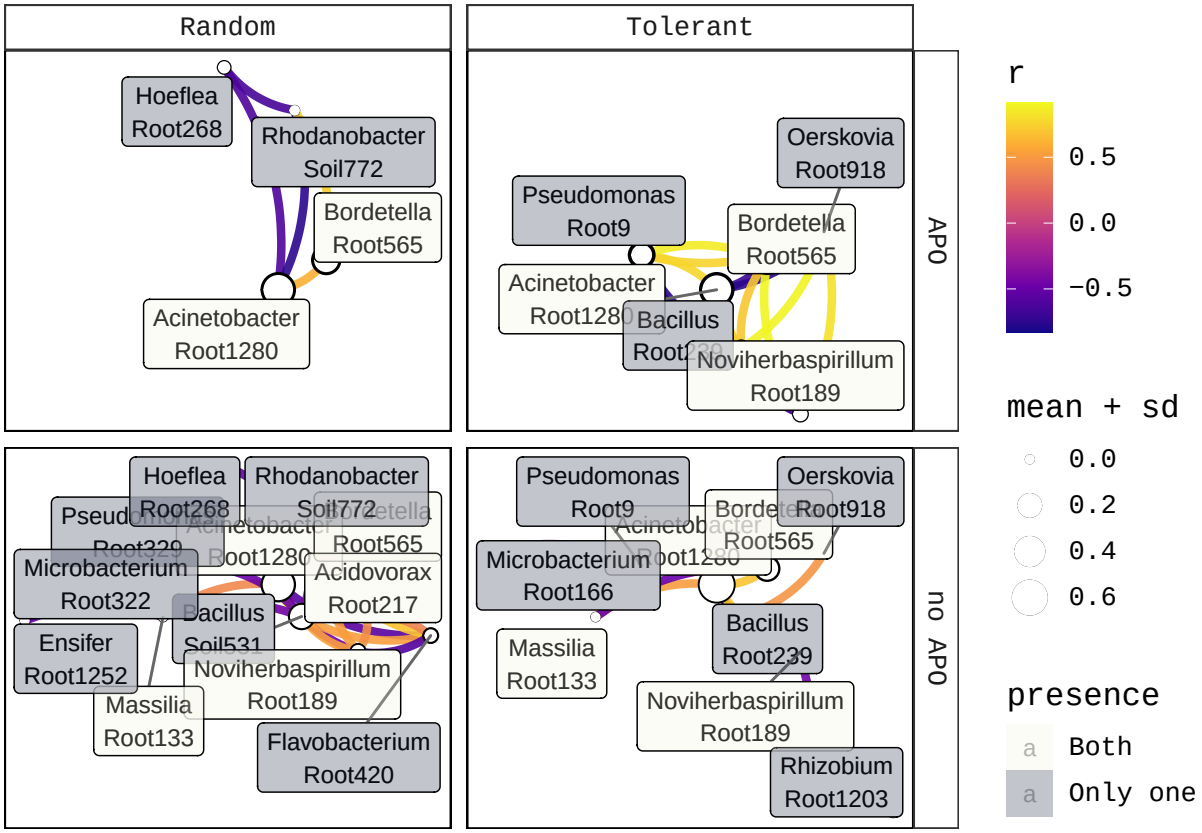
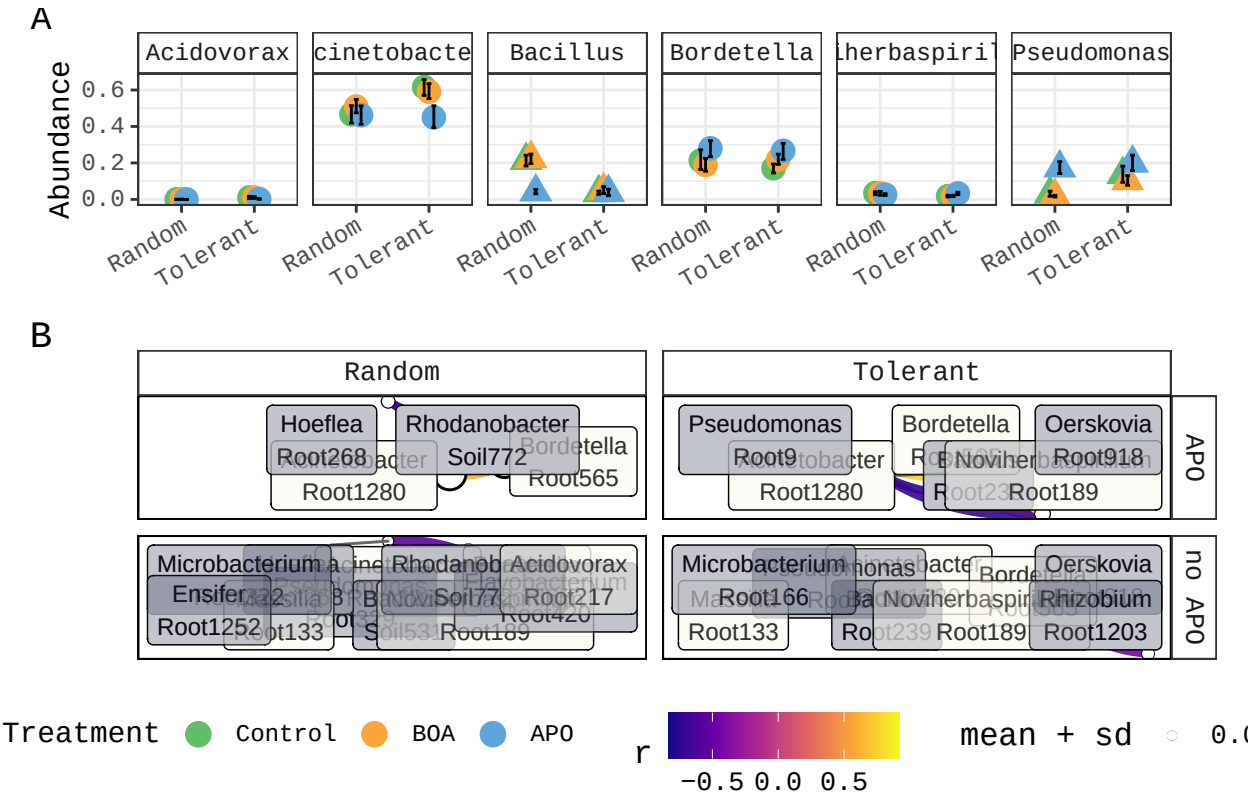


Figure 4 arranged

```
genus_abundance_plot_random_tolerant / plot_partial_corrs_apo_others +
  plot_layout(nrow = 2,heights = c(1,2), guides = "collect") +
  plot_annotation(tag_levels = 'A') &
  theme(legend.position = "bottom",
        text = element_text(family = "NimbusMon"))
```


A
Acidovorax
cinetobacte
Bacillus
Bordetella
herbaspiri
Pseudomonas
Abundance
0.6
0.4
0.2
0.0
Random
Tolerant
B
Random
Tolerant
Hoeflea
Root268
Root1280
Rhodanobacter
Soil772
Bordetella
Root565
Pseudomonas
Root9
Root1280
Bordetella
Oerskovia
Root918
Root189
APO
Microbacterium
Ensifer
Root1252
Rhodanob
Soil77
Acidovorax
Root217
Root420
Root189
Microbacterium
Root166
Root133
Oerskovia
Rhizobium
Root1203
no APO
Treatment
Control
BOA
APO
r
−0.5 0.0 0.5
mean + sd