**Codes for data analysis**

1. **Principal component analysis (PCA) of the metabolites detected in peanut rhizosphere soil.**

options(warn=-1)

library("ropls")

mymethod<-paste("/home/Pon/meta/metabolite/META\_DADABASE/R\_library", "PlotPLS2DScore.r",sep="/")

source(mymethod)

cdata <- read.csv("/home/Pon/meta/project/20220711-SNKY04262206300101-CY-Nontarget-70/data/exp\_all.hasqc.txt", head = TRUE, sep = "\t", row.names = 1,quote='\"', na.strings = 'NA')

csif <- read.csv("/home/Pon/meta/project/20220711-SNKY04262206300101-CY-Nontarget-70/data/exp\_all.hasqc.txt.tmp\_group.txt",header=T,sep="\t",comment.char = "",check.names = FALSE,row.names =1)

classFc <- as.factor(csif[,1])

cdata <- cdata[which(rowSums(cdata)!=0),]

gsamp <- rownames(csif)

samp <- colnames(cdata)

cdata <- cdata[,which(samp %in% gsamp)]

cdata <- t(cdata)

cdata <- cdata[rownames(csif),]

ci <- strsplit("0.95",";")[[1]]

permutation <- strsplit("200",";")[[1]]

mul\_type = strsplit("pca",";")[[1]]

scale = strsplit("UV",";")[[1]]

drawell = strsplit("group",";")[[1]]

log = strsplit("False",";")[[1]]

if(length(gsamp) <7){

crossvalI = length(gsamp)

}else{

crossvalI = 7 # package default

}

# for plsda ,oplsda result table

get\_pls\_result <- function(pls\_result, mytype, out\_prefix,confidence){

pls <- pls\_result

out\_prefix <- paste(out\_prefix,mytype,sep=".")

name.pls.x <- paste(out\_prefix,"sites.xls",sep=".")

name.pls.loading <- paste(out\_prefix,"loading.xls",sep=".")

name.pls.vip <- paste(out\_prefix,"vip.xls",sep=".")

name.pls.sum <- paste(out\_prefix,"model.xls",sep=".")

name.pls.permMN<- paste(out\_prefix,"permMN.xls",sep=".")

name.pls.intercept <- paste(out\_prefix,"intercept.xls",sep=".")

name.pls.ellipse <- paste(out\_prefix,"ellipse.xls",sep=".")

pls.perMN <- pls@suppLs$permMN[,c(2,3,7)]

pls.all\_model <- (pls@modelDF)[,1:6]

if(mytype=="PLS-DA"){

pls.x <- getScoreMN(pls)

pls.loading <- getLoadingMN(pls)

}else{

pls.p <- getScoreMN(pls)

pls.o <- pls@orthoScoreMN

pls.x <- cbind(pls.p,pls.o)

pls.pl <- getLoadingMN(pls)

pls.ol <- pls@orthoLoadingMN

pls.loading <- cbind(pls.pl,pls.ol)

}

#pls.loading <- getLoadingMN(pls)

pls.sum <- pls@modelDF

pls.vip <- getVipVn(pls)

pls.vip <- as.data.frame(pls.vip)

colnames(pls.vip) <- "VIP"

pls.z1 <- lm(pls.perMN[,1]~pls.perMN[,3])$coefficients[1]

pls.z2 <- lm(pls.perMN[,2]~pls.perMN[,3])$coefficients[1]

pls.z <- as.matrix(c(pls.z1,pls.z2))

ellipse.data <- add\_ellipse(pls,classFc,confidence)

write.table(pls.x,name.pls.x,sep="\t",quote=F,col.names=NA)

write.table(pls.loading,name.pls.loading,sep="\t",quote=F,col.names=NA)

write.table(pls.vip,name.pls.vip,sep="\t",quote=F,col.names=NA)

write.table(pls.sum,name.pls.sum,sep="\t",quote=F,col.names=NA)

write.table(pls.z,name.pls.intercept,sep="\t",quote=F)

write.table(ellipse.data,name.pls.ellipse,sep="\t",row.names=F,quote=F,col.names=T)

write.table(pls.perMN,name.pls.permMN,sep="\t",row.names=F,quote=F)

}

# from mul\_type get ci,perm,sacle value

get\_function\_var <- function(m\_type,myvar,is\_numeric=T){

if(is\_numeric){

result <- as.numeric(myvar[which(mul\_type == m\_type )])

}else{

result <- myvar[which(mul\_type == m\_type )]

}

return(result)

}

# from scale abbreviation to scale method

get\_scale <- function(abbreviation){

if(abbreviation == "UV"){

scale <- "standard"

}else if(abbreviation =="Ctr"){

scale <- "center"

}else if(abbreviation == "Par"){

scale <- "pareto"

}else{

scale <- "none"

}

return(scale)

}

if ("pca" %in% mul\_type){

confidence <- get\_function\_var("pca",ci)

trans <- get\_function\_var("pca",scale,is\_numeric=F)

ellipse <- get\_function\_var("pca",drawell,is\_numeric=F)

log <- get\_function\_var("pca",log,is\_numeric=F)

pca <- opls(cdata,printL=F,plotL=F,predI=NA,scaleC=get\_scale(trans),crossvalI=crossvalI,log10L=(log==as.character(TRUE)))

if(pca@summaryDF[["pre"]]=="1"){

pca <- opls(cdata,printL=F,plotL=F,predI=2,scaleC=get\_scale(trans),crossvalI=crossvalI,log10L=(log==as.character(TRUE)))

}

name.pca.x <- paste("ALL\_SAMPLE\_PCA\_HASQC","PCA.sites.xls",sep=".")

name.pca.loading <- paste("ALL\_SAMPLE\_PCA\_HASQC","PCA.loading.xls",sep=".")

name.pca.sum <- paste("ALL\_SAMPLE\_PCA\_HASQC","PCA.model.xls",sep=".")

name.pca.ellipse <- paste("ALL\_SAMPLE\_PCA\_HASQC","PCA.ellipse.xls",sep=".")

pca.x <- getScoreMN(pca)

pca.loading <- getLoadingMN(pca)

pca.sum <- pca@modelDF[1:2]

ellipse.data <- add\_ellipse(pca,classFc,confidence)

write.table(pca.x,name.pca.x,sep="\t",quote=F,col.names=NA)

write.table(pca.loading,name.pca.loading,sep="\t",quote=F,col.names=NA)

write.table(pca.sum,name.pca.sum,sep="\t",quote=F,col.names=NA)

write.table(ellipse.data,name.pca.ellipse,sep="\t",row.names=F,quote=F,col.names=T)

PlotPCA2DScore(pca,csif,"ALL\_SAMPLE\_PCA\_HASQC", width=NA, ellipse,confidence, show=as.numeric("1"), grey.scale=0)

}

if("plsda" %in% mul\_type){

confidence <- get\_function\_var("plsda",ci)

trans <- get\_function\_var("plsda",scale,is\_numeric=F)

log <- get\_function\_var("plsda",log,is\_numeric=F)

perm <- get\_function\_var("plsda",permutation)

ellipse <- get\_function\_var("plsda",drawell,is\_numeric=F)

plsda <- opls(cdata,classFc,printL=F,plotL=F,predI=2,scaleC=get\_scale(trans),permI=perm,crossvalI=crossvalI,log10L=(log==as.character(TRUE)))

if(plsda@modelDF$Signif.[2] == "NS" | plsda@modelDF$Signif.[2] == "N4"| plsda@modelDF$Signif.[1] == "NS"|plsda@modelDF$Signif.[1] == "N4"){

plsda <- opls(cdata,classFc,printL=F,plotL=F,predI=2,scaleC=get\_scale(trans),permI=perm,crossvalI=crossvalI,log10L=(log==as.character(TRUE)))

}else{

plsda <- opls(cdata,classFc,printL=F,plotL=F,predI=NA,scaleC=get\_scale(trans),permI=perm,crossvalI=crossvalI,log10L=(log==as.character(TRUE)))

}

get\_pls\_result(plsda,"PLS-DA","ALL\_SAMPLE\_PCA\_HASQC",confidence)

PlotPLS2DScore(plsda,csif,"ALL\_SAMPLE\_PCA\_HASQC", width=NA, ellipse, confidence, show=as.numeric("1"), grey.scale=0)

PlotModelPerm(plsda,"ALL\_SAMPLE\_PCA\_HASQC","PLS-DA")

}

if("oplsda" %in% mul\_type){

confidence <- get\_function\_var("oplsda",ci)

trans <- get\_function\_var("oplsda",scale,is\_numeric=F)

log <- get\_function\_var("oplsda",log,is\_numeric=F)

perm <- get\_function\_var("oplsda",permutation)

ellipse <- get\_function\_var("oplsda",drawell,is\_numeric=F)

oplsda <- opls(cdata, classFc, predI=1, orthoI=1,printL=F,plotL=F,scaleC=get\_scale(trans),permI=perm,crossvalI=crossvalI,log10L=(log==as.character(TRUE)))

if(oplsda@modelDF[[1,"Signif."]] !="NS" & oplsda@modelDF[[2,"Signif."]] !="NS" & oplsda@modelDF[[1,"Signif."]] !="N4" & oplsda@modelDF[[2,"Signif."]] !="N4" ){

oplsda <- opls(cdata,classFc,predI=1,orthoI=NA,printL=F,plotL=F,scaleC=get\_scale(trans),permI=perm,crossvalI=crossvalI,log10L=(log==as.character(TRUE)))

}

get\_pls\_result(oplsda,"OPLS-DA","ALL\_SAMPLE\_PCA\_HASQC",confidence)

PlotOPLS2DScore(oplsda,csif,"ALL\_SAMPLE\_PCA\_HASQC", width=NA, ellipse,confidence, show= as.numeric("1"), grey.scale=0)

PlotModelPerm(oplsda,"ALL\_SAMPLE\_PCA\_HASQC","OPLS-DA")

}

1. **Hierarchical clustering and Heatmap of peanut rhizosphere metabolites in the different cropping systems.**

library(ComplexHeatmap)

library(readxl)

data <- read\_xlsx("./70-SCORE-META.xlsx")

df <- as.data.frame(data[!is.na(data$classA),])

df\_r <- df[order(df$classA),]

rowanno <- as.data.frame(df\_r[,"classA"])

colnames(rowanno) <- "classinfo"

rownames(rowanno) <- df\_r$Metabolite

colanno <- as.data.frame(c(rep("rp",7),rep("pp",7),rep("rmp",7),rep("QC",4)))

rownames(colanno) <- colnames(df\_r)[6:30]

colnames(colanno) <- "sample group"

mapdf <- as.data.frame(df\_r[,-c(1:5)])

rownames(mapdf) <- df\_r$Metabolite

library(pheatmap)

ann\_colors = list(

`sample group` = c(QC="#35978f",pp="#de77ae",rmp="#762a83",rp="#b2182b"),

classinfo = c(Benzenoids = "#b15928",

`Hydrocarbon derivatives` = "#ffff99",

`Lipids and lipid-like molecules`="#b2df8a",

`Nucleosides, nucleotides, and analogues`="#33a02c",

`Organic acids and derivatives`="#fb9a99",

`Organic nitrogen compounds`="#e31a1c",

`Organic oxygen compounds`="#fdbf6f",

`Organohalogen compounds`="#ff7f00",

`Organoheterocyclic compounds`="#cab2d6",

`Phenylpropanoids and polyketides`="#6a3d9a"))

pheatmap(mapdf,

border\_color = NA,

clustering\_distance\_cols = "correlation",

show\_rownames = FALSE,

cluster\_row = FALSE,

scale="row",

gaps\_row=c(41,42,189,195,214,219,260,261,299),

annotation\_col = colanno,

annotation\_row = rowanno,

annotation\_colors = ann\_colors,

# cellwidth = 10,

color = colorRampPalette(c("navy", "white", "firebrick3"))(50))

1. **The codes for RNA-sequencing (including Principal component analysis (PCA) of root transcriptomic variance and DEG profile of transcripts associated with phenylpropanoid and flavonoid biosynthesis) were deposited at http://github.com/fjxc1893/RNA-seq.**
2. **Principal coordinate analysis (PCoA) of bacterial dispersion among samples based on Bray-Curtis distance.**

library(vegan)

otu=read.table (file.choose(),header=T,row.names = 1)

otu=t(otu)

dis <- vegdist(otu)

groups <- factor(c(rep(1,9), rep(2,9),rep(3,9)), labels = c("PPr","P-Rr","PM-Rr"))

groups <- factor(c(rep(1,9), rep(2,9)), labels = c("PPr","PM-Rr"))

groups <- factor(c(rep(1,9), rep(2,9)), labels = c("PPr","P-Rr"))

mod <- betadisper(dis, groups)

plot(mod, ellipse = TRUE, hull = FALSE, conf = 0.80)

summary(mod)

scrs <- scores(mod)

scrs

my\_cols <- c("#1b9e77", "#7570b3")

plot(mod, col = my\_cols, pch = c(16,17), cex = 1.1)

1. **Ternary plot of bacterial OTUs shared among the different peanut rhizosphere communities.**

library(vcd)

otu=read.table("Ternary\_Diagram\_CM.txt",header=T)

otu1=otu[,c(1:9)]

pch <- otu$pch

col <- as.character(otu$col)

pdf(file="Ternary\_Diagram\_CM.pdf")

ternaryplot(otu1, scale=1, #scale,row sums scale to be used

dimnames=NULL, #dimension labels (defaults to the column names of x),

dimnames\_position=c("corner","edge","none"), dimnames\_color="black",

id=NULL, id\_color = "black",

id\_just=c("center", "center"),

coordinates=FALSE, #if TRUE, the coordinates of the points are plotted below them. coordinates and id are mutual exclusive.

grid=TRUE, #if TRUE, a grid is plotted. May optionally be a string indicating the line type (default: "dotted").

grid\_color="gray",

labels="outside",labels\_color="darkgray",labels=c("inside", "outside", "none").

border="black", bg="white",

pch=pch,

cex=1, #a numerical value giving the amount by which plotting text and symbols should be scaled relative to the default. Ignored for the symbol size if propsize is not FALSE.

prop\_size=FALSE, #if TRUE, the symbol size is plotted proportional to the row sum of the three variables, i.e., represents the weight of the observation.

col=col,

main="ternary plot")

dev.off()

1. **Residual plots for checking the assumptions of homoscedasticity and normality**

df=read.table("data\_reg.txt",header=T)

model <- lm(N4\_biomass~N4\_fixation, data=df)

summary(model)

#define residuals

res <- resid(model)

#produce residual vs. fitted plot

plot(fitted(model), res)

#add a horizontal line at 0

abline(0,0)

pdf("homoscedasticity\_N4.pdf",width=10,height=8)

plot(fitted(model), res)

abline(0,0)

dev.off()

#create Q-Q plot for residuals

qqnorm(res)

#add a straight diagonal line to the plot

qqline(res)

pdf("normality\_N4.pdf",width=10,height=8)

qqnorm(res)

qqline(res)

dev.off()