Network exploration

### Installing/loading libraries

# install.packages("BiocManager")   
# BiocManager::install("WGCNA")   
  
packs <- c("sna", "tsna", "ndtv", "tidyverse", "Hmisc")  
for (i in 1:length(packs)){  
 if(packs[i] %in% rownames(installed.packages()) == FALSE){  
 install.packages(packs[i], repos="http://cran.univ-paris1.fr/")  
 }  
 library(packs[i], character.only = TRUE)  
}

### Functions

flat\_cor\_mat <- function(cor\_r, cor\_p){  
 #This function provides a simple formatting of a correlation matrix  
 #into a table with 4 columns containing :  
 # Column 1 : row names (variable 1 for the correlation test)  
 # Column 2 : column names (variable 2 for the correlation test)  
 # Column 3 : the correlation coefficients  
 # Column 4 : the p-values of the correlations  
 library(tidyr)  
 library(tibble)  
 cor\_r <- rownames\_to\_column(as.data.frame(cor\_r), var = "row")  
 cor\_r <- gather(cor\_r, column, cor, -1)  
 cor\_p <- rownames\_to\_column(as.data.frame(cor\_p), var = "row")  
 cor\_p <- gather(cor\_p, column, p, -1)  
 cor\_p\_matrix <- left\_join(cor\_r, cor\_p, by = c("row", "column"))  
 cor\_p\_matrix  
}

### Data

* Input data files

indir <- "/Users/susana.martinez/Documents/LAO\_TS\_analysis/MGE\_analysis/Host\_CRISPR\_MGE\_analysis/Networks"  
  
# networks  
net\_ph.in <- paste(indir, "Final\_rbin\_phagenet.tsv", sep = "/")  
net\_pl.in <- paste(indir, "Final\_rbin\_plasmidnet.tsv", sep = "/")  
  
# node attribute files, with the abundance/expression of each omic  
# MG  
mg.in <- paste(indir, "ALL\_nodes\_mg\_attr.txt", sep = "/")  
# MT  
mt.in <- paste(indir, "ALL\_nodes\_mt\_attr.txt", sep = "/")

### Loading data

# networks  
net\_ph <- read\_tsv(net\_ph.in)

## Parsed with column specification:  
## cols(  
## rbin = col\_character(),  
## pspcc = col\_character()  
## )

net\_pl <- read\_tsv(net\_pl.in)

## Parsed with column specification:  
## cols(  
## rbin = col\_character(),  
## pspcc = col\_character()  
## )

#### Check that there are not ambiguous MGEs  
intersect(unique(net\_ph$pspcc),unique(net\_pl$pspcc))

## character(0)

# mg and mt information  
mg <- read\_tsv(mg.in)

## Parsed with column specification:  
## cols(  
## .default = col\_double(),  
## node = col\_character(),  
## type = col\_character()  
## )

## See spec(...) for full column specifications.

mt <- read\_tsv(mt.in)

## Parsed with column specification:  
## cols(  
## .default = col\_double(),  
## node = col\_character(),  
## type = col\_character()  
## )  
## See spec(...) for full column specifications.

### Correlations between phages and hosts

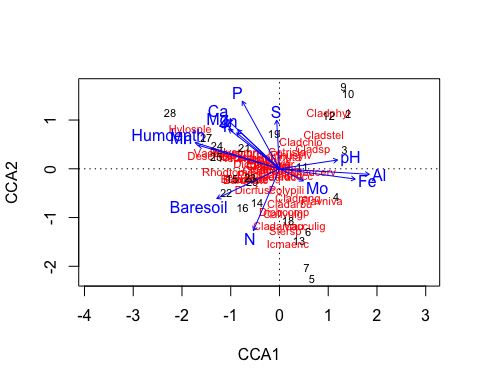
mg.ph <- mg %>% filter(node %in% net\_ph$pspcc) %>% gather(date, mg, -node, -type) %>% spread(node, mg) %>% select(-type, -date)  
mg.host <- mg %>% filter(node %in% net\_ph$rbin) %>% gather(date, mg, -node, -type) %>% spread(node, mg) %>% select(-type, -date)  
all\_to\_corr <- cbind(mg.host, mg.ph)  
  
# Pearson correlation  
my\_cor <- rcorr(as.matrix(all\_to\_corr))  
my\_cor\_matrix <- flat\_cor\_mat(my\_cor$r, my\_cor$P)  
  
as.tibble(my\_cor\_matrix) %>%   
 filter(p < 0.005) %>%   
 select(row, column, cor) %>%   
 filter(row %in% net\_ph$rbin) %>%   
 filter(column %in% net\_ph$pspcc) %>%   
 spread(column, cor) -> corr\_tb  
  
as.tibble(my\_cor\_matrix) %>%   
 filter(p < 0.005) %>%   
 select(row, column, p) %>%   
 filter(row %in% net\_ph$rbin) %>%   
 filter(column %in% net\_ph$pspcc) %>%   
 spread(column, p) -> p\_tb  
  
# cca = canonical correlation analysis  
library(vegan)  
## add PC parameters and scale  
## plot it with bacteria as arrows

EXAMPLE of CCA

data(varespec)  
data(varechem)  
## Common but bad way: use all variables you happen to have in your  
## environmental data matrix  
vare.cca <- cca(varespec, varechem)  
vare.cca

## Call: cca(X = varespec, Y = varechem)  
##   
## Inertia Proportion Rank  
## Total 2.0832 1.0000   
## Constrained 1.4415 0.6920 14  
## Unconstrained 0.6417 0.3080 9  
## Inertia is scaled Chi-square   
##   
## Eigenvalues for constrained axes:  
## CCA1 CCA2 CCA3 CCA4 CCA5 CCA6 CCA7 CCA8 CCA9 CCA10   
## 0.4389 0.2918 0.1628 0.1421 0.1180 0.0890 0.0703 0.0584 0.0311 0.0133   
## CCA11 CCA12 CCA13 CCA14   
## 0.0084 0.0065 0.0062 0.0047   
##   
## Eigenvalues for unconstrained axes:  
## CA1 CA2 CA3 CA4 CA5 CA6 CA7 CA8 CA9   
## 0.19776 0.14193 0.10117 0.07079 0.05330 0.03330 0.01887 0.01510 0.00949

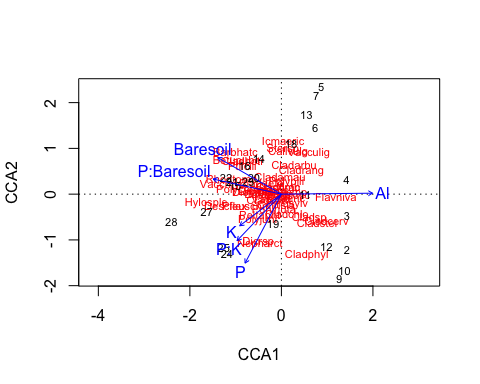
plot(vare.cca)



## Formula interface and a better model  
vare.cca <- cca(varespec ~ Al + P\*(K + Baresoil), data=varechem)  
vare.cca

## Call: cca(formula = varespec ~ Al + P \* (K + Baresoil), data =  
## varechem)  
##   
## Inertia Proportion Rank  
## Total 2.083 1.000   
## Constrained 1.046 0.502 6  
## Unconstrained 1.038 0.498 17  
## Inertia is scaled Chi-square   
##   
## Eigenvalues for constrained axes:  
## CCA1 CCA2 CCA3 CCA4 CCA5 CCA6   
## 0.3756 0.2342 0.1407 0.1323 0.1068 0.0561   
##   
## Eigenvalues for unconstrained axes:  
## CA1 CA2 CA3 CA4 CA5 CA6 CA7 CA8   
## 0.27577 0.15411 0.13536 0.11803 0.08887 0.05511 0.04919 0.03781   
## (Showing 8 of 17 unconstrained eigenvalues)

plot(vare.cca)



## `Partialling out' and `negative components of variance'  
cca(varespec ~ Ca, varechem)

## Call: cca(formula = varespec ~ Ca, data = varechem)  
##   
## Inertia Proportion Rank  
## Total 2.08320 1.00000   
## Constrained 0.15722 0.07547 1  
## Unconstrained 1.92598 0.92453 22  
## Inertia is scaled Chi-square   
##   
## Eigenvalues for constrained axes:  
## CCA1   
## 0.15722   
##   
## Eigenvalues for unconstrained axes:  
## CA1 CA2 CA3 CA4 CA5 CA6 CA7 CA8   
## 0.4745 0.2939 0.2140 0.1954 0.1748 0.1171 0.1121 0.0880   
## (Showing 8 of 22 unconstrained eigenvalues)

cca(varespec ~ Ca + Condition(pH), varechem)

## Call: cca(formula = varespec ~ Ca + Condition(pH), data =  
## varechem)  
##   
## Inertia Proportion Rank  
## Total 2.0832 1.0000   
## Conditional 0.1458 0.0700 1  
## Constrained 0.1827 0.0877 1  
## Unconstrained 1.7547 0.8423 21  
## Inertia is scaled Chi-square   
##   
## Eigenvalues for constrained axes:  
## CCA1   
## 0.18269   
##   
## Eigenvalues for unconstrained axes:  
## CA1 CA2 CA3 CA4 CA5 CA6 CA7 CA8   
## 0.3834 0.2749 0.2123 0.1760 0.1701 0.1161 0.1089 0.0880   
## (Showing 8 of 21 unconstrained eigenvalues)

## RDA  
data(dune)  
data(dune.env)  
dune.Manure <- rda(dune ~ Manure, dune.env)  
plot(dune.Manure)

