Genetic Effects of a Foundation Tree on Networks of Leaf Modifying Arthropods

Species/Functional Groups

- PB = P. betae
- pb.pred = parasitic fly
- pb.abort =
- edge.fold = sawfly
- pinch =
- mid.miner = lep
- edge.miner = lep
- tip.miner = lep
- fish.eye = ??
- tier = lep
- thrips = Thysanura
- chew.holes = ??
- chomp = large herbivore
- scrape = radula
- chew.edge = lep?

Pre-processing Data

Load datasets

```
pit <- read.csv('../data/acn/arth_cooc_PIT_Lau.csv')
pit[is.na(pit)] <- 0
```

Restrict main analysis to live leaves only

```
### separating out "fungal" necrosis
### necrosis <- pit$fungal
pit <- pit[,colnames(pit) != 'fungal']
```

Remove Genotype 1007

```
pit <- pit[pit[, "geno"] != "1007", ]
```

Remove mite

```
pit <- pit[, colnames(pit) != "mite"]
```

Combine gall variants

```r
pb <- pit$pb.upper + pit$pb.lower + pit$pb.woody
pb.pred <- pit$pb.pred + pit$pb.woody.pred + pit$pb.hole
pb.abort <- pit$pb.abort
pit <- pit[,!(grepl('pb',colnames(pit)))]
pit <- data.frame(pit,pb.abort,pb.pred,pb)
pit <- data.frame(pit[,1:6],pit[,ncol(pit):7])
tree.info <- paste(pit[, "tree"], pit[, "geno"], pit[, "leaf.type"])
tree.arth <- pit[, 7:ncol(pit)]
tree.arth <- split(tree.arth, tree.info)
```

Create the community matrix

```r
com.acn <- do.call(rbind, lapply(tree.arth, function(x) apply(x, 2, sum)))
```

Tree level network models

```r
cn.acn <- lapply(tree.arth, coNet, ci.p = 95, cond = TRUE)
```

Tree network distances

```r
d.cn.acn <- netDist(cn.acn, method = "euclidean")
```

Calculate network metrics

```r
l.cn.acn <- do.call(rbind, lapply(cn.acn, enaR:::structure.statistics))[, "L"]
cen.cn.acn <- unlist(lapply(cn.acn, function(x)
    sna::centralization(x, FUN = sna::degree, normalize = FALSE)),
                 )
nm.cn.acn <- data.frame(L = l.cn.acn, C = cen.cn.acn)
```

Tree info compilation

```r
acn.dat <- data.frame(do.call(rbind, strsplit(names(cn.acn), split = " ")))
names(acn.dat) <- c("tree", "geno", "leaf.type")
```

Limit senscent trees with greater than 20 leaves

```r
n.sen <- unlist(lapply(tree.arth[acn.dat[, "leaf.type"] == "sen"], nrow))
n.live <- unlist(lapply(tree.arth[acn.dat[, "leaf.type"] == "live"], nrow))
sen.trees <- tree.arth[names(tree.arth) %in% names(n.sen)[n.sen >= 20]]
sen.dat <- data.frame(do.call(rbind, strsplit(names(sen.trees), split = " ")))
```

```r
colnames(sen.dat) <- c("tree", "geno", "leaf.type")
com.sen <- do.call(rbind, lapply(sen.trees, function(x) apply(x, 2 ,sum)))
```

Network ordination

```r
if (file.exists("../results/nms_cn_acn.rda")){
    nms.cn.acn <- dget("../results/nms_cn_acn.rda")
}else {
    set.seed(1234)
    ##    nms.cn.acn <- nmds(d.cn.acn, 2, 2)
    nms.cn.acn <- nmds(netDist(cn.acn[grepl("live", names(cn.acn))]), 2, 2)
    dput(nms.cn.acn, "../results/nms_cn_acn.rda")
}
ord.cn.acn <- nmds.min(nms.cn.acn)
vec.com.acn <- envfit(ord.cn.acn,
                      com.acn[grepl("live", rownames(com.acn)), apply(com.acn, 2, sum) > 10])
vec.nm.acn <- envfit(ord.cn.acn, nm.cn.acn[grepl("live", rownames(nm.cn.acn)), ])
```

Senescent network models

```r
cn.sen <- lapply(sen.trees, coNet, ci.p = 95, cond = TRUE)
```

Main Results

Note: genotype 1007 is removed because it only has one rep

Genotype replication

```r
table(acn.dat[acn.dat[, "leaf.type"] == "live", "geno"])
```

```
##
## 1000 1008 1017 1023   11  996   T6
##    6    5    5    5    5    5    4
```

```r
table(sen.dat[, "geno"])
```

```
##
## 1000 1008 1017 1023   11  996   T6
##    4    4    3    4    4    3    4
```

Total abundance

```r
abund <- apply(com.acn, 1, sum)
abund <- apply(com.acn[acn.dat[, "tree"] %in% sen.dat[, "tree"], ], 1, sum)
reml.abund.acn <- lme4::lmer(I(abund^(1/2)) ~ (1 | geno),
                           data = acn.dat[acn.dat[, "tree"] %in% sen.dat[ ,"tree"],],
                           REML = TRUE)
p.reml.abund.acn <- RLRsim::exactRLRT(reml.abund.acn)
p.reml.abund.acn
```

```
##
##  simulated finite sample distribution of RLRT.
##
##  (p-value based on 10000 simulated values)
##
## data:
## RLRT = 6.9127, p-value = 0.0019
```

```
                                        # live trees
RLRsim::exactRLRT(
    lme4::lmer(I(abund[acn.dat[, "leaf.type"] == "live"]^(1/2)) ~ (1 | geno),
               data = acn.dat[acn.dat[, "leaf.type"] == "live", ],
               REML = TRUE)
)
```

```
## singular fit
```

```
##
##  simulated finite sample distribution of RLRT.
##
##  (p-value based on 10000 simulated values)
##
## data:
## RLRT = 0, p-value = 1
```

```
                                        # sen trees
sen.A <- apply(com.sen, 1, sum)
RLRsim::exactRLRT(
    lme4::lmer(I(sen.A) ~ (1 | geno),
               data = sen.dat,
               REML = TRUE)
)
```

```
##
##  simulated finite sample distribution of RLRT.
##
##  (p-value based on 10000 simulated values)
##
## data:
## RLRT = 4.102, p-value = 0.0176
```

Richness

```
rich <- apply(com.acn[acn.dat[, "tree"] %in% sen.dat[, "tree"], ], 1,
              function(x) sum(sign(x)))
reml.rich.acn <- lme4::lmer(I(rich^(1/2)) ~ (1 | geno),
                           data = acn.dat[acn.dat[, "tree"] %in% sen.dat[, "tree"], ],
                           REML = TRUE)
p.reml.rich.acn <- RLRsim::exactRLRT(reml.rich.acn)
p.reml.rich.acn
```

```
##
##  simulated finite sample distribution of RLRT.
##
##  (p-value based on 10000 simulated values)
##
## data:
## RLRT = 7.1066, p-value = 0.0019
```

```
                                        # live trees
RLRsim::exactRLRT(
    lme4::lmer(I(rich[acn.dat[, "leaf.type"] == "live"]^(1/2)) ~ (1 | geno),
               data = acn.dat[acn.dat[, "leaf.type"] == "live", ],
               REML = TRUE)
)
```

```
##
##  simulated finite sample distribution of RLRT.
##
##  (p-value based on 10000 simulated values)
##
## data:
## RLRT = 0.49745, p-value = 0.2055
```

```
                                        # sen trees
sen.R <- apply(com.sen, 1, function(x) sum(sign(x)))
RLRsim::exactRLRT(
    lme4::lmer(I(sen.R) ~ (1 | geno),
               data = sen.dat,
               REML = TRUE)
)
```

```
##
##  simulated finite sample distribution of RLRT.
##
##  (p-value based on 10000 simulated values)
##
## data:
## RLRT = 4.7246, p-value = 0.0119
```

Community Similarity

```
rel.com.acn <- rel(com.acn[acn.dat[, "tree"] %in% sen.dat[, "tree"], ])
ds <- rep(min(rel.com.acn[rel.com.acn != 0]) / 100, nrow(rel.com.acn))
rel.com.acn <- cbind(rel.com.acn, ds)
                                        # combined
set.seed(12234)
vegan::adonis2(rel.com.acn ~ geno * leaf.type,
               data = acn.dat[acn.dat[, "tree"] %in% sen.dat[, "tree"], ],
               strata = acn.dat[acn.dat[, "tree"] %in% sen.dat[, "tree"], "tree"],
               perm = 10000, sqrt.dist = TRUE, mrank = TRUE)
```

```
## Permutation test for adonis under reduced model
## Terms added sequentially (first to last)
## Permutation: free
## Number of permutations: 10000
```

```
##
## vegan::adonis2(formula = rel.com.acn ~ geno * leaf.type, data = acn.dat[acn.dat[, "tree"] %in% sen.da
##                 Df SumOfSqs      R2      F    Pr(>F)
## geno             6   2.4295 0.14588 1.3928    0.0043 **
## leaf.type        1   1.3471 0.08089 4.6336 9.999e-05 ***
## geno:leaf.type   6   1.8302 0.10990 1.0492    0.3271
## Residual        38  11.0475 0.66334
## Total           51  16.6544 1.00000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
                                        # live
set.seed(12234)
vegan::adonis2(rel.com.acn[acn.dat[acn.dat[, "tree"] %in% sen.dat[, "tree"],
                                   "leaf.type"] == "live", ] ~ geno,
               data = acn.dat[acn.dat[, "leaf.type"] == "live" &
                                  acn.dat[, "tree"] %in% sen.dat[, "tree"], ],
               perm = 10000, sqrt.dist = FALSE, mrank = TRUE)
```

```
## Permutation test for adonis under reduced model
## Terms added sequentially (first to last)
## Permutation: free
## Number of permutations: 10000
##
## vegan::adonis2(formula = rel.com.acn[acn.dat[acn.dat[, "tree"] %in% sen.dat[, "tree"], "leaf.type"]
##          Df SumOfSqs      R2      F Pr(>F)
## geno      6   1.3372 0.32436 1.5202 0.0461 *
## Residual 19   2.7855 0.67564
## Total    25   4.1227 1.00000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
                                        # sen
set.seed(12234)
vegan::adonis2(com.sen ~ geno,
               data = sen.dat,
               perm = 10000, sqrt.dist = TRUE, mrank = TRUE)
```

```
## Permutation test for adonis under reduced model
## Terms added sequentially (first to last)
## Permutation: free
## Number of permutations: 10000
##
## vegan::adonis2(formula = com.sen ~ geno, data = sen.dat, permutations = 10000, sqrt.dist = TRUE, mran
##          Df SumOfSqs      R2      F   Pr(>F)
## geno      6   2.3265 0.33532 1.5976 0.009899 **
## Residual 19   4.6115 0.66468
## Total    25   6.9380 1.00000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Network similarity

```
set.seed(12234)
vegan::adonis2(netDist(cn.acn[acn.dat[, "tree"] %in% sen.dat[, "tree"]]) ~
                   leaf.type * geno,
               data = acn.dat[acn.dat[, "tree"] %in% sen.dat[, "tree"], ],
               strata = acn.dat[acn.dat[, "tree"] %in% sen.dat[, "tree"], "tree"],
               sqrt.dist = FALSE, mrank = TRUE)
```

```
## Permutation test for adonis under reduced model
## Terms added sequentially (first to last)
## Permutation: free
## Number of permutations: 999
## 
## vegan::adonis2(formula = netDist(cn.acn[acn.dat[, "tree"] %in% sen.dat[, "tree"]]) ~ leaf.type * gen
##                 Df SumOfSqs      R2      F Pr(>F)  
## leaf.type        1     2.75 0.00360 0.2207  0.718  
## geno             6   200.43 0.26273 2.6821  0.028 *
## leaf.type:geno   6    86.43 0.11330 1.1566  0.349  
## Residual        38   473.26 0.62037                
## Total           51   762.87 1.00000                
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
                                        # Live
set.seed(12234)
vegan::adonis2(netDist(cn.acn[acn.dat[, "leaf.type"] == "live"]) ~ geno,
               data = acn.dat[acn.dat[, "leaf.type"] == "live", ],
               sqrt.dist = FALSE, mrank = TRUE)
```

```
## Permutation test for adonis under reduced model
## Terms added sequentially (first to last)
## Permutation: free
## Number of permutations: 999
## 
## vegan::adonis2(formula = netDist(cn.acn[acn.dat[, "leaf.type"] == "live"]) ~ geno, data = acn.dat[ac
##          Df SumOfSqs      R2      F Pr(>F)   
## geno      6   127.79 0.40888 3.2279   0.01 **
## Residual 28   184.75 0.59112                 
## Total    34   312.54 1.00000                 
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
                                        # Senescence
set.seed(12234)
vegan::adonis2(netDist(cn.sen) ~ geno,  data = sen.dat,
               sqrt.dist = FALSE, mrank = TRUE)
```

```
## Permutation test for adonis under reduced model
## Terms added sequentially (first to last)
## Permutation: free
## Number of permutations: 999
## 
## vegan::adonis2(formula = netDist(cn.sen) ~ geno, data = sen.dat, sqrt.dist = FALSE, mrank = TRUE)
##          Df SumOfSqs      R2      F Pr(>F)
## geno      6   57.311 0.28482 1.2611  0.311
## Residual 19  143.911 0.71518              
## Total    25  201.222 1.00000
```

Network metrics

```
                                        # number of links
reml.l.acn <- lme4::lmer(I(l.cn.acn^(1/1)) ~ (1 | geno),
                           data = acn.dat,
                           REML = TRUE)
p.reml.l.acn <- RLRsim::exactRLRT(reml.l.acn)
p.reml.l.acn
```

```
## 
##  simulated finite sample distribution of RLRT.
## 
##  (p-value based on 10000 simulated values)
## 
## data:  
## RLRT = 5.8921, p-value = 0.0047
```

```
RLRsim::exactRLRT(
    lme4::lmer(I(l.cn.acn[acn.dat[, "leaf.type"] == "live"]^(1/1)) ~ (1 | geno),
               data = acn.dat[acn.dat[, "leaf.type"] == "live", ],
               REML = TRUE)
)
```

```
## 
##  simulated finite sample distribution of RLRT.
## 
##  (p-value based on 10000 simulated values)
## 
## data:  
## RLRT = 7.197, p-value = 0.0032
```

```
                                        # centralization
reml.cen.acn <- lme4::lmer(I(cen.cn.acn^(1/2)) ~ (1 | geno),
                           data = acn.dat,
                           REML = TRUE)
p.reml.cen.acn <- RLRsim::exactRLRT(reml.cen.acn)
p.reml.cen.acn
```

```
## 
##  simulated finite sample distribution of RLRT.
## 
##  (p-value based on 10000 simulated values)
## 
## data:  
## RLRT = 2.6561, p-value = 0.0409
```

```
RLRsim::exactRLRT(
    lme4::lmer(I(cen.cn.acn[acn.dat[, "leaf.type"] == "live"]^(1/2)) ~ (1 | geno),
               data = acn.dat[acn.dat[, "leaf.type"] == "live", ],
               REML = TRUE)
)
```

```
## 
```

```
##  simulated finite sample distribution of RLRT.
##
##  (p-value based on 10000 simulated values)
##
## data:
## RLRT = 2.3608, p-value = 0.0468
```

Proportion of PB singles, doubles and triples

```
n.leaf <- unlist(lapply(tree.arth, nrow))
pb1 <- unlist(lapply(tree.arth, function(x) sum(sign(x[, "pb"] == 1))))
pb2 <- unlist(lapply(tree.arth, function(x) sum(sign(x[, "pb"] == 2))))
pb3 <- unlist(lapply(tree.arth, function(x) sum(sign(x[, "pb"] == 3))))
pb4 <- unlist(lapply(tree.arth, function(x) sum(sign(x[, "pb"] == 4))))
pb.1 <- unlist(lapply(tree.arth, function(x) sum(sign(x[, "pb"] == 1))))
pb1.d <- pb1[grepl("live", names(pb1))]/n.leaf[grepl("live", names(n.leaf))] -
    pb1[grepl("sen", names(pb1))]/n.leaf[grepl("sen", names(n.leaf))]
pb2.d <- pb2[grepl("live", names(pb2))]/n.leaf[grepl("live", names(n.leaf))] -
    pb2[grepl("sen", names(pb2))]/n.leaf[grepl("sen", names(n.leaf))]
pb3.d <- pb3[grepl("live", names(pb3))]/n.leaf[grepl("live", names(n.leaf))] -
    pb3[grepl("sen", names(pb3))]/n.leaf[grepl("sen", names(n.leaf))]

                                        # pb1
RLRsim::exactRLRT(
    lme4::lmer(I(pb1.d[acn.dat[, "tree"] %in% sen.dat[, "tree"]]) ~ (1 | geno),
               data = acn.dat[acn.dat[, "tree"] %in% sen.dat[, "tree"], ],
               REML = TRUE)
)
```

```
##
##  simulated finite sample distribution of RLRT.
##
##  (p-value based on 10000 simulated values)
##
## data:
## RLRT = 0.00011911, p-value = 0.4459
```

```
                                        # pb2
RLRsim::exactRLRT(
    lme4::lmer(I(pb2.d[acn.dat[, "tree"] %in% sen.dat[, "tree"]]) ~ (1 | geno),
               data = acn.dat[acn.dat[, "tree"] %in% sen.dat[, "tree"], ],
               REML = TRUE)
)
```

```
##
##  simulated finite sample distribution of RLRT.
##
##  (p-value based on 10000 simulated values)
##
## data:
## RLRT = 0.00015571, p-value = 0.4383
```

```
                                        # pb3
RLRsim::exactRLRT(
    lme4::lmer(I(pb3.d[acn.dat[, "tree"] %in% sen.dat[, "tree"]]) ~ (1 | geno),
               data = acn.dat[acn.dat[, "tree"] %in% sen.dat[, "tree"], ],
               REML = TRUE)
)
```

```
##
##  simulated finite sample distribution of RLRT.
##
##  (p-value based on 10000 simulated values)
##
## data:
## RLRT = 0.048602, p-value = 0.3615
```

No genotype effect, so all trees are tested together

```
t.test(pb1.d)
```

```
##
##  One Sample t-test
##
## data:  pb1.d
## t = -5.5738, df = 34, p-value = 3.084e-06
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -0.3001851 -0.1397729
## sample estimates:
## mean of x
## -0.219979
```

```
t.test(pb2.d)
```

```
##
##  One Sample t-test
##
## data:  pb2.d
## t = -3.4302, df = 34, p-value = 0.001599
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -0.06366719 -0.01629347
## sample estimates:
##   mean of x
## -0.03998033
```

```
t.test(pb3.d)
```

```
##
##  One Sample t-test
##
## data:  pb3.d
## t = -0.44204, df = 34, p-value = 0.6613
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -0.003198507  0.002055650
## sample estimates:
```

```
##     mean of x
## -0.0005714286
```

Modularity of bipartite networks

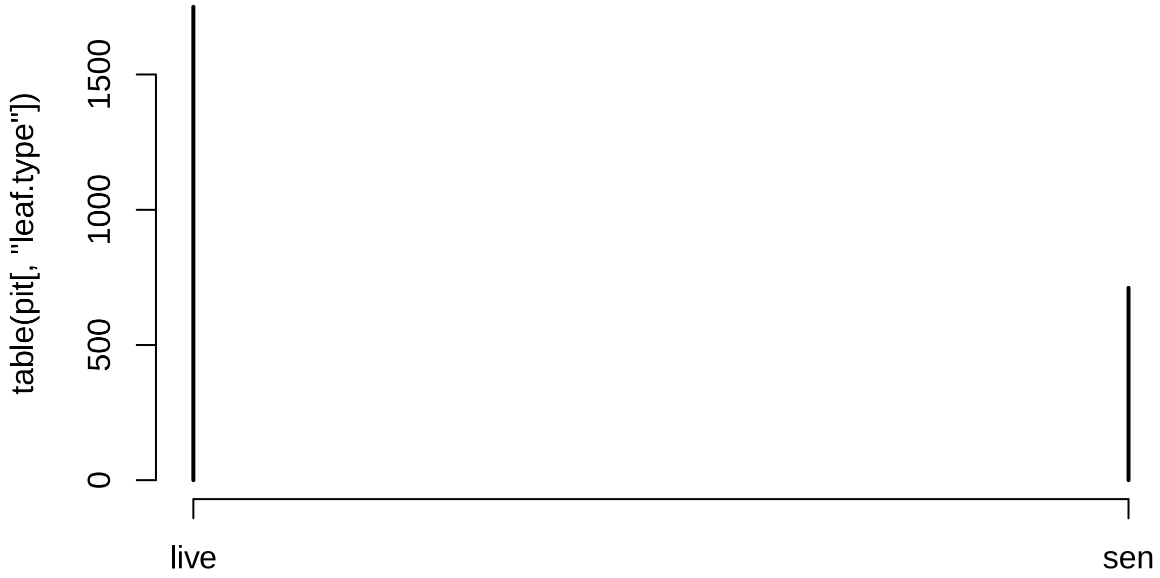
```
bipartite::computeModules(com.acn[grepl("live",
                                        rownames(com.acn)), ])
bipartite::computeModules(com.acn[grepl("sen",
                                        rownames(com.acn)), ])
```

Plots

Main Results

- Genotypes differ in network structure on living leaves
- Response to PB is linked to senescence, more PB higher prob(senscence)

```
                                        # Pb frequency
plot(table(pit[, "leaf.type"]))
```



```
table(pit[pit[, "leaf.type"] == "live", "pb"])
```

```
##
##    0    1    2    3
## 1682   65    2    1
```

```
table(pit[pit[, "leaf.type"] == "sen", "pb"])
```

```
##
##   0   1   2   3   4
## 502 181  26   1   1
```

Need to fix the following:

```
                                        # total abundance (live vs sen)
                                        # richness (live vs sen)
mdc.plot(acn.dat[, "leaf.type"], abund, ylim = c(-1.5, 1.5),
         xlab = "Tree Genotype", ylab = "Value", std = TRUE,
         ord = order(tapply(abund, acn.dat[, "leaf.type"], mean), decreasing = TRUE))
mdc.plot(acn.dat[, "leaf.type"], rich, add = TRUE, pch = 1, xjit = 0,
         ord = order(tapply(abund, acn.dat[, "leaf.type"], mean), decreasing = TRUE))
legend("topright", legend = c("Abundance", "Richness"), pch = c(19, 1), bty = "none")
```

Need to fix the following:

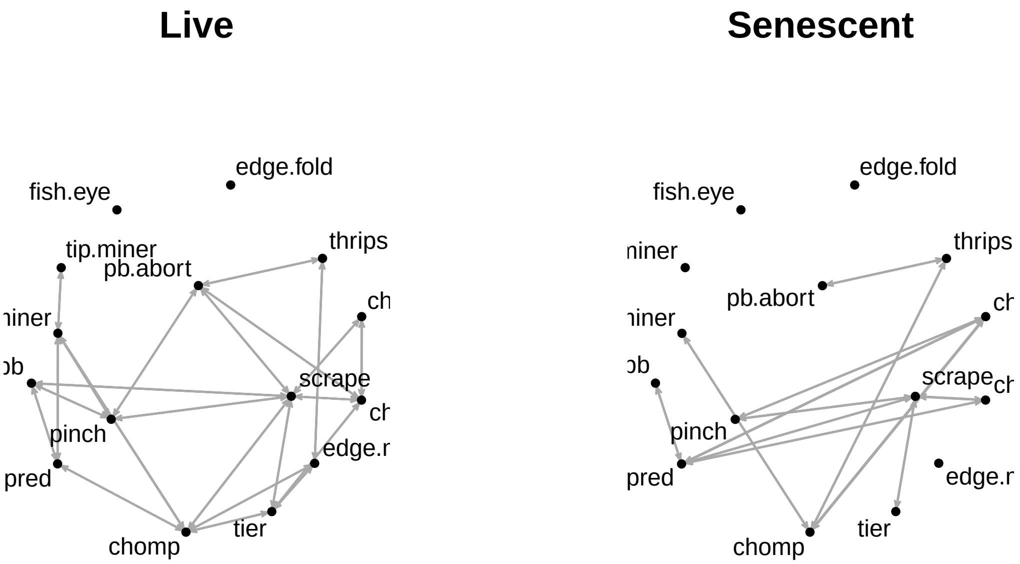
```
                                     # abundance and richness
mdc.plot(acn.dat[, "geno"], abund, ylim = c(-1.5, 1.5),
         xlab = "Tree Genotype", ylab = "Standardized Metric",
         ord = order(tapply(abund, acn.dat[, "geno"], mean), decreasing = TRUE))
mdc.plot(acn.dat[, "geno"], rich, add = TRUE, pch = 1, xjit = 0.01,
         ord = order(tapply(abund, acn.dat[, "geno"], mean), decreasing = TRUE))
legend("topright", legend = c("Abundance", "Richness"), pch = c(19, 1), bty = "none")
```

Need to fix the following:

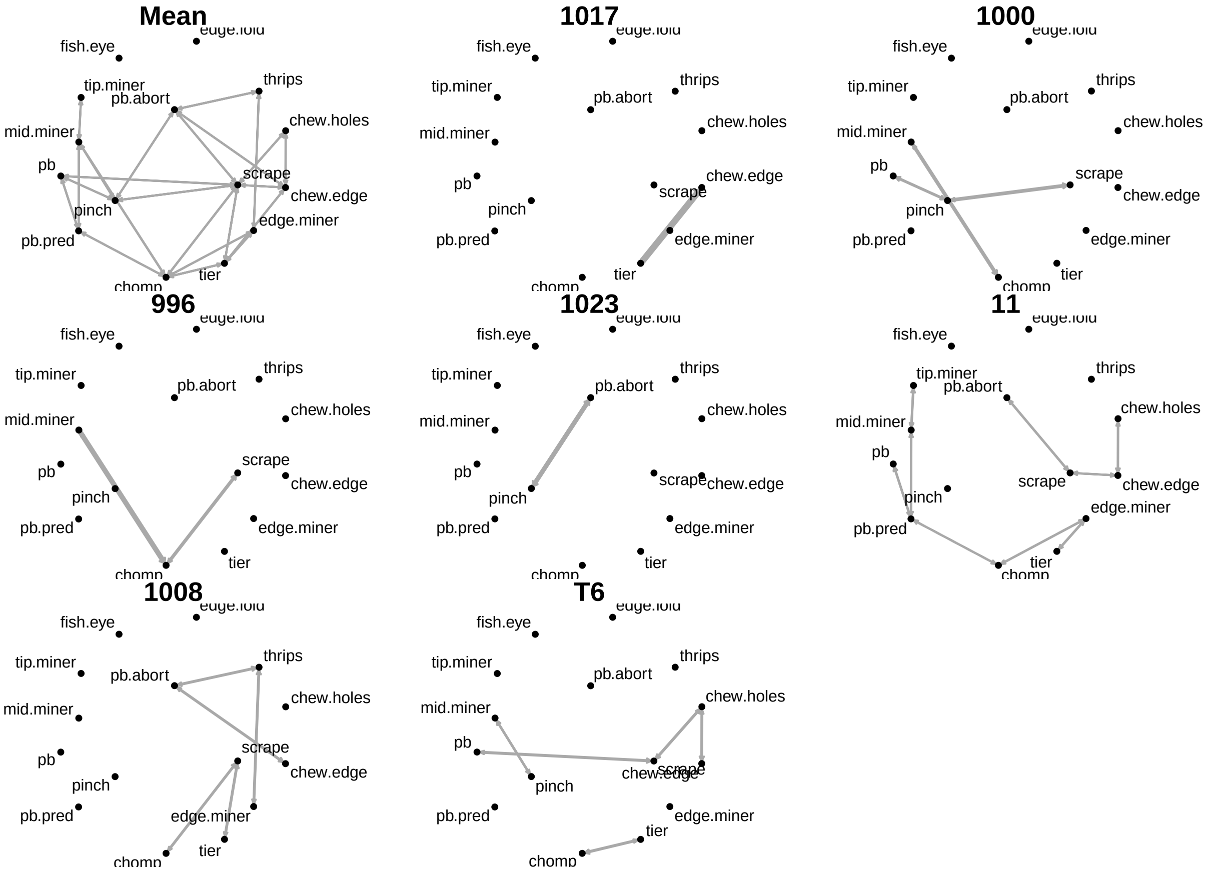
```
                                        # Links and centralization
mdc.plot(acn.dat[, "geno"], l.cn.acn, ylim = c(-1, 1.5),
         xlab = "Tree Genotype", ylab = "Standardized Metric",
         ord = order(tapply(abund, acn.dat[, "geno"], mean), decreasing = TRUE))
mdc.plot(acn.dat[, "geno"], cen.cn.acn, add = TRUE, pch = 1, xjit = 0.01,
         ord = order(tapply(abund, acn.dat[, "geno"], mean), decreasing = TRUE))
legend("topright", legend = c("Links", "Centralization"), pch = c(19, 1), bty = "none")
```

Network Plots

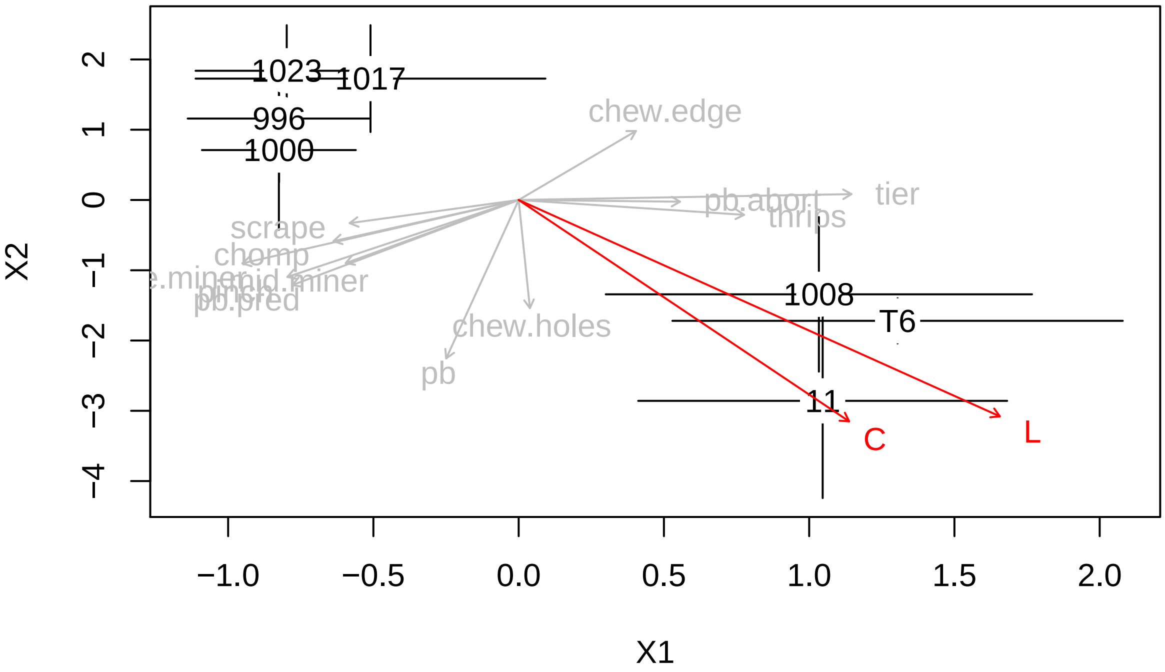
```
                                       # Live versus Sen
par(mfrow = c(1, 2))
set.seed(12234)
net.col <- sign(netMean(cn.acn))
net.col[net.col == -1] <- "red"
net.col[net.col == "1"] <- "darkgrey"
coord <- gplot(abs(netMean(cn.acn[acn.dat[, "leaf.type"] == "live"])),
               gmode = "digraph",
               displaylabels = TRUE,
               edge.lwd = (abs(netMean(cn.acn[acn.dat[, "leaf.type"] == "live"]))) * 10,
               edge.col = net.col,
               vertex.col = "black",
               vertex.cex = 0.5,
               arrowhead.cex = 0.5,
               label.cex = 0.75,
               main = "Live")
net.col <- sign(netMean(cn.sen))
net.col[net.col == -1] <- "red"
net.col[net.col == "1"] <- "darkgrey"
gplot(abs(netMean(cn.sen)),
      coord = coord,
      gmode = "digraph",
      displaylabels = TRUE,
      edge.lwd = (abs(netMean(cn.sen))) * 10,
      edge.col = net.col,
      vertex.col = "black",
      vertex.cex = 0.5,
      arrowhead.cex = 0.5,
      label.cex = 0.75,
      main = "Senescent")
```



```
                                        # By Genotype
par(mfrow = c(1, 2))
set.seed(12234)
net.col <- sign(netMean(cn.acn))
net.col[net.col == -1] <- 2
net.col[net.col == 1] <- "darkgrey"
# pdf(file = "../results/acn_live_nets.pdf", width = 9, height = 9)
par(mfrow = c(3, 3), mar = c(0, 0, 1, 0))
coord <- gplot(abs(netMean(cn.acn[acn.dat[, "leaf.type"] == "live"])),
               gmode = "digraph",
               displaylabels = TRUE,
               edge.lwd = (abs(netMean(cn.acn[acn.dat[, "leaf.type"] == "live"]))) * 10,
               edge.col = net.col,
               vertex.col = "black",
               vertex.cex = 0.5,
               arrowhead.cex = 0.5,
               label.cex = 0.75,
               main = "Mean")
for (i in unique(acn.dat[, "geno"])){
    gplot(abs(netMean(cn.acn[acn.dat[, "geno"] == i &
                                  acn.dat[, "leaf.type"] == "live"])),
          coord = coord,
          gmode = "digraph",
          displaylabels = TRUE,
          edge.lwd = (abs(netMean(cn.acn[acn.dat[, "geno"] == i &
                                              acn.dat[, "leaf.type"] == "live"]))) * 10,
          edge.col = net.col,
          vertex.col = "black",
          vertex.cex = 0.5,
          arrowhead.cex = 0.5,
          label.cex = 0.75,
          main = i)
}
# dev.off()
```



```
                                        # Network ordination
coord <- ch.plot(ord.cn.acn, g = acn.dat[acn.dat[, "leaf.type"] == "live", "geno"],
                 cex = 3, mu.pch = 19, pt.col = "white")
text(coord, labels = rownames(coord))
plot(vec.com.acn, col = "grey")
plot(vec.nm.acn, col = "red")
```



Library call script

```
libs <- c("ComGenR", "coNet")
if (!(all(libs %in% installed.packages()[, 1]))){
    devtools::install_github("ECGen/ComGenR")
    devtools::install_github("ECGen/coNet", ref = "master")
}
sapply(libs, require, character.only = TRUE, quietly = TRUE)
```