GENOTYPIC VARIATION INFLUENCES LICHEN CO-OCCURRENCE NETWORK STRUCTURE

M.K. LAU, L.J. LAMIT, T.G. WHITHAM

1. Things to Do

(1) Do a whole network analysis using PerMANOVA
(2) Can we do pathway de-proliferation?
(3) Use eigenvector centrality. What is the dominant eigenvalue for each matrix?
(4) Use Fath and Patten's 1998 network mutualism in utility analysis
(5) Fix the gplot inputs for genotype vertex attributes (it is currently using the tree level data)
(6)
(7)

2. Generate lichen network models for all genotypes in each garden

2.1. Load packages.

```
> require(RLRsim)
> require(lme4)
> require(sna)
     Tools for Social Network Analysis
Version      2.2-0 created on      2010-11-21.
copyright (c) 2005, Carter T. Butts, University of California-Irvine
Type help(package="sna") to get started.
> require(vegan)
> require(xtable)
> require(gplots)
gdata: read.xls support for 'XLS' (Excel 97-2004) files ENABLED.

gdata: read.xls support for 'XLSX' (Excel 2007+) files ENABLED.
> source("source/CorNets.R")
> source("source/pairs.R")
> se = function(x) {
+     sd(x)/sqrt(length(x))
+ }
> binary = function(x, bin = 0) {
+     x[x > bin] = 1
+     return(x)
+ }
> richness = function(x) {
+     x = apply(x, 2, sum)
+     x[x != 0] = 1
+     sum(x)
+ }
> bplot = function(x, y, ylab = "") {
+     mu. = tapply(y, x, mean)
+     se. = tapply(y, x, se)
+     barplot2(mu., plot.ci = TRUE, ci.u = mu. + se., ci.l = mu. -
+         se., las = 2, ylab = ylab)
+ }
> which.garden = function(x) {
+     x = unlist(strsplit(as.character(x), split = ".", fixed = TRUE))[1]
+     x = unlist(strsplit(x, split = ""))
+     if (length(x) < 3) {
+         x = "ONC"
+     }
+     else {
+         x = "PIT"
+     }
+     return(x)
+ }
> lco2quad = function(x) {
+     com = x[, -1:-6]
+     com = apply(com, 1, sum)
+     x = x[, 5:6]
+     out = array(0, dim = c(max(x), max(x)))
+     for (i in (1:nrow(x))) {
+         out[x[i, 1], x[i, 2]] = com[i]
+     }
+     return(out)
+ }
> exp.m <- function(mat, n) {
+     if (n == 1)
+         return(mat)
+     result <- diag(1, ncol(mat))
+     while (n > 0) {
+         if (n%%2 != 0) {
+             result <- result %*% mat
+             n <- n - 1
+         }
+         mat <- mat %*% mat
+         n <- n/2
+     }
+     return(result)
+ }
> proliferate = function(g = "graph", lim = 10000) {
+     n = numeric()
+     for (i in 1:lim) {
+         n[i] = sum(exp.m(g, i))
+         if (n[i] == Inf) {
+             break
+         }
+     }
+     n = n[-i]
+     return(n)
+ }
```

2.2. **Data Summary.** NOTE: there was a data entry error for tree N4.30, changed 11 to 1 in LCO_data_ONC_PIT.csv.

```
> lco = read.csv("data/LCO_data_ONC_PIT.csv")
> xtable(summary(lco[, 1:6]), table.placement = "tbp")
```

	Tree	Geno	Year	Quadrat	Row	Col
1	N1.10 : 100	996 :1000	Min. :2010	n45.55:4850	Min. : 1.0	Min. : 1.0
2	N1.11 : 100	1008 : 900	1st Qu.:2011	n80.90:3050	1st Qu.: 3.0	1st Qu.: 3.0
3	N1.24 : 100	1017 : 750	Median :2011		Median : 5.5	Median : 5.5
4	N1.27 : 100	11 : 700	Mean :2011		Mean : 5.5	Mean : 5.5
5	N1.28 : 100	10 : 600	3rd Qu.:2011		3rd Qu.: 8.0	3rd Qu.: 8.0
6	N1.31 : 100	1023 : 600	Max. :2011		Max. :10.0	Max. :10.0
7	(Other):7300	(Other):3350				

```
> xtable(summary(lco[, 7:10]), table.placement = "tbp")
```

	Xgal	Csub	Lsp	Chol
1	Min. :0.0000	Min. :0.00000	Min. :0.00000	Min. :0.000000
2	1st Qu.:0.0000	1st Qu.:0.00000	1st Qu.:0.00000	1st Qu.:0.000000
3	Median :1.0000	Median :0.00000	Median :0.00000	Median :0.000000
4	Mean :0.5084	Mean :0.09165	Mean :0.02076	Mean :0.009494
5	3rd Qu.:1.0000	3rd Qu.:0.00000	3rd Qu.:0.00000	3rd Qu.:0.000000
6	Max. :1.0000	Max. :1.00000	Max. :1.00000	Max. :1.000000

```
> xtable(summary(lco[, 13:ncol(lco)]), table.placement = "tbp")
```

	Pads	Pund	Rsp
1	Min. :0.000000	Min. :0.000000	Min. :0.000000
2	1st Qu.:0.000000	1st Qu.:0.000000	1st Qu.:0.000000
3	Median :0.000000	Median :0.000000	Median :0.000000
4	Mean :0.002532	Mean :0.002025	Mean :0.002532
5	3rd Qu.:0.000000	3rd Qu.:0.000000	3rd Qu.:0.000000
6	Max. :1.000000	Max. :1.000000	Max. :1.000000

2.3. Separate Community Matrixes by Tree.

```
> lco = lco[lco$Quadrat == "n45.55", ]
> lco.l = list()
> tree.names = as.character(unique(lco$Tree))
```

```
> for (i in (1:length(tree.names))) {
+     lco.l[[i]] = lco[lco$Tree == tree.names[i], ]
+ }
> names(lco.l) = tree.names
> any(table(tree.names) != 1)
```

```
[1] FALSE
```

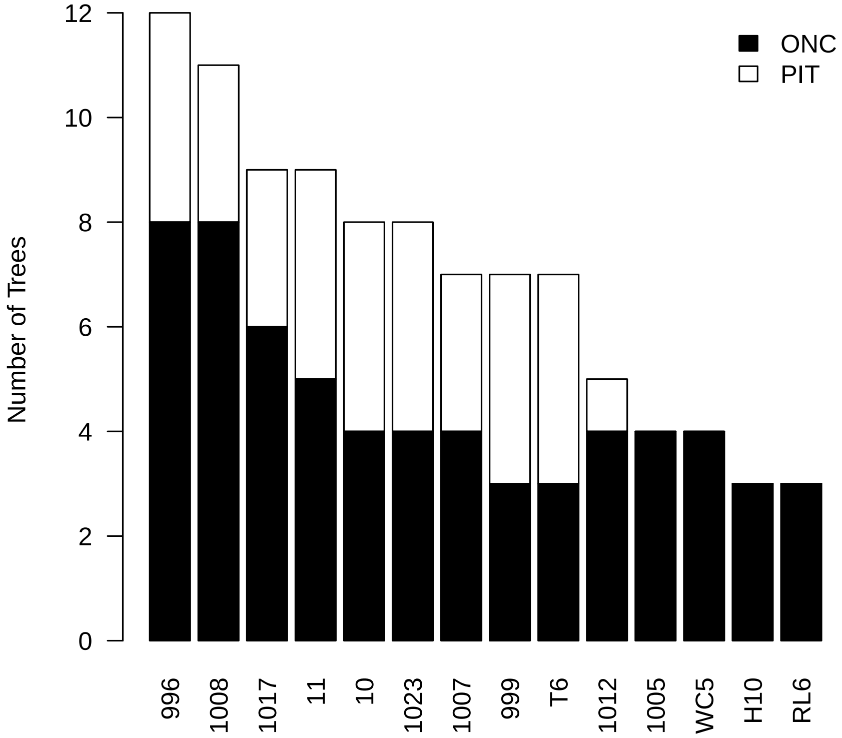
2.4. Separate garden, genotype and community data.

```
> garden = sapply(tree.names, which.garden)
> geno = unlist(lapply(lco.l, function(x) as.character(x$Geno[1])))
> com.l = lapply(lco.l, function(x) x[, (1:ncol(x))[colnames(x) ==
+     "Xgal"]:ncol(x)])
```

```
> geno.table = tapply(factor(geno), garden, table)
> geno.table = rbind(geno.table$ONC, geno.table$PIT)
> rownames(geno.table) = c("ONC", "PIT")
> geno.table = geno.table[, order(apply(geno.table, 2, sum), decreasing = TRUE)]
> xtable(geno.table)
```

	996	1008	1017	11	10	1023	1007	999	T6	1012	1005	WC5	H10	RL6
ONC	8	8	6	5	4	4	4	3	3	4	4	4	3	3
PIT	4	3	3	4	4	4	3	4	4	1	0	0	0	0

```
> barplot(geno.table, beside = FALSE, las = 2, ylab = "Number of Trees",
+     col = c(1, 0))
> legend("topright", c("ONC", "PIT"), fill = c(1, 0), bty = "n")
```



2.5. **Convert to Quadrat View.**

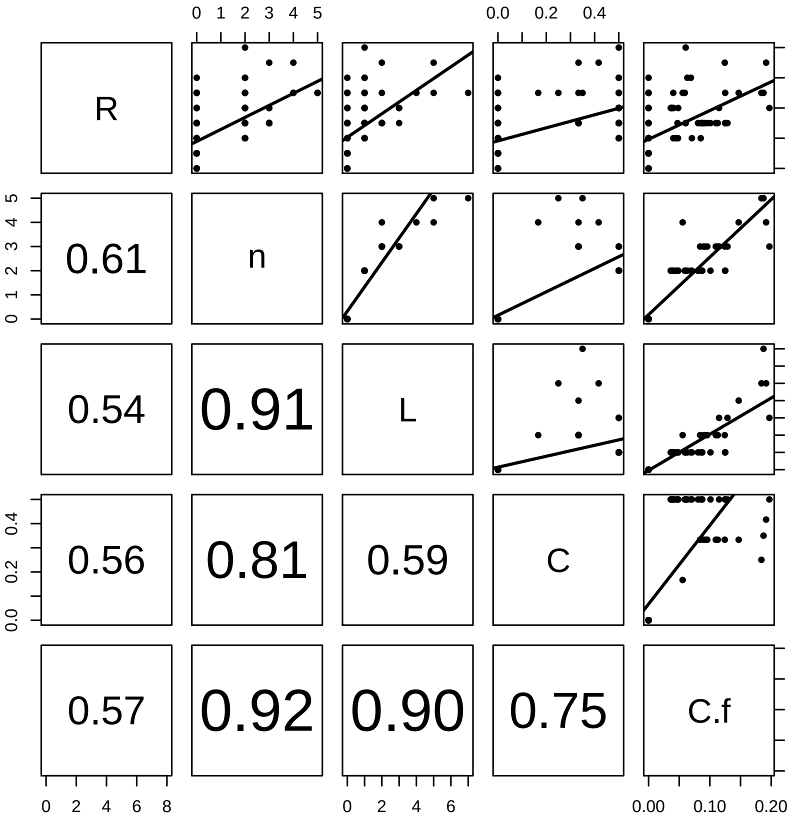
```
> quad.l = lapply(lco.l, lco2quad)
```

Using the `kendall.pairs` function at an α level of 0.05 and adjusting for multiple tests, we can model each community network using the pairwise correlations of all species pairs.

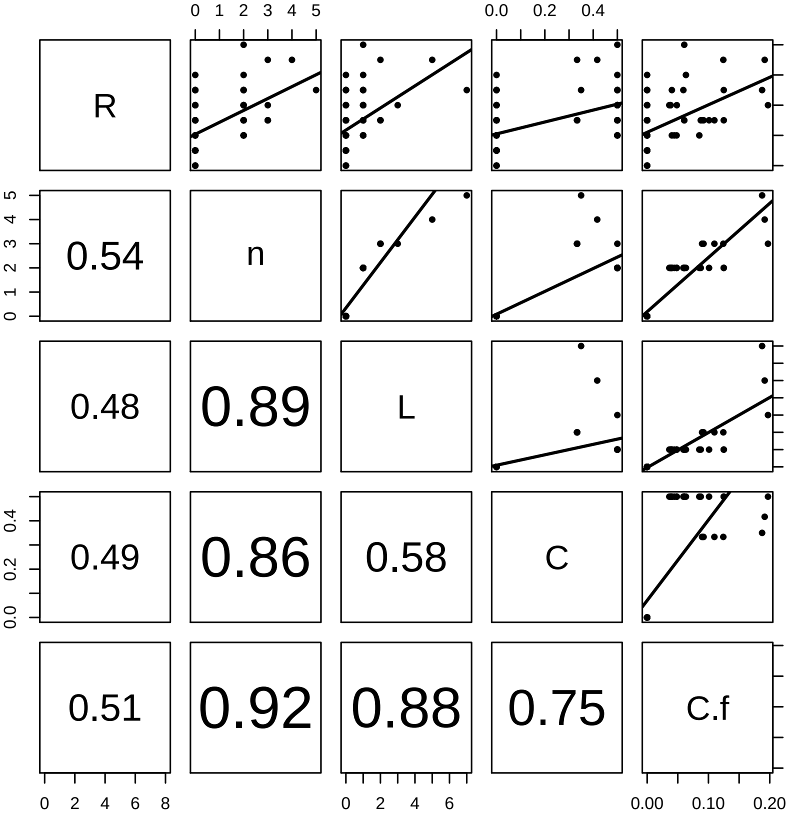
```
> zero.na = function(x) {
+     x[is.na(x)] = 0
+     return(x)
+ }
> cor.l = lapply(com.l, kendall.pairs, alpha = 0.05, p.adj = TRUE)
> cor.l = lapply(cor.l, zero.na)
```

3. Network Statistics

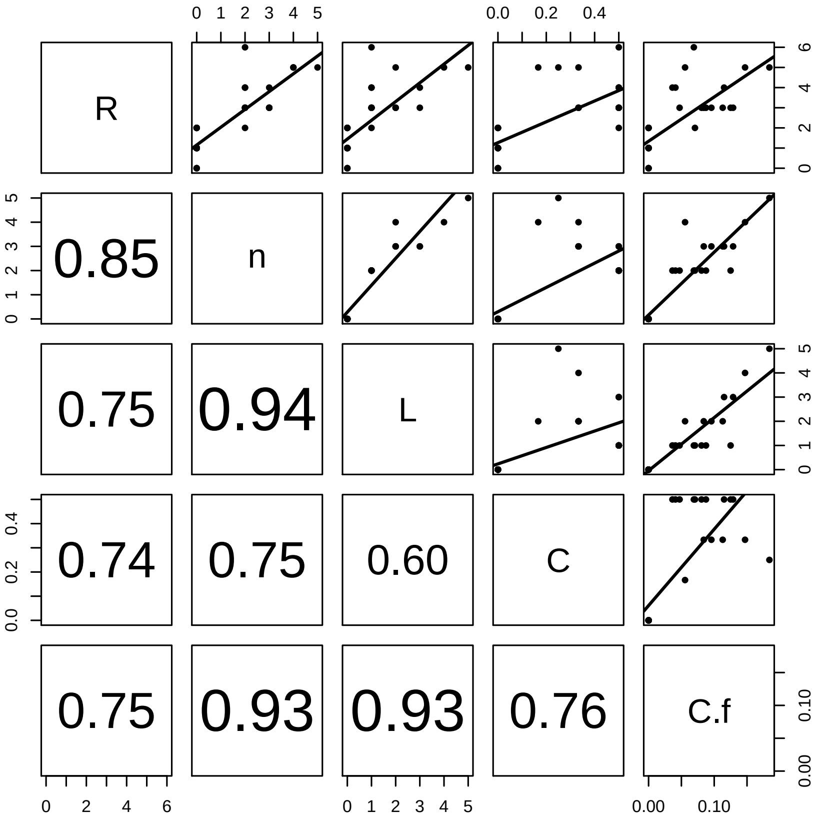
```
> A = lapply(com.l, function(x) apply(x, 2, sum))
> A. = lapply(A, binary)
> R = unlist(lapply(com.l, richness))
> n = unlist(lapply(cor.l, function(x) length(apply(x, 1, sum)[abs(apply(x,
+     1, sum)) > 0])))
> L = unlist(lapply(cor.l, function(x) length(x[x != 0])/2))
> C = L/(n * (n - 1))
> C[is.na(C)] = 0
> abs.cor.l = lapply(cor.l, abs)
> C.f = unlist(lapply(abs.cor.l, function(x) centralization(x,
+     degree, mode = "graph")))

> input.pairs = cbind(R, n, L, C, C.f)
> pairs.onc = input.pairs[garden == "ONC", ]
> pairs.pit = input.pairs[garden == "PIT", ]

> pairs(input.pairs, upper.panel = panel.lm, lower.panel = panel.cor)
```



```
> pairs(pairs.onc, upper.panel = panel.lm, lower.panel = panel.cor)
```



```
> pairs(pairs.pit, upper.panel = panel.lm, lower.panel = panel.cor)
```



3.1. **REML.**

Separate the analysis of ONC and PIT due to unequal representation of genotypes at both gardens.

```
> geno.onc = geno[garden == "ONC"]
> n.onc = n[garden == "ONC"]
> log.n.onc = log(n.onc + 0.5)
> L.onc = L[garden == "ONC"]
> log.L.onc = log(L.onc + 0.5)
> C.onc = C[garden == "ONC"]
> Cf.onc = C.f[garden == "ONC"]
> asin.sqrt.CF.onc = asin(sqrt(Cf.onc))
> exactRLRT(lmer(n.onc ~ 1 | geno.onc))
```

```
simulated finite sample distribution of RLRT. (p-value based on 10000 simulated values)
   data: RLRT = 3.151, p-value = 0.0325
```

```
> exactRLRT(lmer(log.n.onc ~ 1 | geno.onc))
```

```
simulated finite sample distribution of RLRT. (p-value based on 10000 simulated values)
   data: RLRT = 4.9866, p-value = 0.0099
```

```
> exactRLRT(lmer(L.onc ~ 1 | geno.onc))
```

```
simulated finite sample distribution of RLRT. (p-value based on 10000 simulated values)
   data: RLRT = 0, p-value = 0.4589
```

```
> exactRLRT(lmer(log.L.onc ~ 1 | geno.onc))
```

```
simulated finite sample distribution of RLRT. (p-value based on 10000 simulated values)
   data: RLRT = 3.3797, p-value = 0.0287
```

```
> exactRLRT(lmer(C.onc ~ 1 | geno.onc))
```

```
simulated finite sample distribution of RLRT. (p-value based on 10000 simulated values)
   data: RLRT = 7.0159, p-value = 0.0038
```

```
> exactRLRT(lmer(Cf.onc ~ 1 | geno.onc))
```

```
simulated finite sample distribution of RLRT. (p-value based on 10000 simulated values)
   data: RLRT = 0.8348, p-value = 0.16
```

```
> exactRLRT(lmer(asin.sqrt.CF.onc ~ 1 | geno.onc))
```

```
simulated finite sample distribution of RLRT. (p-value based on 10000 simulated values)
   data: RLRT = 2.8031, p-value = 0.0423
```

```
> geno.pit = geno[garden == "PIT"]
> n.pit = n[garden == "PIT"]
> log.n.pit = log(n.pit + 0.5)
> L.pit = L[garden == "PIT"]
> log.L.pit = log(L.pit + 0.5)
> C.pit = C[garden == "PIT"]
> Cf.pit = C.f[garden == "PIT"]
> asin.sqrt.CF.pit = asin(sqrt(Cf.pit))
> exactRLRT(lmer(n.pit ~ 1 | geno.pit))
```

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

```
> exactRLRT(lmer(log.n.pit ~ 1 | geno.pit))
```

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

```
> exactRLRT(lmer(L.pit ~ 1 | geno.pit))
```

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

```
> exactRLRT(lmer(log.L.pit ~ 1 | geno.pit))
```

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

```
> exactRLRT(lmer(C.pit ~ 1 | geno.pit))
```

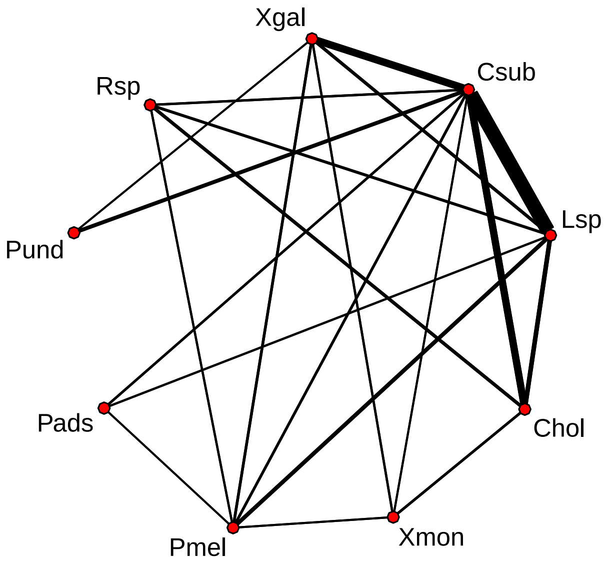
```
simulated finite sample distribution of RLRT. (p-value based on 10000 simulated values)
   data: RLRT = 0, p-value = 1
> exactRLRT(lmer(Cf.pit ~ 1 | geno.pit))
simulated finite sample distribution of RLRT. (p-value based on 10000 simulated values)
   data: RLRT = 0, p-value = 1
> exactRLRT(lmer(asin.sqrt.CF.pit ~ 1 | geno.pit))
simulated finite sample distribution of RLRT. (p-value based on 10000 simulated values)
   data: RLRT = 0, p-value = 1
```

4. Plots

4.1. Network Graphs.

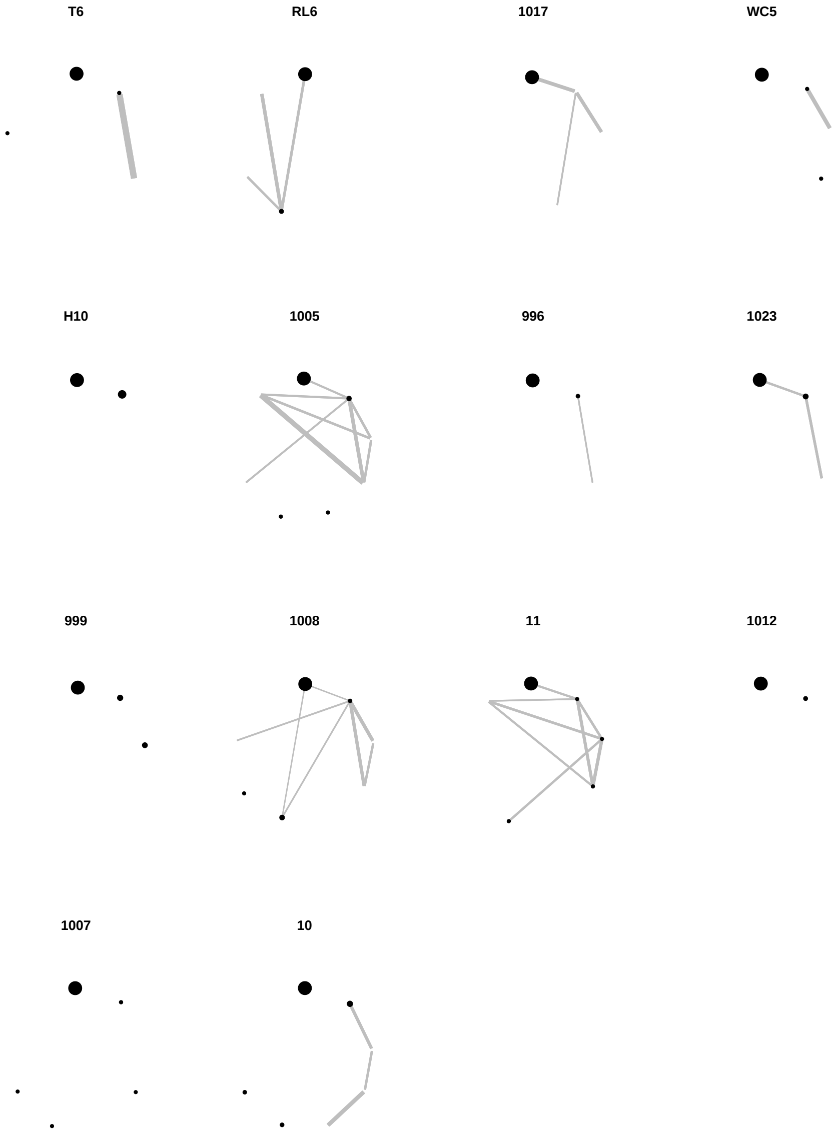
Graph of the mean connections across all trees.

```
> gplot(G.graph, gmode = "graph", displaylabels = TRUE, mode = "circle",
+      edge.lwd = G.graph * 200)
```



Mean graphs for each genotype.

```
> cor.l. = cor.l[garden == "ONC"]
> geno.A = A[garden == "ONC"]
> geno.A. = A.[garden == "ONC"]
> geno.net = list()
> for (i in (1:length(unique(geno.onc)))) {
+     x = cor.l.[geno.onc == unique(geno.onc)[i]]
+     geno.net[[i]] = x[[1]]
+     for (j in (2:length(x))) {
+         geno.net[[i]] = geno.net[[i]] + x[[j]]
+     }
+     geno.net[[i]] = geno.net[[i]]/length(x)
+ }
> names(geno.net) = unique(geno)
```



T6 RL6 1017 WC5

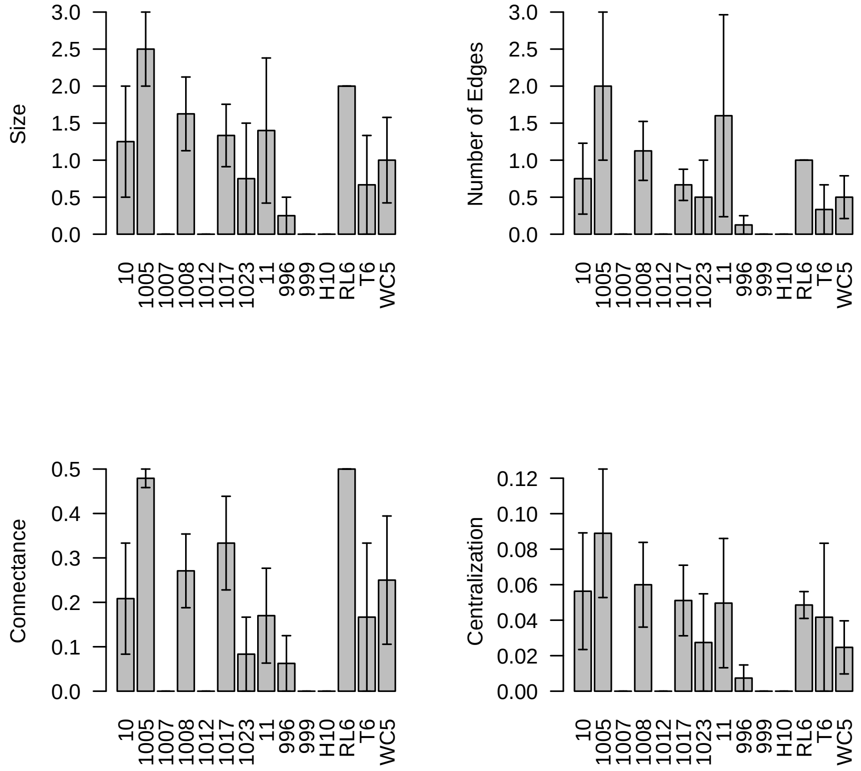
H10 1005 996 1023

999 1008 11 1012

1007 10

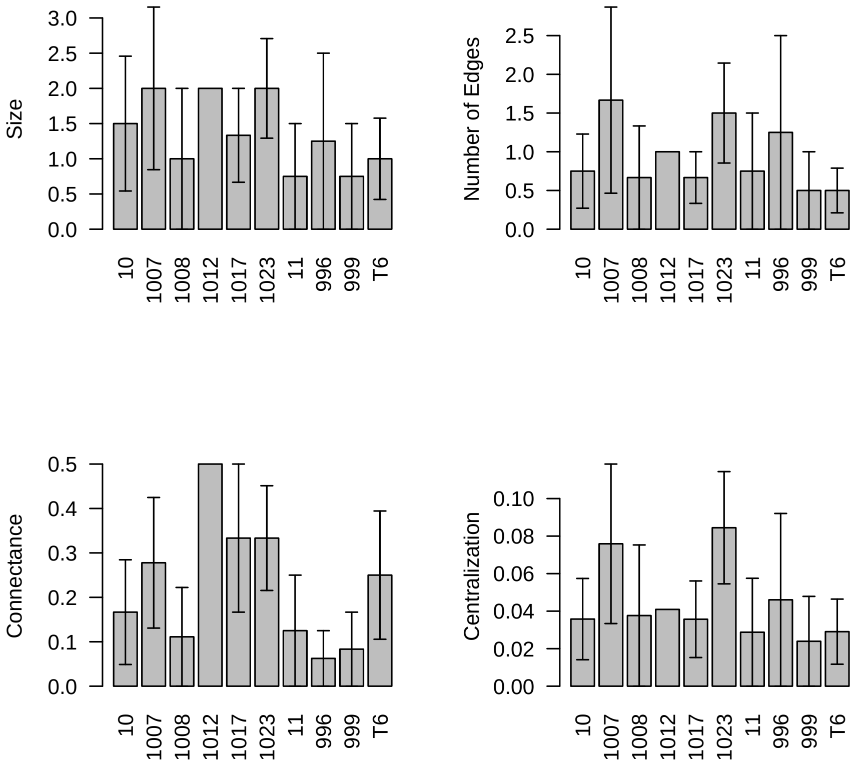
ONC Network Statistics

```
> par(mfrow = c(2, 2))
> bplot(geno.onc, n.onc, "Size")
> bplot(geno.onc, L.onc, "Number of Edges")
> bplot(geno.onc, C.onc, "Connectance")
> bplot(geno.onc, Cf.onc, "Centralization")
```



PIT Network Statistics

```
> par(mfrow = c(2, 2))
> bplot(geno.pit, n.pit, "Size")
> bplot(geno.pit, L.pit, "Number of Edges")
> bplot(geno.pit, C.pit, "Connectance")
> bplot(geno.pit, Cf.pit, "Centralization")
```


Size
Number of Edges
Connectance
Centralization
T6