Chapter 2 R Markdown

J. P. Cuff

17 November 2020

Chapter 2

This chapter involves the comparison of macronutrient contents between invertebrates and the visual representation of these differences.

Libraries and data

First, these are the necessary libraries:

```
library('mvabund')
library('ggtern')
```

And the necessary files (available upon request):

```
medi <- read.csv("MEDI Example Specimens.csv")
```

Macronutrient content comparison

To test for differences in macronutrient content between taxa and to visualise these differences, we will use "mvabund", so we first need to create an mvabund object. Before that, we will plot histograms to assess the normality of the macronutrient data.

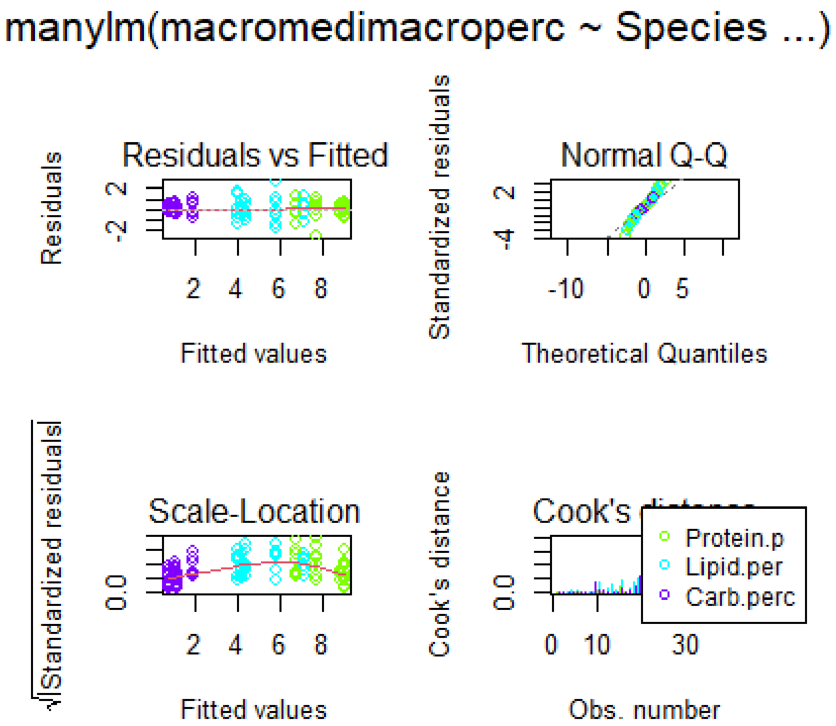
```
hist(medi[,18])
hist(medi[,19])
hist(medi[,20])
```

Given the non-normal distribution of carbohydrates, we will square-root transform the mvabund macronutrient object. We will use proportional macronutrient content (% total macronutrient mass) since body mass measurements could not be accurately obtained for all species.

```
hist(sqrt(medi[,19]))

macromedimacroperc <- mvabund(sqrt(medi[,18:20]))
```

Now we can create the multivariate linear model using the mvabund object that we created above. We can plot the model to check that it meets the necessary assumptions before using the anova function to ascertain whether our taxa significantly differ in their proportional macronutrient contents, including univariate analyses to determine differences in specific macronutrient proportions.

```
mod4<-manylm(macromedimacroperc~Species, data=medi)
plot(mod4)
```



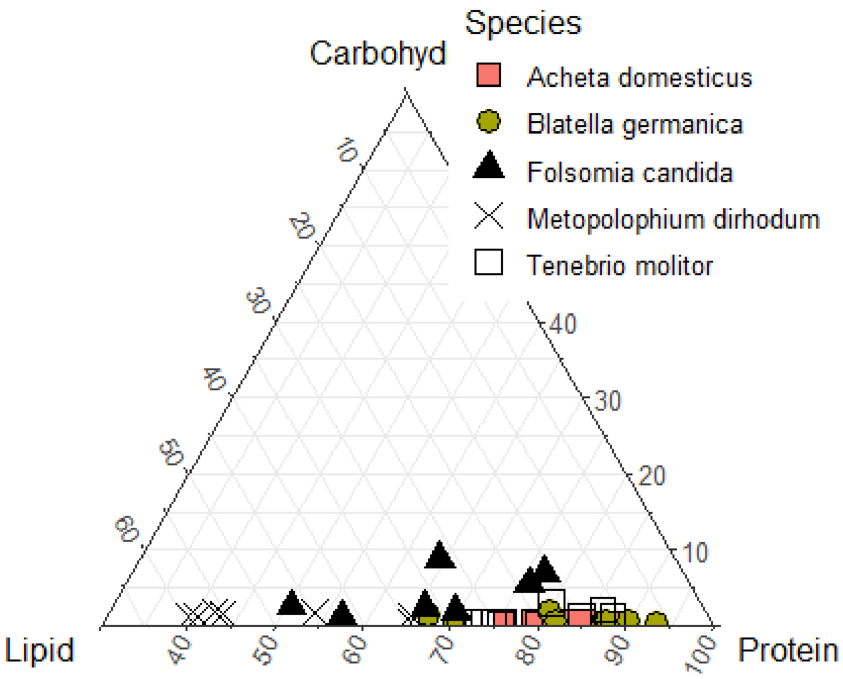
```
anova(mod4, p.uni="adjusted")

## Analysis of Variance Table
##
## Model: manylm(formula = macromedimacroperc ~ Species, data = medi)
##
## Overall test for all response variables
## Test statistics:
##             Res.Df Df.diff val(F) Pr(>F)
## (Intercept)     39
## Species         35       4  38.91  0.002 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Univariate Tests
## Test statistics:
##             Protein.percent.macros        Carb.percent.macros
##                            F value Pr(>F)             F value Pr(>F)
## (Intercept)
## Species                     14.325  0.002              10.522  0.002
##             Lipid.percent.macros
##                          F value Pr(>F)
## (Intercept)
## Species                   14.063  0.002
```

```
##
## Arguments: with 999 resampling iterations using residual (without replacem
ent) resampling and response assumed to be uncorrelated
```

We can see from the plotting output that the model assumptions are generally fine: *Residuals vs Fitted: No dramatic fanning in the top left plot.* Normality: Points approximately follow the qq-line. *Heteroscedasticity: Fairly evenly spread points and level variance.* Residuals vs. Leverage: No obvious influential points.

To visualise the significant difference in macronutrient content between species, we can use a ternary plot via "ggtern".

```
ggtern(medi, aes(x=Lipid.percent.macros,y=Carb.percent.macros, z=Protein.perc
ent.macros))+
  geom_point(size=4, aes(fill=Species, shape=Species)) +
  scale_shape_manual(values=c(22,21,17,4,0)) +
  #scale_colour_manual(values="white", "grey") +
  theme_bw() +
  theme_legend_position('tr') +
  #geom_encircle(alpha=0.5,size=1) +
  xlab("Lipid") + ylab("Carbohydrate") + zlab("Protein") +
  scale_T_continuous(limits=c(.0,.7))   +
  scale_L_continuous(limits=c(.0,.7))  +
  scale_R_continuous(limits=c(.3,1.0))
```



To create a high-resolution output, we can save this as in a PDF file. This will be done for all subsequent thesis plots (but the code will not be presented again for the sake of reducing repetition).

```
pdf("percentmacro.pdf", width = 12, height = 6)
ggtern(medi, aes(x=Lipid.percent.macros,y=Carb.percent.macros, z=Protein.perc
ent.macros))+
  geom_point(size=4, aes(fill=Species, shape=Species)) +
  scale_shape_manual(values=c(22,21,17,4,0)) +
  #scale_colour_manual(values="white", "grey") +
  theme_bw() +
  theme_legend_position('tr') +
  #geom_encircle(alpha=0.5,size=1) +
  xlab("Lipid") + ylab("Carbohydrate") + zlab("Protein") +
  scale_T_continuous(limits=c(.0,.7))   +
  scale_L_continuous(limits=c(.0,.7))  +
  scale_R_continuous(limits=c(.3,1.0))
dev.off()
```

Chapter 3 R Markdown

J. P. Cuff

17 November 2020

Chapter 3

Bioinformatics aggregation

In the final stages of the bioinformatic process, it is necessary to aggregate the data output so that all instances of the same taxon are together. This was achieved in R.

```r
BL17agg <- read.csv("BL17_agg.csv", header = T)
Agg <- aggregate(.~Taxon, data=BL17agg, sum)
write.table(Agg, "BL17_Aggregated.csv")

TL17agg <- read.csv("TL17_agg.csv", header = T)
Agg <- aggregate(.~Taxon, data=TL17agg, sum)
write.table(Agg, "TL17_Aggregated.csv")
```

Following aggregation, the datasets for the two separate primer pairs were combined into one dietary dataset by first aggregating by sample name, then by taxon. Depending on the application, the latter was carried out at the species or family level.

```r
TLBL17samagg <- read.csv("BLTL17samagg.csv", header = T)
Agg <- aggregate(.~Sample, data=TLBL17samagg, sum)
write.table(Agg, "BLTL17_SamAggregated.csv")

TLBL17specagg <- read.csv("BLTL17aggspec.csv", header = T)
Agg <- aggregate(.~Species, data=TLBL17specagg, sum)
write.table(Agg, "BLTL17_SpeciesAggregated.csv")

TLBL17aggfam <- read.csv("BLTL17aggfam.csv", header = T)
Agg <- aggregate(.~Family, data=TLBL17aggfam, sum)
write.table(Agg, "BLTL17_FamilyAggregated.csv")
```

Libraries

First, these are the necessary libraries:

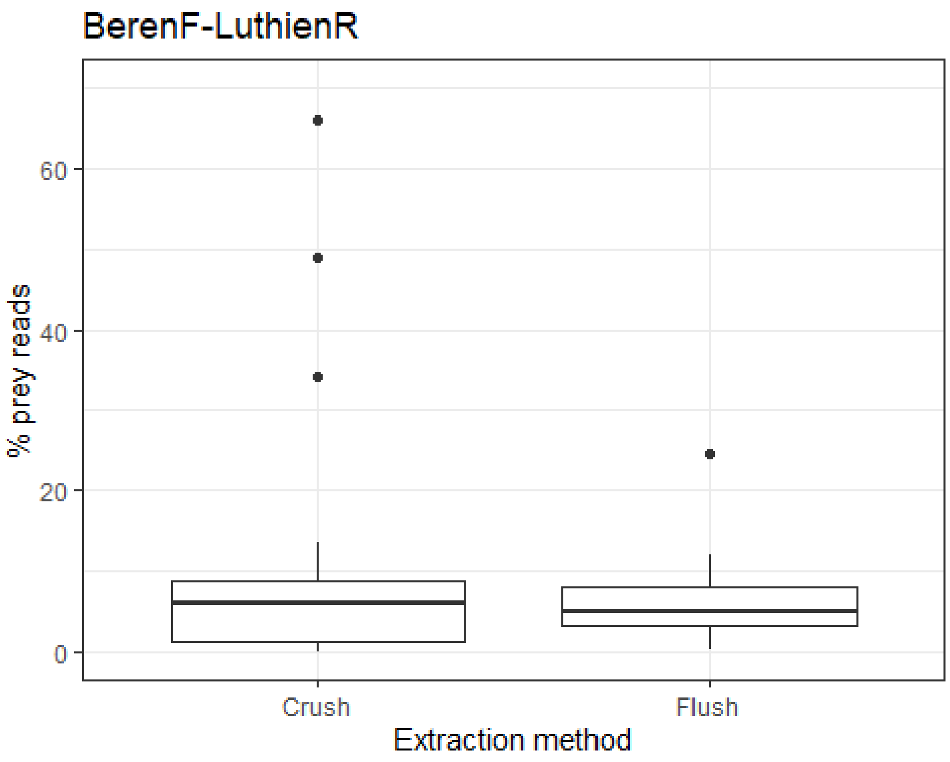
```r
library("devtools")

## Loading required package: usethis

library("vegan")

## Loading required package: permute
```

```
##
## Attaching package: 'permute'

## The following object is masked from 'package:devtools':
##
##     check

## Loading required package: lattice

## This is vegan 2.5-6

library("ggplot2")
library("RColorBrewer")
library("viridis")

## Loading required package: viridisLite

library("mvabund")
library("EcoSimR")

## Loading required package: MASS

library("igraph")

##
## Attaching package: 'igraph'

## The following object is masked from 'package:vegan':
##
##     diversity

## The following object is masked from 'package:permute':
##
##     permute

## The following objects are masked from 'package:stats':
##
##     decompose, spectrum

## The following object is masked from 'package:base':
##
##     union

library("econullnetr")
library("ggrepel")
library("ggplot2")
library("gridExtra")
library("mvabund")
library("econullnetr")
library("fmsb")
```

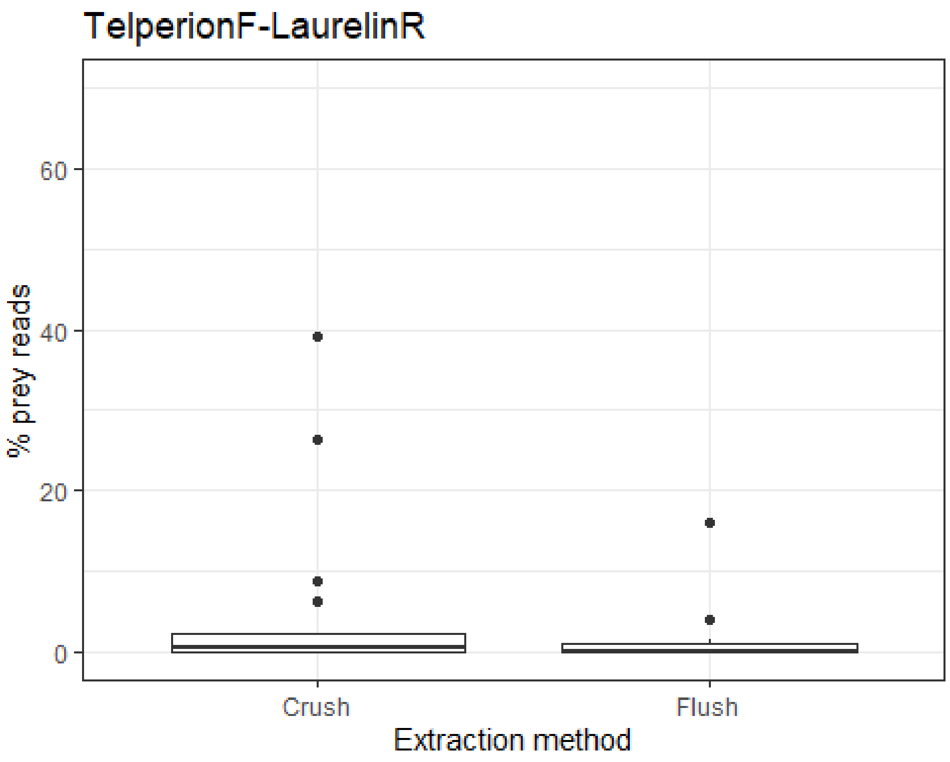
Comparison of DNA extraction techniques

To compare the two extraction techniques used, ANOVA and boxplots were used.

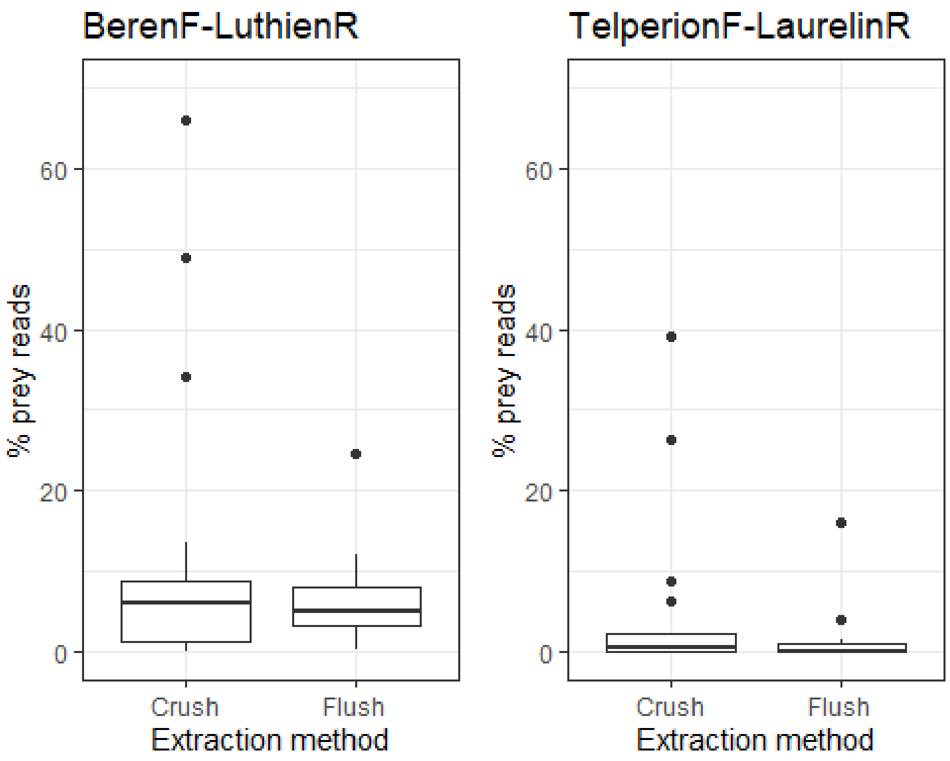
```
genFC <- read.csv("BerenFLuthienRflushcrush.csv")
spiFC <- read.csv("TelperionFLaurelinRflushcrush.csv")

anogenFL <- aov(X.prey ~ Extraction, data=genFC)
anova(anogenFL)

## Analysis of Variance Table
##
## Response: X.prey
##            Df Sum Sq Mean Sq F value Pr(>F)
## Extraction  1  240.1  240.15  1.5826  0.215
## Residuals  44 6676.5  151.74

anospiFL <- aov(X.prey ~ Extraction, data=spiFC)
anova(anospiFL)

## Analysis of Variance Table
##
## Response: X.prey
##            Df  Sum Sq Mean Sq F value Pr(>F)
## Extraction  1   97.61  97.613   1.696 0.2006
## Residuals  38 2187.01  57.553

gggenFL <- ggplot(data = genFC, aes(y = X.prey, x = Extraction)) + geom_boxpl
ot() + ylab("% prey reads")+ xlab("Extraction method") + theme_bw() + ylim(0,
70)+ggtitle("BerenF-LuthienR")
gggenFL
```



```
ggspiFL <- ggplot(data = spiFC, aes(y = X.prey, x = Extraction)) + geom_boxpl
ot() + ylab("% prey reads")+ xlab("Extraction method") + theme_bw() +ylim(0,7
0)+ggtitle("TelperionF-LaurelinR")
ggspiFL
```



```r
grid.arrange(gggenFL, ggspiFL, nrow=1, ncol=2)
```



Invertebrate community comparison

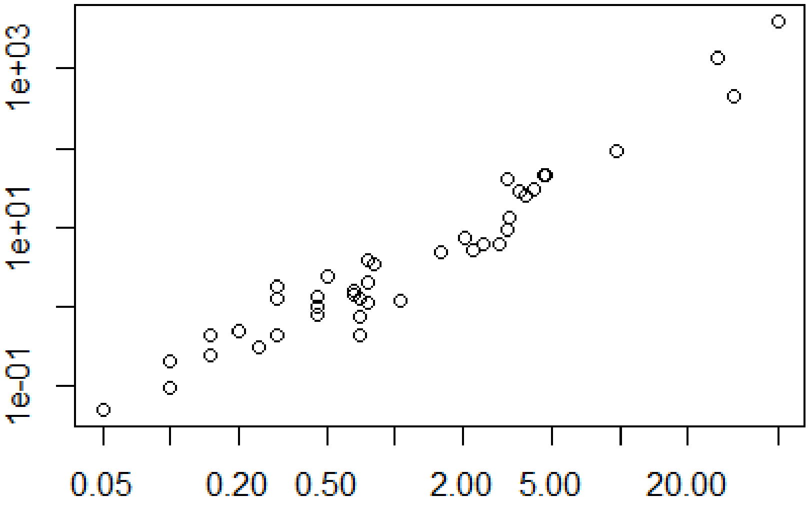
Multivariate GLM

Invertebrate community data were first loaded.

```r
inverts <- read.csv("InvertData.csv")
rownames(inverts) <- inverts[,1]
invertcomm <- inverts[,4:70]
```

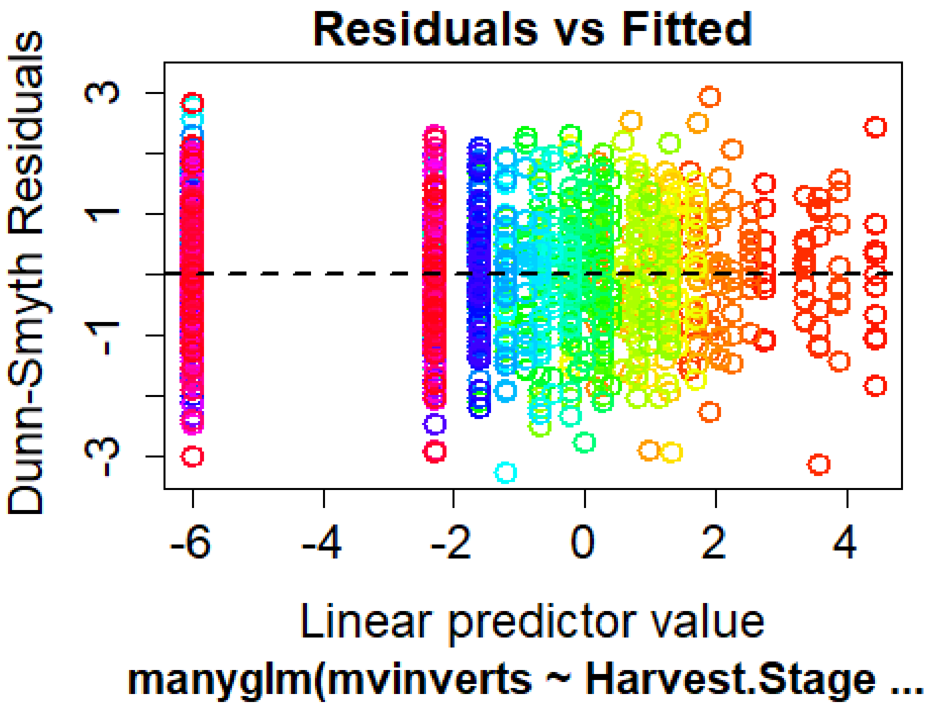
Next, we create an mvabund object and check its mean-variance relationship

```r
mvinverts <- mvabund(inverts[,4:70])
meanvar.plot(mvinverts)
```



Now we can create a multivariate GLM - a negative binomial error family with a cloglog link functon produces the nicest diagnostic plots.

```r
mvinvertsm1 <- manyglm(mvinverts ~ Harvest.Stage, family = "negative.binomial(cloglog)", data = inverts)

plot(mvinvertsm1)
```



Next, we can determine the results of the analysis.

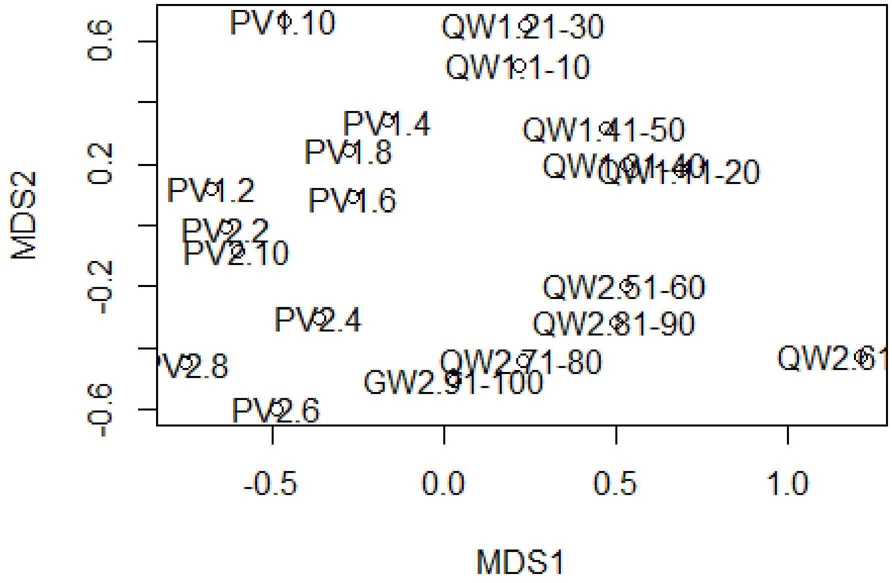
```
anomvinvertsm1 <- anova.manyglm(mvinvertsm1, p.uni = "adjusted", resamp = "mo
ntecarlo")

## Time elapsed: 0 hr 0 min 43 sec

anomvinvertsm1

## Analysis of Deviance Table
##
## Model: mvinverts ~ Harvest.Stage
##
## Multivariate test:
##               Res.Df Df.diff   Dev Pr(>Dev)
## (Intercept)       19
## Harvest.Stage     18       1 227.8    0.001 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Univariate Tests:
##               Agromyzidae          Anisopodidae          Anthocoridae
##                       Dev Pr(>Dev)          Dev Pr(>Dev)          Dev Pr(>
Dev)
## (Intercept)
## Harvest.Stage       3.133    0.934        1.386    1.000        1.386    1
.000
```

```
##               Aphelinidae          Aphididae          Bethylidae
##                       Dev Pr(>Dev)       Dev Pr(>Dev)        Dev Pr(>Dev)
## (Intercept)
## Harvest.Stage       0.597    1.000         0    1.000      1.386    1.000
##               Sminthuroidea          Braconidae          Campichoetidae
##                         Dev Pr(>Dev)        Dev Pr(>Dev)            Dev
## (Intercept)
## Harvest.Stage         1.081    1.000      1.301    1.000          2.773
##                        Canaceidae          Carabidae          Cecidomyiida
e
##               Pr(>Dev)        Dev Pr(>Dev)       Dev Pr(>Dev)           De
v
## (Intercept)
## Harvest.Stage    0.998      1.386    1.000     3.825    0.915         8.56
7
##                        Ceraphronidae          Chironomidae          Chloro
pidae
##               Pr(>Dev)           Dev Pr(>Dev)          Dev Pr(>Dev)
Dev
## (Intercept)
## Harvest.Stage    0.086          2.46    0.999        1.386    1.000
0.355
##                        Ephydridae          Phoridae          Chrysididae
##               Pr(>Dev)        Dev Pr(>Dev)      Dev Pr(>Dev)         Dev
## (Intercept)
## Harvest.Stage    1.000     34.301    0.001    1.473    1.000       1.386
##                        Cicadellidae          Delphacidae          Chrysome
lidae
##               Pr(>Dev)          Dev Pr(>Dev)         Dev Pr(>Dev)
Dev
## (Intercept)
## Harvest.Stage    1.000        1.177    1.000       2.171    0.999
2.426
##                        Chyromyidae          Cryptophagidae
##               Pr(>Dev)         Dev Pr(>Dev)            Dev Pr(>Dev)
## (Intercept)
## Harvest.Stage    0.999       2.773    0.993          2.773    0.998
##               Curculionidae          Cynipidae          Diapriidae
##                         Dev Pr(>Dev)       Dev Pr(>Dev)        Dev Pr(>Dev
)
## (Intercept)
## Harvest.Stage         1.386    1.000     0.463    1.000      0.285    1.00
0
##               Diplopoda          Drosophilidae          Dryomyzidae
##                     Dev Pr(>Dev)           Dev Pr(>Dev)         Dev Pr(>De
v)
## (Intercept)
## Harvest.Stage     1.386    1.000          1.47    1.000       1.473    1.0
00
##               Empididae          Entomobryidae          Erirhinidae
##                      Dev Pr(>Dev)           Dev Pr(>Dev)         Dev Pr(>De
v)
## (Intercept)                                                              
## Harvest.Stage     1.386    1.000         2.626    0.999       6.695    0.2
39
##              Eucoilidae          Eupelmidae          Formicidae         
##                       Dev Pr(>Dev)        Dev Pr(>Dev)        Dev Pr(>Dev)
## (Intercept)                                                            
## Harvest.Stage      3.995    0.908     11.728    0.008      1.386    1.000
##              Henicopidae         Ichneumonidae          Isotomidae      
##                        Dev Pr(>Dev)           Dev Pr(>Dev)        Dev Pr(>D
ev)
## (Intercept)                                                              
## Harvest.Stage       2.773    0.996         0.912    1.000     18.761    0.
001
##              Latrididae          Leiodidae          Limoniidae         
##                       Dev Pr(>Dev)       Dev Pr(>Dev)        Dev Pr(>Dev)
## (Intercept)                                                           
## Harvest.Stage      2.773    0.996     1.473    1.000      1.386    1.000
##              Linyphiidae         Lonchopteridae         Lycosidae       
##                        Dev Pr(>Dev)            Dev Pr(>Dev)       Dev Pr(>D
ev)
## (Intercept)                                                              
## Harvest.Stage       1.915    1.000          0.419    1.000     1.386    1.
000
##              Megaspilidae         Mesostigmata         Microphysidae    
##                         Dev Pr(>Dev)          Dev Pr(>Dev)           Dev
## (Intercept)                                                           
## Harvest.Stage        1.386    1.000        1.041    1.000        12.957
##                         Mymaridae          Nabidae          Orchesellidae
##              Pr(>Dev)       Dev Pr(>Dev)     Dev Pr(>Dev)           Dev
## (Intercept)                                                           
## Harvest.Stage    0.004     1.159    1.000   1.386    1.000          0.01
##                         Oribatida          Pallopteridae          Parasitif
ormes
##              Pr(>Dev)       Dev Pr(>Dev)          Dev Pr(>Dev)            
Dev
## (Intercept)                                                              
## Harvest.Stage    1.000     3.041    0.962         3.238    0.934          1
1.879
##                         Platygastridae          Proctotrupidae          
##              Pr(>Dev)           Dev Pr(>Dev)           Dev Pr(>Dev)
## (Intercept)                                                        
## Harvest.Stage    0.007          1.386    1.000          1.386    1.000
##              Pscodidae          Ptiliidae          Reduviidae         
##                     Dev Pr(>Dev)       Dev Pr(>Dev)       Dev Pr(>Dev)
## (Intercept)                                                          
## Harvest.Stage     2.773    0.996     1.386    1.000          0    1.000
##              Rhopalosomatidae          Rotoitidae          Sciaridae    
##                            Dev Pr(>Dev)        Dev Pr(>Dev)       Dev Pr(>
Dev)
## (Intercept)
## Harvest.Stage            1.386    1.000      1.471    1.000     2.616    0
.999
##               Sepsidae           Sphaeroceridae          Staphylinidae
##                    Dev Pr(>Dev)              Dev Pr(>Dev)             Dev Pr(>
Dev)
## (Intercept)
## Harvest.Stage    1.386    1.000          3.075    0.957         7.422    0
.145
##               Tanaostigmatidae          Thripidae          Torymidae
##                            Dev Pr(>Dev)       Dev Pr(>Dev)       Dev Pr(>D
ev)
## (Intercept)
## Harvest.Stage            2.773    0.998    16.821    0.001     2.216    0.
999
## Arguments:
##  Test statistics calculated assuming uncorrelated response (for faster com
putation)
## P-value calculated using 999 iterations via parametric resampling.
```

Non-metric multidimensional scaling

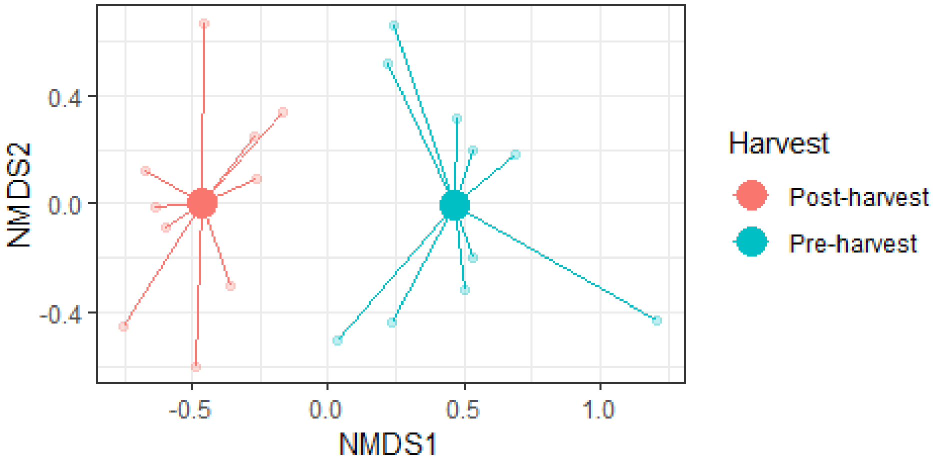
To visualise the data, we can create a non-metric multidimensional scaling array.

```
invert.mds <- metaMDS(comm = invertcomm, distance = "bray", trymax=999, k=2,
trace = FALSE, autotransform = FALSE, na.rm = FALSE)

invert.mds$stress

## [1] 0.1065973

plot(invert.mds$points); text(invert.mds, row.names(invert.mds))
```



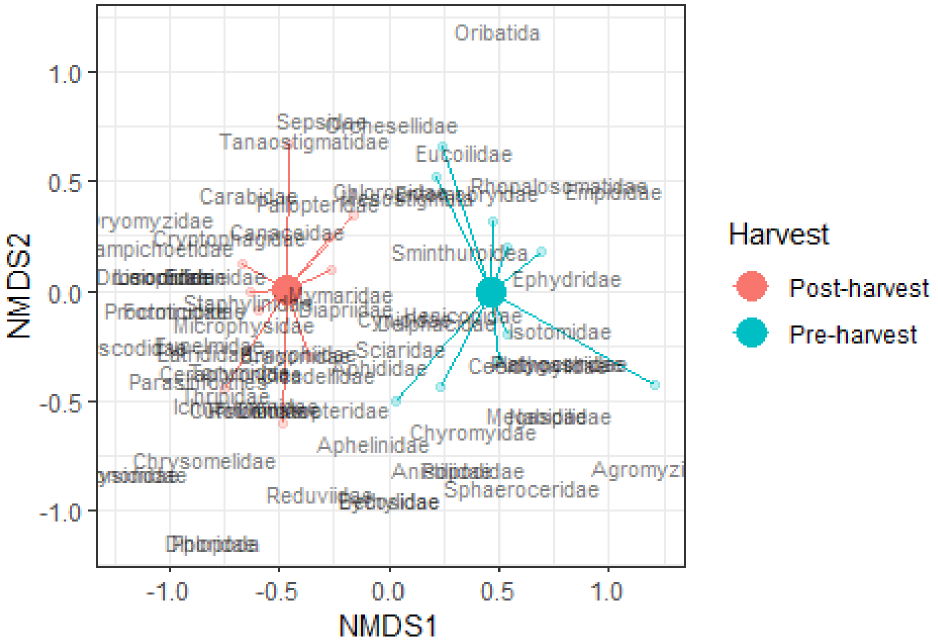
To make this prettier and to create appropriately-titled "spider plots" for comparison of groups, we must first process the NMDS output before then plotting it.

```
scrsi <- scores(invert.mds, display = 'sites')
scrsi <- cbind(as.data.frame(scrsi), Harvest = inverts$Harvest.Stage)
centi <- aggregate(cbind(NMDS1, NMDS2) ~ Harvest, data = scrsi, FUN = mean)
segsi <- merge(scrsi, setNames(centi, c('Harvest', 'oNMDS1', 'oNMDS2')), by =
'Harvest', sort = FALSE)

invert.spider <- ggplot(scrsi, aes(x = NMDS1, y = NMDS2, colour = Harvest)) +
scale_fill_brewer(2,  "Accent") + geom_segment(data = segsi, mapping = aes(xe
nd = oNMDS1, yend = oNMDS2)) + geom_point(data = centi, size = 5, alpha=1) +
geom_point(alpha=0.25) + coord_fixed() + theme_bw()

invert.spider
```



If we want to see the species overlaid, we can add them too.

```
species.scoresi <- as.data.frame(scores(invert.mds, "species"))
species.scoresi$species <- rownames(species.scoresi)
names(species.scoresi)[c(1, 2)] <- c("x", "y")
species.scoresi$z <- NA

invert.spider.sp <- ggplot(species.scoresi, aes(x = x, y = y)) + theme_bw() +
  geom_text(data=species.scoresi,aes(x=x,y=y,label=species), color="black", s
ize=4,alpha=0.75, angle=0)

invert.spiderfull <- ggplot(scrsi, aes(x = NMDS1, y = NMDS2, colour = Harvest
)) + scale_fill_brewer(2,  "Accent") + geom_segment(data = segsi, mapping = a
es(xend = oNMDS1, yend = oNMDS2)) + geom_point(data = centi, size = 5, alpha=
1) + geom_point(alpha=0.25) + coord_fixed() + theme_bw() +
  geom_text(data=species.scoresi,aes(x=x,y=y,label=species), color="black", s
ize=3,alpha=0.5, angle=0) #+ coord_equal()

invert.spiderfull
```



Dietary comparison

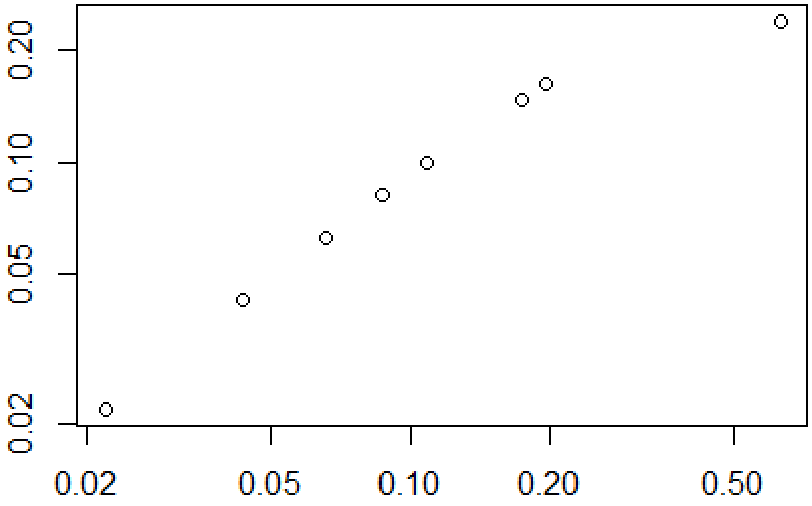
Dietary MGLM

First, we load the data.

```
diet <- read.csv("DietaryData.csv")
rownames(diet) <- diet[,1]
prey <- diet[,9:23]
```

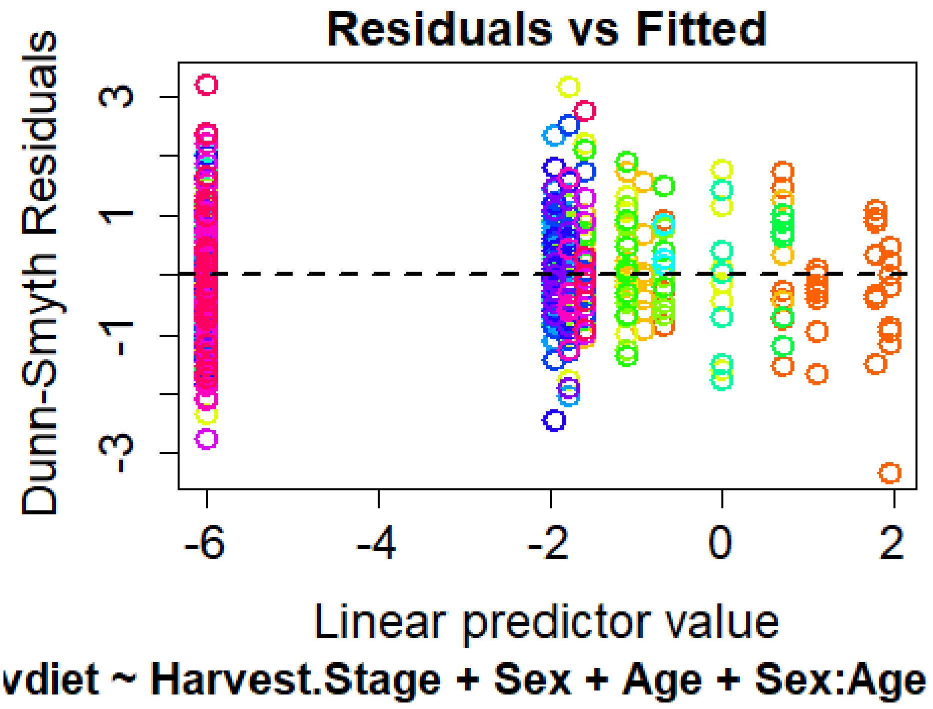
Next, we create the mvabund object.

```
mvdiet <- mvabund(diet[,9:23])
meanvar.plot(mvdiet)
```



And then the model.

```
mvdietm1 <- manyglm(mvdiet ~ Harvest.Stage + Sex + Age +Sex:Age + Harvest.Sta
ge:Sex + Harvest.Stage:Age, family = "binomial", data = diet)
plot(mvdietm1)
```



We can simplify this model based on AIC using 'step'.

```
step(mvdietm1)
```

And, finally, produce the model output.

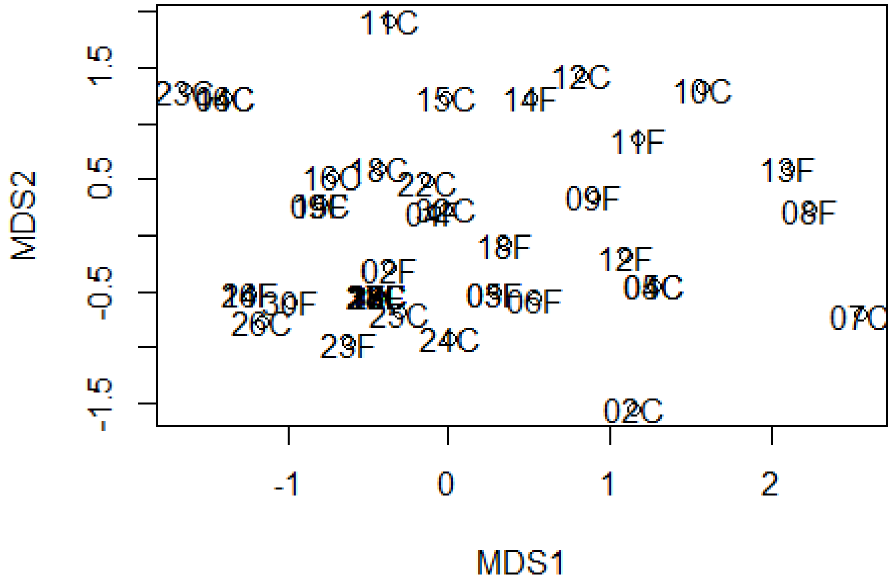
```
mvdietm1 <- manyglm(mvdiet ~ Harvest.Stage + Sex + Age + Harvest.Stage:Age, f
amily = "binomial", data = diet)

anomvdietm1 <- anova.manyglm(mvdietm1, p.uni = "adjusted", resamp = "montecar
lo")

## Time elapsed: 0 hr 0 min 21 sec

anomvdietm1

## Analysis of Deviance Table
##
## Model: mvdiet ~ Harvest.Stage + Sex + Age + Harvest.Stage:Age
##
## Multivariate test:
##                   Res.Df Df.diff   Dev Pr(>Dev)
## (Intercept)           37
## Harvest.Stage         36       1 27.93    0.032 *
## Sex                   35       1 13.62    0.681
## Age                   34       1 22.38    0.171
## Harvest.Stage:Age     33       1 27.43    0.001 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Univariate Tests:
##                   Aphididae          Sminthuroidea          Cecidomyiidae
##                         Dev Pr(>Dev)           Dev Pr(>Dev)           Dev
## (Intercept)
## Harvest.Stage         1.308    0.976         4.148    0.418         5.558
## Sex                   1.056    0.998          0.29    1.000          4.74
## Age                   1.477    0.976         0.252    0.999         7.638
## Harvest.Stage:Age         0    0.776         3.153    0.414             0
##                            Chironomidae          Chloropidae
##                   Pr(>Dev)          Dev Pr(>Dev)         Dev Pr(>Dev)
## (Intercept)
## Harvest.Stage         0.263        1.524    0.947       3.196    0.538
## Sex                   0.348        0.376    1.000        0.04    1.000
## Age                   0.073        1.606    0.945       0.025    0.999
## Harvest.Stage:Age     0.776            0    0.776         0.6    0.776
##                   Cicadellidae          Delphacidae          Entomobryidae
##                            Dev Pr(>Dev)         Dev Pr(>Dev)           Dev
## (Intercept)
## Harvest.Stage            1.308    0.976       0.369    0.991         0.329
## Sex                      1.056    0.998       0.591    1.000         0.017
## Age                      1.477    0.976       0.285    0.999          0.37
## Harvest.Stage:Age            0    0.776       3.404    0.414         2.674
##                            Ephydridae          Isotomidae          Linyphi
idae
##                   Pr(>Dev)        Dev Pr(>Dev)        Dev Pr(>Dev)
Dev
## (Intercept)
## Harvest.Stage         0.991       4.77    0.393      0.492    0.991       3
.221
## Sex                   1.000      0.621    1.000      0.142    1.000       0
.277
## Age                   0.999       0.01    0.999      0.006    0.999       3
.251
## Harvest.Stage:Age     0.516          0    0.776      0.762    0.776       8
.592
##                            Orchesellidae          Phoridae          Sciari
dae
##                   Pr(>Dev)           Dev Pr(>Dev)      Dev Pr(>Dev)
Dev
## (Intercept)
## Harvest.Stage         0.538        0.006    0.991    0.263    0.991     0.
492
## Sex                   1.000        0.478    1.000    0.002    1.000     0.
142
## Age                   0.658        1.519    0.958    3.069    0.778     0.
006
## Harvest.Stage:Age     0.027        1.897    0.616    0.479    0.776     0.
762
```

```
##                             Thripidae
##                   Pr(>Dev)       Dev  Pr(>Dev)
## (Intercept)
## Harvest.Stage        0.991     0.941    0.979
## Sex                  1.000     3.789    0.606
## Age                  0.999      1.39    0.976
## Harvest.Stage:Age    0.776     5.108    0.162
## Arguments:
##  Test statistics calculated assuming uncorrelated response (for faster com
putation)
## P-value calculated using 999 iterations via parametric resampling.
```

Dietary NMDS

As with the invertebrate data, we can visualise differences in diet via NMDS.

```
diet.mds <- metaMDS(comm = prey, distance = "jaccard", trymax=999, k=2, trace
= FALSE, autotransform = FALSE)
plot(diet.mds$points); text(diet.mds, row.names(diet.mds))
```

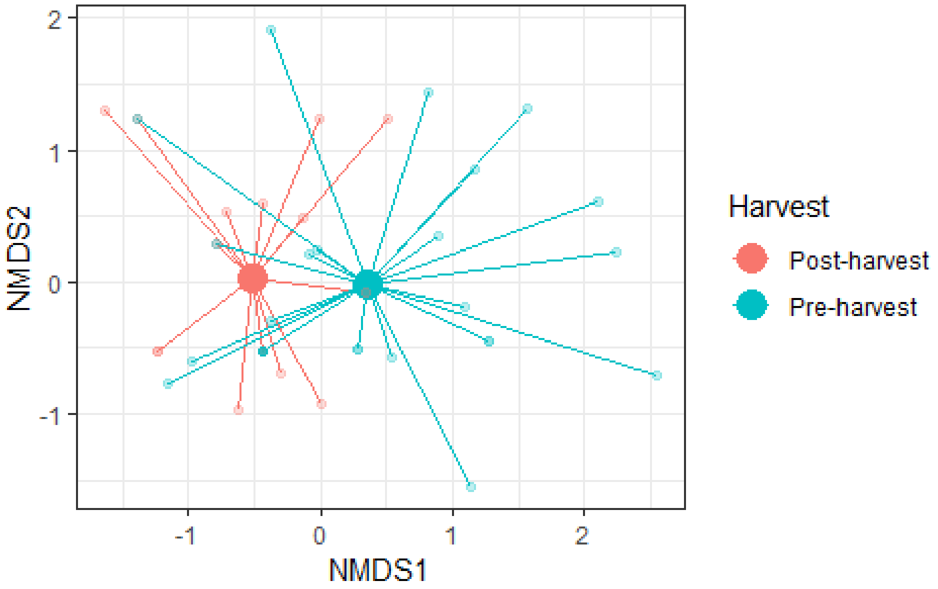


```
diet.mds$stress

## [1] 0.08244087
```

Which we can present as a spider plot.

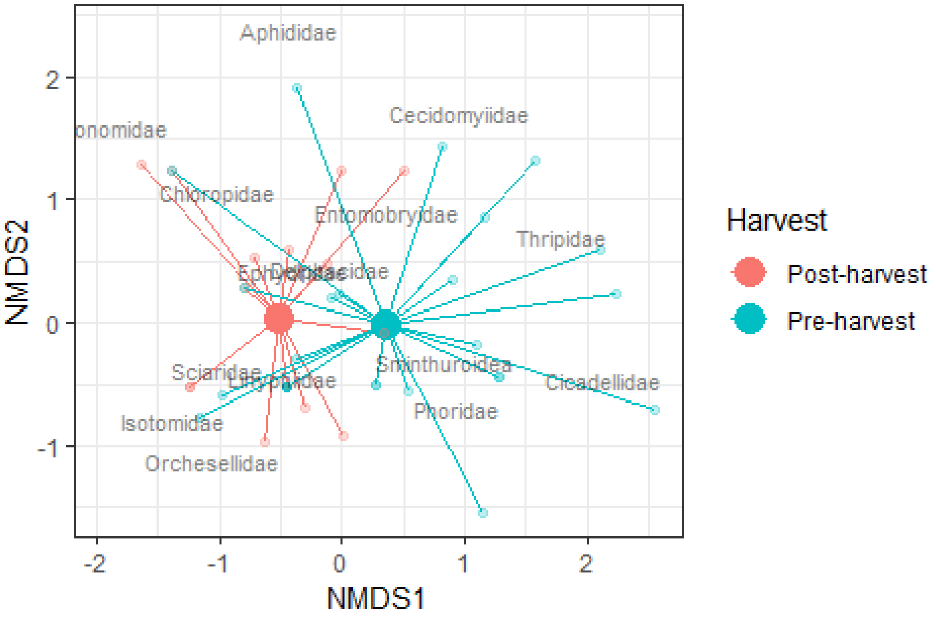
```
scrsd <- scores(diet.mds, display = 'sites')
scrsd <- cbind(as.data.frame(scrsd), Harvest = diet$Harvest.Stage)
```

```
centd <- aggregate(cbind(NMDS1, NMDS2) ~ Harvest, data = scrsd, FUN = mean)
segsd <- merge(scrsd, setNames(centd, c('Harvest', 'oNMDS1', 'oNMDS2')), by =
'Harvest', sort = FALSE)

diet.spider <- ggplot(scrsd, aes(x = NMDS1, y = NMDS2, colour = Harvest)) + s
cale_fill_brewer(2,  "Accent") + geom_segment(data = segsd, mapping = aes(xen
d = oNMDS1, yend = oNMDS2)) + geom_point(data = centd, size = 5, alpha=1) + g
eom_point(alpha=0.25) + coord_fixed() + theme_bw()

diet.spider
```



Again, we can overlay the prey families.

```
species.scoresd <- as.data.frame(scores(diet.mds, "species"))
species.scoresd$species <- rownames(species.scoresd)
names(species.scoresd)[c(1, 2)] <- c("x", "y")
species.scoresd$z <- NA

diet.spider.sp <- ggplot(species.scoresd, aes(x = x, y = y)) + theme_bw() +
  geom_text(data=species.scoresd,aes(x=x,y=y,label=species), color="black", s
ize=4,alpha=0.75, angle=0)

diet.spiderfull <- ggplot(scrsd, aes(x = NMDS1, y = NMDS2, colour = Harvest))
+ scale_fill_brewer(2,  "Accent") + geom_segment(data = segsd, mapping = aes(
xend = oNMDS1, yend = oNMDS2)) + geom_point(data = centd, size = 5, alpha=1)
+ geom_point(alpha=0.25) + coord_fixed() + theme_bw() +
  geom_text(data=species.scoresd,aes(x=x,y=y,label=species), color="black", s
ize=3,alpha=0.5, angle=0) #+ coord_equal()

diet.spiderfull
```



Prey choice analysis

To analyse prey choice, we will use 'econullnetr'. First, we need the correctly formatted data.

```
dietennr <- read.csv("ENNRDietData.csv")
invertsennr <- read.csv("ENNRInvertData.csv")
ENNRdiet.fl <- read.csv("ENNRdiet.fl.csv")
```

Now we create the model.

```
harvest.null <- generate_null_net(dietennr[,2:69], invertsennr[,2:68],
                    sims = 999, data.type = "names",
                    summary.type = "sum",
                    r.samples = invertsennr[,1],
                    c.samples = dietennr[,1],
                    r.weights = ENNRdiet.fl)

## Warning in generate_null_net(dietennr[, 2:69], invertsennr[, 2:68], sims =
999, : One or more instances detected where a consumer interacted with a
##        resource that has zero abundance in 'resources'
```

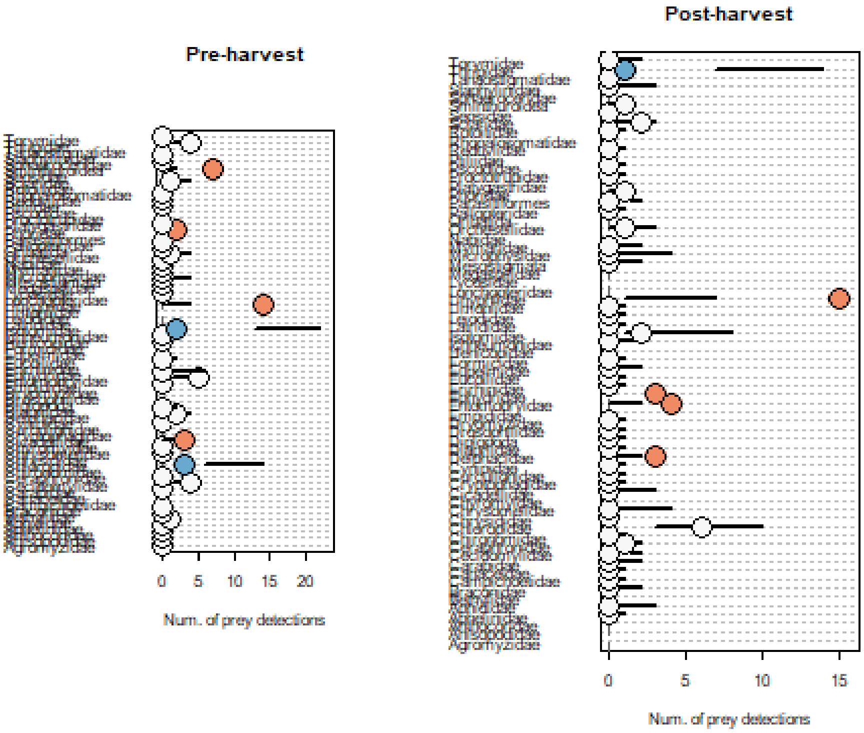
The results can be visually represented using preference plots for each category of interest.

```
par(mfrow = c(1,2))
plot_preferences(harvest.null, "Pre-harvest", signif.level = 0.95, type = "co
unts",
                 xlab = "Num. of prey detections", p.cex = 1.5, l.cex = 0.5,
lwd = 2)
```

```
## Warning in test_interactions(nullnet, signif.level = signif.level): Be car
eful
## of Type I errors due to the large number of tests
```

```
plot_preferences(harvest.null, "Post-harvest", signif.level = 0.95, type = "c
ounts",
                 xlab = "Num. of prey detections", p.cex = 1.5, l.cex = 0.5,
lwd = 2)
```

```
## Warning in test_interactions(nullnet, signif.level = signif.level): Be car
eful
## of Type I errors due to the large number of tests
```



To produce smaller plots with just the significant results, we can extract and plot those separately.

We first need to create the relevant numerical objects.

```
har.links <- test_interactions(harvest.null, signif.level = 0.95)
```

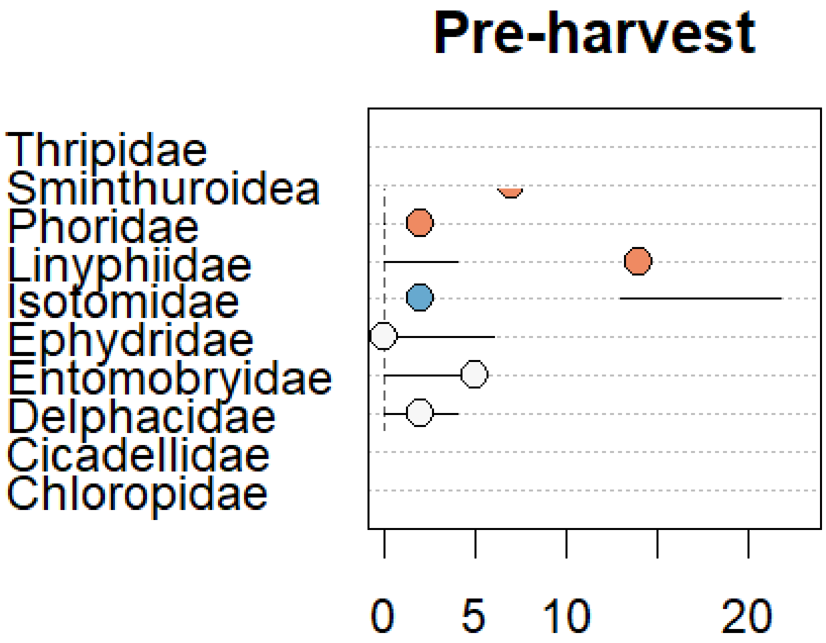
```
## Warning in test_interactions(harvest.null, signif.level = 0.95): Be carefu
l of
## Type I errors due to the large number of tests
```

Then plot, first for pre-harvest spiders.

```
eti <- test_interactions(harvest.null, signif.level = 0.95)
```

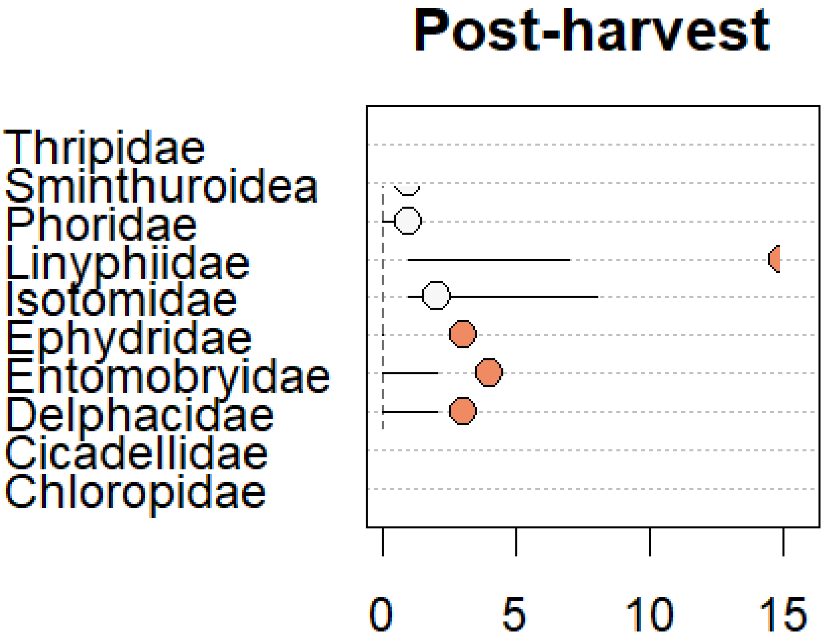
```
## Warning in test_interactions(harvest.null, signif.level = 0.95): Be carefu
l of
## Type I errors due to the large number of tests
```

```
eti <- eti[eti$Consumer == "Pre-harvest", ]
eti[, 3] <- ifelse(rowSums(eti[, 3:6]) == 0, NA, eti[, 3])
eti[, 4] <- ifelse(rowSums(eti[, 3:6]) == 0, NA, eti[, 4])
eti[, 5] <- ifelse(rowSums(eti[, 3:6]) == 0, NA, eti[, 5])
eti[, 6] <- ifelse(rowSums(eti[, 3:6]) == 0, NA, eti[, 6])

# EDIT 'eti' - to just the prey taxa that you want to show on the plot

eti <- eti[c(14, 18, 22, 28, 29, 36, 40, 52, 62, 66),]


# Set up maximum x-axis value for xlim. Add an additional 5%
emin.x <- min(eti[, 3:6], na.rm = TRUE)
emin.x <- max(0, emin.x, na.rm = TRUE)
emax.x <- max(eti[, 3:6], na.rm = TRUE)
emax.x <- emax.x * 1.05
eti$Setup <- seq(emin.x, emax.x, length.out = nrow(eti))

# Plot built up in 2 stages: i) using min and max values to set the
#   y-axis range without having to use ylim (so this can be customised
#   by the user), ii) the main dbarplot and label, and iii) the error
graphics::dotchart(eti$Setup, labels = paste(eti$Resource, " ", sep = ""),
                   col = 1, pt.cex = 0, cex = 1.5, main = "Pre-harvest")
graphics::abline(v = 0, lty = 2, col = "dimgrey")


res.col <- c("#67A9CF", "#F7F7F7", "#EF8A62")

for (i in 1:nrow(eti)){
  eval(parse(text = paste("lines(x = c(eti$Lower.", 0.95 * 100,
                          ".CL[i], eti$Upper.", 0.95 * 100,
                          ".CL[i]), y = c(i, i))", sep = "")))
  if(eti$Test[i] == "Weaker") p.col <- res.col[1]
  if(eti$Test[i] == "ns" | is.na(eti$Test[i])) p.col <- res.col[2]
  if(eti$Test[i] == "Stronger") p.col <- res.col[3]
  graphics::points(eti$Observed[i], i, pch = 21, col = "black",
                   bg = p.col, cex = 2)
}
```



Then for post-harvest spiders.

```
oti <- test_interactions(harvest.null, signif.level = 0.95)

## Warning in test_interactions(harvest.null, signif.level = 0.95): Be carefu
l of
## Type I errors due to the large number of tests

oti <- oti[oti$Consumer == "Post-harvest", ]
oti[, 3] <- ifelse(rowSums(oti[, 3:6]) == 0, NA, oti[, 3])
oti[, 4] <- ifelse(rowSums(oti[, 3:6]) == 0, NA, oti[, 4])
oti[, 5] <- ifelse(rowSums(oti[, 3:6]) == 0, NA, oti[, 5])
oti[, 6] <- ifelse(rowSums(oti[, 3:6]) == 0, NA, oti[, 6])

# EDIT 'ti' - to just the prey taxa that you want to show on the plot

oti <- oti[c(14, 18, 22, 28, 29, 36, 40, 52, 62, 66),]


# Set up maximum x-axis value for xlim. Add an additional 5%
omin.x <- min(oti[, 3:6], na.rm = TRUE)
omin.x <- max(0, omin.x, na.rm = TRUE)
omax.x <- max(oti[, 3:6], na.rm = TRUE)
omax.x <- omax.x * 1.05
oti$Setup <- seq(omin.x, omax.x, length.out = nrow(oti))

# Plot built up in 2 stages: i) using min and max values to set the
#   y-axis range without having to use ylim (so this can be customised
#   by the user), ii) the main dbarplot and label, and iii) the error
graphics::dotchart(oti$Setup, labels = paste(oti$Resource, " ", sep = ""),
                   col = 1, pt.cex = 0, cex = 1.5, main = "Post-harvest")
graphics::abline(v = 0, lty = 2, col = "dimgrey")


res.col <- c("#67A9CF", "#F7F7F7", "#EF8A62")

for (i in 1:nrow(oti)){
  eval(parse(text = paste("lines(x = c(oti$Lower.", 0.95 * 100,
                          ".CL[i], oti$Upper.", 0.95 * 100,
                          ".CL[i]), y = c(i, i))", sep = "")))
  if(oti$Test[i] == "Weaker") p.col <- res.col[1]
  if(oti$Test[i] == "ns" | is.na(oti$Test[i])) p.col <- res.col[2]
  if(oti$Test[i] == "Stronger") p.col <- res.col[3]
  graphics::points(oti$Observed[i], i, pch = 21, col = "black",
                   bg = p.col, cex = 2)
}
```



Primer comparison

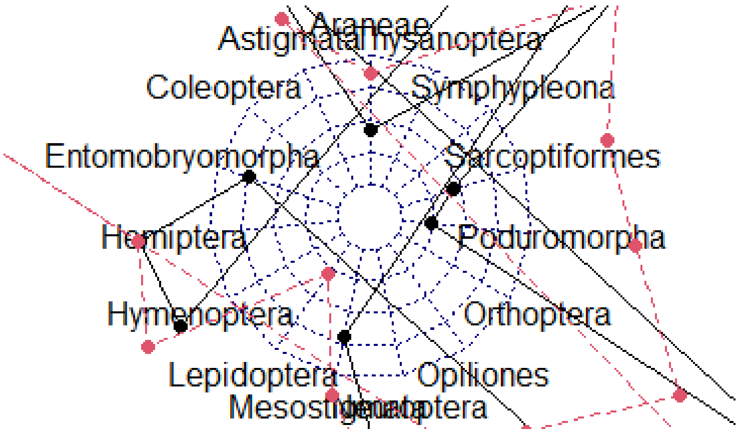
In silico primer testing

First, the data.

```
pmcomp <- read.csv("PrimerMinerInSilicoResults.csv")
rownames(pmcomp) <- pmcomp[,1]
pmcomp <- pmcomp[,-1]
```

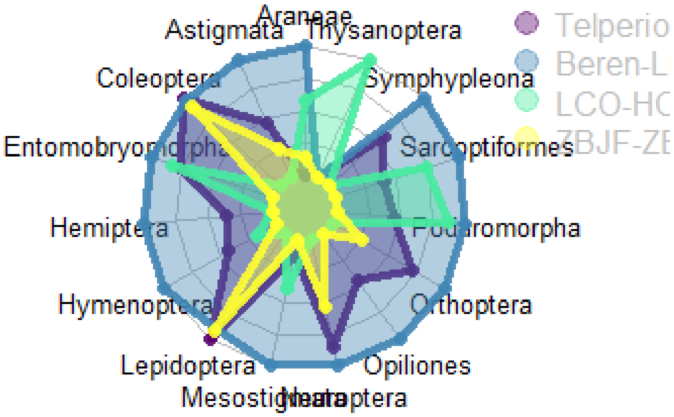
We can create a very simple radar chart to show the efficacy of primers for different taxa quite easily.

```
radarchart(pmcomp)
```



If we want something prettier though, we need to define the colours and a better legend.

```
colors_border=c( rgb(0.34,0.01,0.42,0.9), rgb(0.25,0.52,0.71,0.9) , rgb(0.27,
0.92,0.62,0.9), rgb(1.00, 1.00, 0.18, 0.9) )
colors_in=c( rgb(0.34,0.01,0.42,0.4), rgb(0.25,0.52,0.71,0.4) , rgb(0.27,0.92
,0.62,0.4), rgb(1.00, 1.00, 0.18, 0.4))

radarchart( pmcomp  , axistype=0 , maxmin=F,
            #custom polygon
            pcol=colors_border , pfcol=colors_in , plwd=4 , plty=1,
            #custom the grid
            cglcol="grey", cglty=1, axislabcol="black", cglwd=0.8,
            #custom labels
            vlcex=0.8
)
legend(x=1.2, y=1.38, legend = rownames(pmcomp), bty = "n", pch=20 , col=colo
rs_in , text.col = "grey", cex=1, pt.cex=2.5)
```



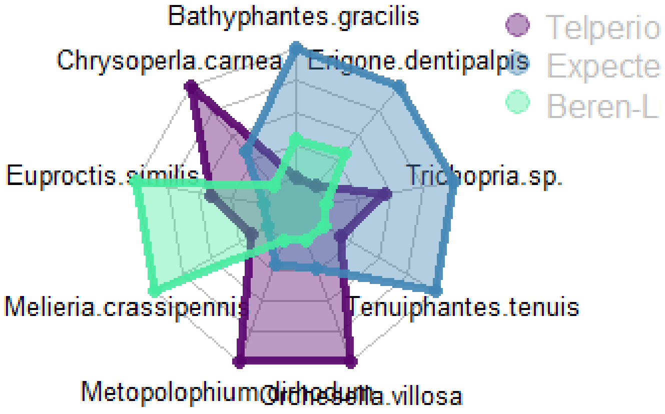
In vitro mock community testing

Each individual mock community must be loaded before plotting in radar charts to visualise bias.

```
m1 <- read.csv("Mock Community Mix 1.csv")
rownames(m1) <- m1[,1]
m1 <- m1[,-1]

m2 <- read.csv("Mock Community Mix 2.csv")
rownames(m2) <- m2[,1]
m2 <- m2[,-1]

m3 <- read.csv("Mock Community Mix 3.csv")
rownames(m3) <- m3[,1]
m3 <- m3[,-1]

m4 <- read.csv("Mock Community Mix 4.csv")
m4 <- m4[1:3,]
rownames(m4) <- m4[,1]
m4 <- m4[,-1]

m5 <- read.csv("Mock Community Mix 5.csv")
m5 <- m5[1:3,]
rownames(m5) <- m5[,1]
m5 <- m5[,-1]
```

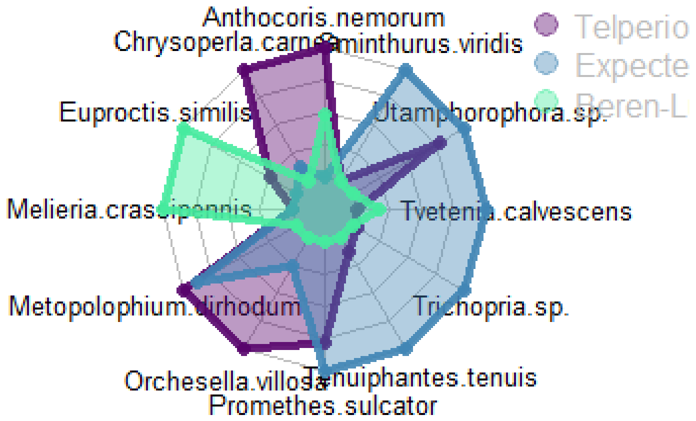
Mock community 1.

```
colors_border=c( rgb(0.34,0.01,0.42,0.9), rgb(0.25,0.52,0.71,0.9) , rgb(0.27,
0.92,0.62,0.9), rgb(1.00, 1.00, 0.18, 0.9) )
colors_in=c( rgb(0.34,0.01,0.42,0.4), rgb(0.25,0.52,0.71,0.4) , rgb(0.27,0.92
,0.62,0.4), rgb(1.00, 1.00, 0.18, 0.4))

radarchart( m1  , axistype=0 , maxmin=F,
            #custom polygon
            pcol=colors_border , pfcol=colors_in , plwd=4 , plty=1,
            #custom the grid
            cglcol="grey", cglty=1, axislabcol="black", cglwd=0.8,
            #custom labels
            vlcex=0.8
)
legend(x=1.2, y=1.38, legend = rownames(m1), bty = "n", pch=20 , col=colors_i
n , text.col = "grey", cex=1, pt.cex=2.5)
```



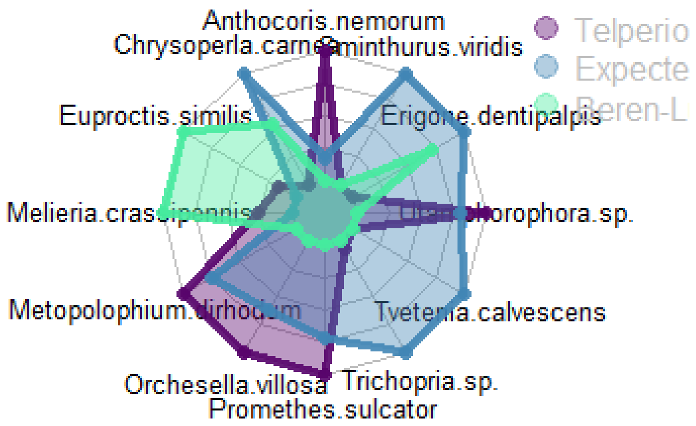
Mock community 2.

```
colors_border=c( rgb(0.34,0.01,0.42,0.9), rgb(0.25,0.52,0.71,0.9) , rgb(0.27,
0.92,0.62,0.9), rgb(1.00, 1.00, 0.18, 0.9) )
colors_in=c( rgb(0.34,0.01,0.42,0.4), rgb(0.25,0.52,0.71,0.4) , rgb(0.27,0.92
,0.62,0.4), rgb(1.00, 1.00, 0.18, 0.4))

radarchart( m2  , axistype=0 , maxmin=F,
            #custom polygon
            pcol=colors_border , pfcol=colors_in , plwd=4 , plty=1,
            #custom the grid
            cglcol="grey", cglty=1, axislabcol="black", cglwd=0.8,
            #custom labels
            vlcex=0.8
)
legend(x=1.2, y=1.38, legend = rownames(m2), bty = "n", pch=20 , col=colors_i
n , text.col = "grey", cex=1, pt.cex=2.5)
```



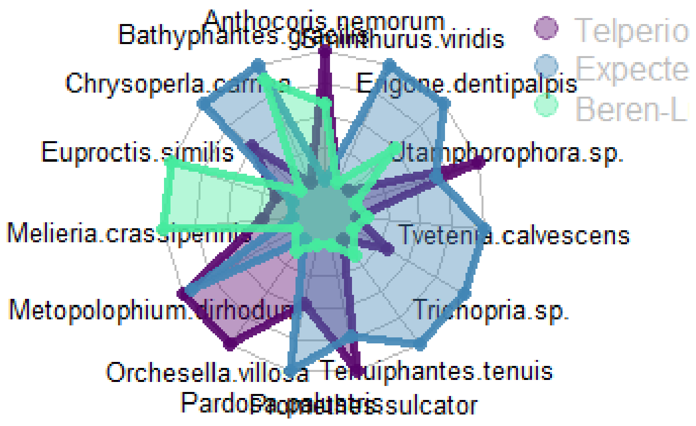
Mock community 3.

```
colors_border=c( rgb(0.34,0.01,0.42,0.9), rgb(0.25,0.52,0.71,0.9) , rgb(0.27,
0.92,0.62,0.9), rgb(1.00, 1.00, 0.18, 0.9) )
colors_in=c( rgb(0.34,0.01,0.42,0.4), rgb(0.25,0.52,0.71,0.4) , rgb(0.27,0.92
,0.62,0.4), rgb(1.00, 1.00, 0.18, 0.4))

radarchart( m3  , axistype=0 , maxmin=F,
            #custom polygon
            pcol=colors_border , pfcol=colors_in , plwd=4 , plty=1,
            #custom the grid
            cglcol="grey", cglty=1, axislabcol="black", cglwd=0.8,
            #custom labels
            vlcex=0.8
)
legend(x=1.2, y=1.38, legend = rownames(m3), bty = "n", pch=20 , col=colors_i
n , text.col = "grey", cex=1, pt.cex=2.5)
```



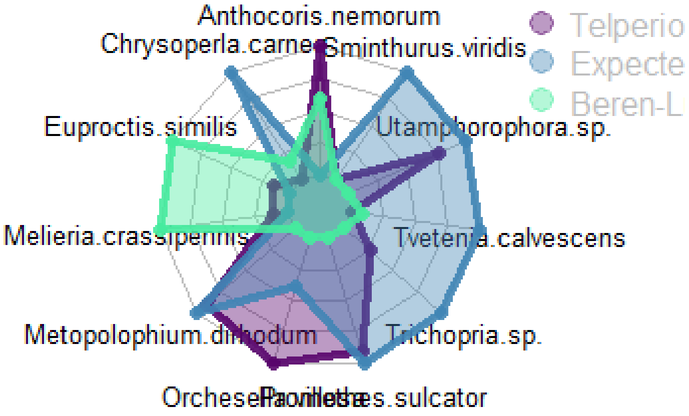
Mock community 4.

```
colors_border=c( rgb(0.34,0.01,0.42,0.9), rgb(0.25,0.52,0.71,0.9) , rgb(0.27,
0.92,0.62,0.9), rgb(1.00, 1.00, 0.18, 0.9) )
colors_in=c( rgb(0.34,0.01,0.42,0.4), rgb(0.25,0.52,0.71,0.4) , rgb(0.27,0.92
,0.62,0.4), rgb(1.00, 1.00, 0.18, 0.4))

radarchart( m4  , axistype=0 , maxmin=F,
            #custom polygon
            pcol=colors_border , pfcol=colors_in , plwd=4 , plty=1,
            #custom the grid
            cglcol="grey", cglty=1, axislabcol="black", cglwd=0.8,
            #custom labels
            vlcex=0.8
)
legend(x=1.2, y=1.38, legend = rownames(m4), bty = "n", pch=20 , col=colors_i
n , text.col = "grey", cex=1, pt.cex=2.5)
```



Mock community 5.

```
colors_border=c( rgb(0.34,0.01,0.42,0.9), rgb(0.25,0.52,0.71,0.9) , rgb(0.27,
0.92,0.62,0.9), rgb(1.00, 1.00, 0.18, 0.9) )
colors_in=c( rgb(0.34,0.01,0.42,0.4), rgb(0.25,0.52,0.71,0.4) , rgb(0.27,0.92
,0.62,0.4), rgb(1.00, 1.00, 0.18, 0.4))

radarchart( m5  , axistype=0 , maxmin=F,
            #custom polygon
            pcol=colors_border , pfcol=colors_in , plwd=4 , plty=1,
            #custom the grid
            cglcol="grey", cglty=1, axislabcol="black", cglwd=0.8,
            #custom labels
            vlcex=0.8
)
legend(x=1.2, y=1.38, legend = rownames(m5), bty = "n", pch=20 , col=colors_i
n , text.col = "grey", cex=1, pt.cex=2.5)
```


Anthocoris.nemorum
Chrysoperla.carnea
Sminthurus.viridis
Utamphorophora.sp.
Euproctis.similis
Melieria.crassipennis
Tvetenia.calvescens
Metopolophium.dirhodum
Trichopria.sp.
Orchesella.villosa
Pa[illegible]es.sulcator
Telperio
Expecte
Beren-L

Chapter 4 R Markdown

J. P. Cuff

17 November 2020

Chapter 4

Bioinformatics aggregation

In the final stages of the bioinformatic process, it is necessary to aggregate the data output so that all instances of the same taxon are together. This was achieved in R.

```
BL17agg <- read.csv("BL18_agg.csv", header = T)
Agg <- aggregate(.~Taxon, data=BL18agg, sum)
write.table(Agg, "BL18_Aggregated.csv")

TL17agg <- read.csv("TL18_agg.csv", header = T)
Agg <- aggregate(.~Taxon, data=TL18agg, sum)
write.table(Agg, "TL18_Aggregated.csv")
```

Following aggregation, the datasets for the two separate primer pairs were combined into one dietary dataset by first aggregating by sample name, then by taxon. Depending on the application, the latter was carried out at the species or family level.

```
TLBL18samagg <- read.csv("BLTL18samagg.csv", header = T)
Agg <- aggregate(.~Sample, data=TLBL18samagg, sum)
write.table(Agg, "BLTL18_SamAggregated.csv")

TLBL18specagg <- read.csv("BLTL18aggspec.csv", header = T)
Agg <- aggregate(.~Species, data=TLBL18specagg, sum)
write.table(Agg, "BLTL18_SpeciesAggregated.csv")

TLBL18aggfam <- read.csv("BLTL18aggfam.csv", header = T)
Agg <- aggregate(.~Family, data=TLBL18aggfam, sum)
write.table(Agg, "BLTL18_FamilyAggregated.csv")
```

Libraries

```
library("mvabund")
library("devtools")

## Loading required package: usethis

library("vegan")

## Loading required package: permute

##
## Attaching package: 'permute'
## The following object is masked from 'package:devtools':
##
##     check
```

```
## Loading required package: lattice
```

```
## This is vegan 2.5-6
```

```r
library("ggplot2")
library("ggthemes")
library("RColorBrewer")
library("viridis")
```

```
## Loading required package: viridisLite
```

```r
library("bipartite")
```

```
## Loading required package: sna
```

```
## Loading required package: statnet.common
```

```
##
## Attaching package: 'statnet.common'
```

```
## The following object is masked from 'package:base':
##
##     order
```

```
## Loading required package: network
```

```
## network: Classes for Relational Data
## Version 1.16.1 created on 2020-10-06.
## copyright (c) 2005, Carter T. Butts, University of California-Irvine
##                     Mark S. Handcock, University of California -- Los Ange
les
##                     David R. Hunter, Penn State University
##                     Martina Morris, University of Washington
##                     Skye Bender-deMoll, University of Washington
##  For citation information, type citation("network").
##  Type help("network-package") to get started.
```

```
## sna: Tools for Social Network Analysis
## Version 2.6 created on 2020-10-5.
## copyright (c) 2005, Carter T. Butts, University of California-Irvine
##  For citation information, type citation("sna").
##  Type help(package="sna") to get started.
```

```
##  This is bipartite 2.15.
##  For latest changes see versionlog in ?"bipartite-package". For citation s
ee: citation("bipartite").
##  Have a nice time plotting and analysing two-mode networks.
```

```
##
## Attaching package: 'bipartite'

## The following object is masked from 'package:vegan':
##
##     nullmodel

library("lme4")

## Loading required package: Matrix

library("nlme")

##
## Attaching package: 'nlme'

## The following object is masked from 'package:lme4':
##
##     lmList

## The following object is masked from 'package:sna':
##
##     gapply

library("LMERConvenienceFunctions")
library("lmtest")

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

library("DHARMa")

## This is DHARMa 0.3.3.0. For overview type '?DHARMa'. For recent changes, t
ype news(package = 'DHARMa') Note: Syntax of plotResiduals has changed in 0.3
.0, see ?plotResiduals for details

library("cooccur")
library("ggrepel")
library("spaa")

##
## Attaching package: 'spaa'

## The following object is masked from 'package:sna':
##
##     geodist

library("EcoSimR")
```

```
## Loading required package: MASS
```

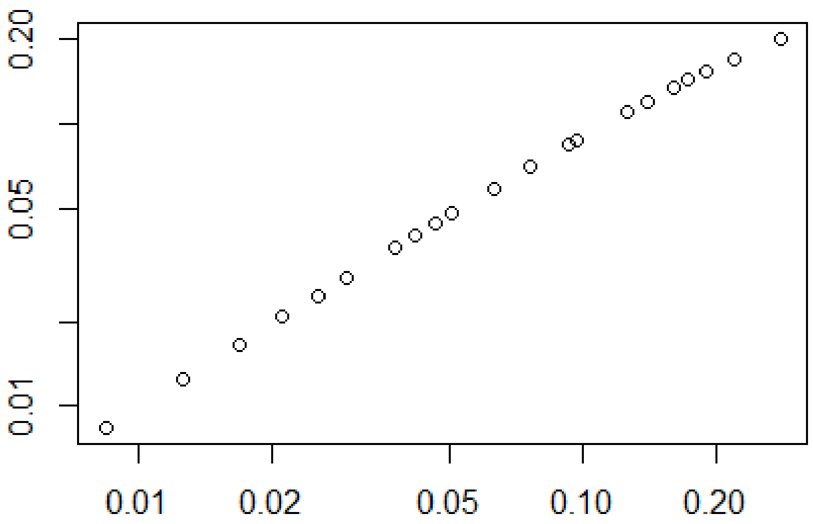
Dietary MGLM

First, the data.

```
diet <- read.csv("2018dietarydatanosingnoblank.csv")

rownames(diet) <- diet[,1]
dietprey <- diet[,25:75]
```

Next, the mvabund object.

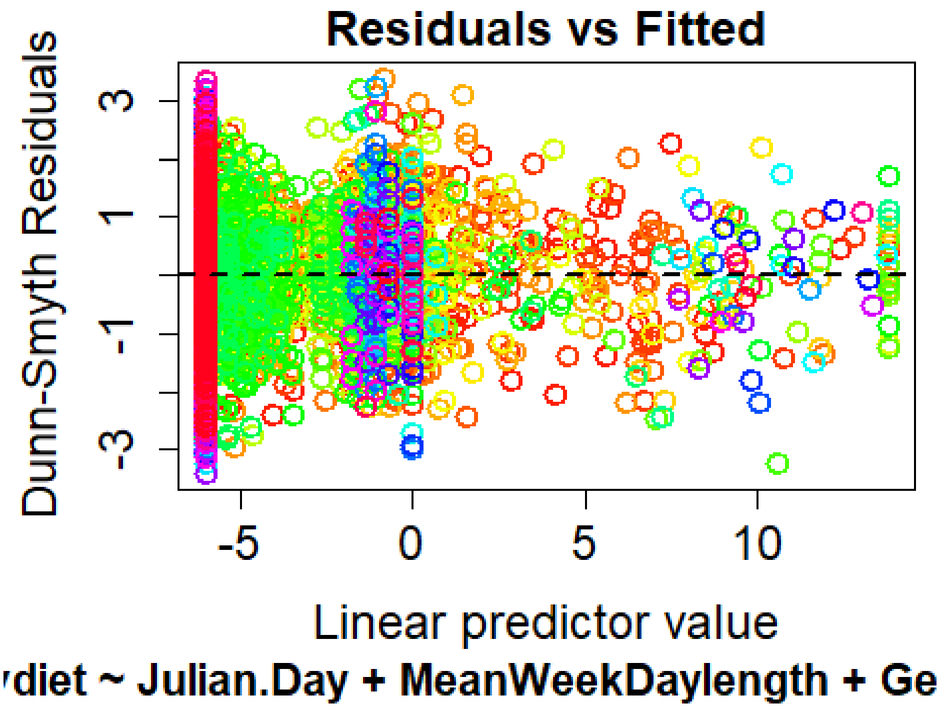
```
mvdiet <- mvabund(diet[,25:75])
meanvar.plot(mvdiet)
```



This dataset contains many variables which will be analysed against the dietary data. These can be coarsely split into categories: *Field Variables: Field + Julian.Day + Harvest.Stage + MeanWeekDaylength* Taxonomy Variables: Family + Genus + Species *Spider Variables: Maturity + Sex + Ectoparasites* Web Variables: Web.Height + Web.Area *Weather Variables: MeanWeekTemp + MeanWeekPrecipitation + MeanWeekDew + MeanWeekWind + MeanWeekPressure

With these, we can create a large model including two-way interactions between the variables.

```r
set.seed(1234)

dietm1<-manyglm(mvdiet ~ Julian.Day + MeanWeekDaylength +
                  Genus + Family +
                  Maturity + Sex + Ectoparasites +
                  MeanWeekTemp + MeanWeekPrecipitation + MeanWeekDew + MeanWe
ekWind + MeanWeekPressure +
                  Julian.Day:MeanWeekDaylength + Julian.Day:Family + Julian.D
ay:Genus + Julian.Day:Maturity + Julian.Day:Sex + Julian.Day:Ectoparasites +
                  Julian.Day:MeanWeekTemp + Julian.Day:MeanWeekPrecipitation
+ Julian.Day:MeanWeekDew + Julian.Day:MeanWeekWind + Julian.Day:MeanWeekPress
ure +
                  MeanWeekDaylength:Genus + MeanWeekDaylength:Family + MeanWe
ekDaylength:Maturity + MeanWeekDaylength:Sex +
                  MeanWeekDaylength:Ectoparasites + MeanWeekDaylength:MeanWee
kTemp + MeanWeekDaylength:MeanWeekPrecipitation +
                  MeanWeekDaylength:MeanWeekDew + MeanWeekDaylength:MeanWeekW
ind + MeanWeekDaylength:MeanWeekPressure +
                  Genus:Maturity + Genus:Sex + Genus:Ectoparasites + Genus:Me
anWeekTemp +
                  Genus:MeanWeekPrecipitation + Genus:MeanWeekDew + Genus:Mea
nWeekWind + Genus:MeanWeekPressure +
                  Family:Maturity + Family:Sex + Family:Ectoparasites + Famil
y:MeanWeekTemp +
                  Family:MeanWeekPrecipitation + Family:MeanWeekDew + Family:
MeanWeekWind + Family:MeanWeekPressure +
                  Maturity:Ectoparasites + Maturity:MeanWeekTemp + Maturity:M
eanWeekPrecipitation +
                  Maturity:MeanWeekDew + Maturity:MeanWeekWind + Maturity:Mea
nWeekPressure +
                  Ectoparasites:MeanWeekTemp + Ectoparasites:MeanWeekPrecipit
ation + Ectoparasites:MeanWeekDew +
                  Ectoparasites:MeanWeekWind + Ectoparasites:MeanWeekPressure
+
                  MeanWeekTemp:MeanWeekPrecipitation + MeanWeekTemp:MeanWeekD
ew + MeanWeekTemp:MeanWeekWind + MeanWeekTemp:MeanWeekPressure +
                  MeanWeekPrecipitation:MeanWeekDew + MeanWeekPrecipitation:M
eanWeekWind + MeanWeekPrecipitation:MeanWeekPressure +
                  MeanWeekDew:MeanWeekWind + MeanWeekDew:MeanWeekPressure + M
eanWeekWind:MeanWeekPressure
                , data=diet, family="binomial")

plot(dietm1)
```

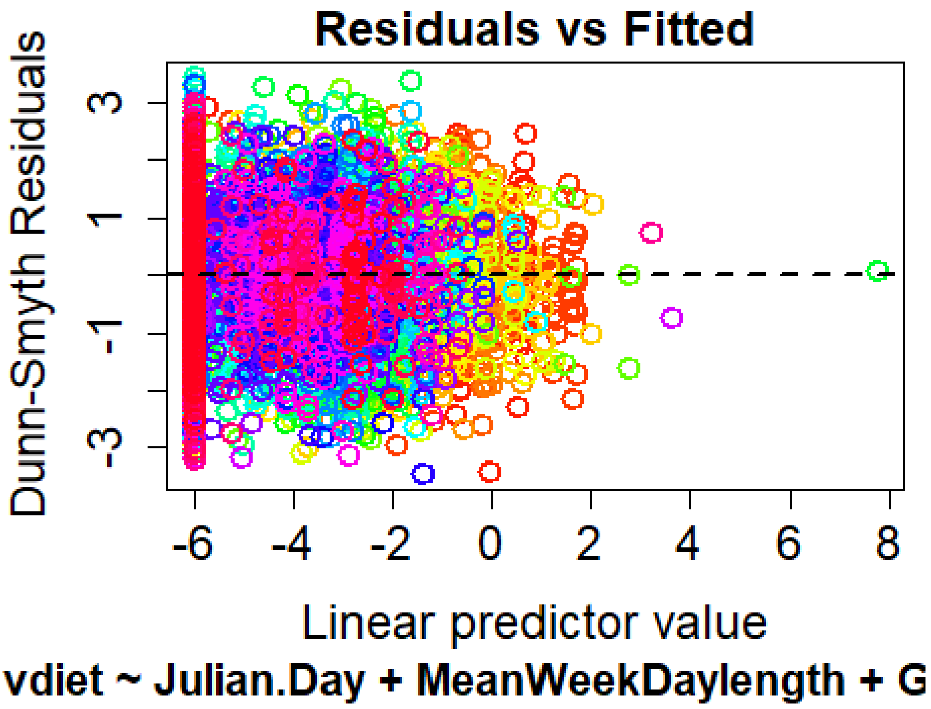


Given how unwieldy this is, we should simplify it.

```
step(dietm1, method = "ChiSq")
```

Based on this, we can create a simpler model.

```
dietm2<-manyglm(mvdiet ~ Julian.Day + MeanWeekDaylength + Genus + Maturity
                  , data=diet, family="binomial")

plot(dietm2)
```



And, finally, see the results of this analysis.

```
anova(dietm2, p.uni="adjusted", resamp="montecarlo")

## Time elapsed: 0 hr 7 min 40 sec

## Analysis of Deviance Table
##
## Model: mvdiet ~ Julian.Day + MeanWeekDaylength + Genus + Maturity
##
## Multivariate test:
##                    Res.Df Df.diff   Dev Pr(>Dev)
## (Intercept)          236
## Julian.Day           235       1 150.8    0.001 ***
## MeanWeekDaylength    234       1 333.8    0.001 ***
## Genus                230       4 489.2    0.001 ***
## Maturity             229       1 124.6    0.001 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Univariate Tests:
##                    Acrodactyla.degener          Aeolothrips.intermedius
##                                   Dev Pr(>Dev)                     Dev Pr(
>Dev)
## (Intercept)
## Julian.Day                       0.51    1.000                   0.698
1.000
```

```
## MeanWeekDaylength              3.868    0.966                   7.738
0.425
## Genus                          3.475    1.000                   3.797
1.000
## Maturity                       1.026    0.998                   2.167
0.993
##                    Agyneta.rurestris         Amischa.sp.         Anagrus.
sp.
##                               Dev Pr(>Dev)         Dev Pr(>Dev)
Dev
## (Intercept)
## Julian.Day                   0.964    1.000       0.082    1.000       0.
205
## MeanWeekDaylength            0.551    1.000       4.437    0.946       8.
065
## Genus                        4.614    0.999           0    1.000       2.
043
## Maturity                     1.506    0.998      -0.002    0.999       4.
452
##                             Anaphothrips.obscurus
##                    Pr(>Dev)                  Dev Pr(>Dev)
## (Intercept)
## Julian.Day            1.000                6.008    0.539
## MeanWeekDaylength     0.410                72.79    0.001
## Genus                 1.000               39.293    0.001
## Maturity              0.826                7.114    0.327
##                    Anotylus.tetracarinatus          Aphelinus.sp.
##                                     Dev Pr(>Dev)           Dev Pr(>Dev)
## (Intercept)
## Julian.Day                        2.141    0.996         3.648    0.902
## MeanWeekDaylength                10.638    0.126        11.139    0.090
## Genus                             6.281    0.969         4.315    0.999
## Maturity                              0    0.999         0.208    0.999
##                    Aphidius.sp.          Bourletiellidae.sp.
##                             Dev Pr(>Dev)                 Dev Pr(>Dev)
## (Intercept)
## Julian.Day                6.559    0.482              6.523    0.483
## MeanWeekDaylength         3.364    0.988              10.03    0.185
## Genus                     5.664    0.987             21.274    0.005
## Maturity                  1.689    0.995              8.558    0.167
##                    Bradysia.urticae         Camptocladius.stercorarius
##                                 Dev Pr(>Dev)                         Dev Pr(
>Dev)
## (Intercept)
## Julian.Day                    1.209    1.000                       1.291
0.999
## MeanWeekDaylength             0.533    1.000                       9.083
0.288
## Genus                          7.72    0.882                       6.851
0.924
```

```
## Maturity                     1.021    0.998                       0.16    
0.999
##                   Cecidomyiidae.sp.          Centromerita.bicolor
##                                 Dev Pr(>Dev)                  Dev Pr(>Dev)
## (Intercept)
## Julian.Day                     5.33    0.632                0.082    1.000
## MeanWeekDaylength             0.035    1.000                6.526    0.644
## Genus                         23.88    0.001                1.308    1.000
## Maturity                      1.953    0.995                1.439    0.998
##                   Chrysoperla.sp.          Copidosoma.floridanum
##                               Dev Pr(>Dev)                   Dev Pr(>Dev)
## (Intercept)
## Julian.Day                   0.082    1.000                 0.689    1.000
## MeanWeekDaylength            0.229    1.000                 10.13    0.174
## Genus                        3.386    1.000                 0.719    1.000
## Maturity                     1.027    0.998                 2.171    0.993
##                   Coproica.ferruginata          Corynoptera.sp.
##                                    Dev Pr(>Dev)             Dev Pr(>Dev)
## (Intercept)
## Julian.Day                         1.48    0.999           0.029    1.000
## MeanWeekDaylength                 7.293    0.505           0.367    1.000
## Genus                             4.407    0.999           7.849    0.882
## Maturity                          0.003    0.999           0.481    0.999
##                   Elachiptera.decipens          Entomobryidae.sp.
##                                    Dev Pr(>Dev)               Dev Pr(>Dev)
## (Intercept)
## Julian.Day                        0.031    1.000             0.403    1.000
## MeanWeekDaylength                 3.703    0.971             6.318    0.720
## Genus                             5.056    0.999              7.77    0.882
## Maturity                          0.095    0.999              3.97    0.887
##                   Erigone.dentipalpis          Eupodidae.sp.
##                                   Dev Pr(>Dev)         Dev Pr(>Dev)
## (Intercept)
## Julian.Day                       0.301    1.000       0.229    1.000
## MeanWeekDaylength                3.971    0.962      10.519    0.128
## Genus                           11.105    0.376      36.459    0.001
## Maturity                         3.248    0.949      15.716    0.005
##                   Frankliniella.tenuicornis          Hemiptera.sp.
##                                         Dev Pr(>Dev)          Dev Pr(>Dev
)
## (Intercept)
## Julian.Day                             1.879    0.999        9.846    0.07
4
## MeanWeekDaylength                     17.956    0.003        0.057    1.00
0
## Genus                                 21.964    0.002        4.996    0.99
9
## Maturity                               1.427    0.998        3.593    0.93
2
##                   Hypogastrura.viatica          Isotomurus.sp.
##                                    Dev Pr(>Dev)            Dev Pr(>Dev)
## (Intercept)
## Julian.Day                       7.094    0.342         11.981    0.029
## MeanWeekDaylength               13.369    0.023              0    1.000
## Genus                                0    1.000          7.839    0.882
## Maturity                        -0.016    0.999          0.372    0.999
##                   Javesella.sp.          Limothrips.denticornis
##                             Dev Pr(>Dev)                    Dev Pr(>Dev)
## (Intercept)
## Julian.Day                0.795    1.000                  2.119    0.996
## MeanWeekDaylength         1.816    1.000                 36.669    0.001
## Genus                     2.724    1.000                  14.25    0.106
## Maturity                  1.307    0.998                 11.526    0.044
##                   Macrosteles.sp.          Metopina.galeata
##                               Dev Pr(>Dev)              Dev Pr(>Dev)
## (Intercept)
## Julian.Day                   1.38    0.999            0.079    1.000
## MeanWeekDaylength           0.093    1.000            0.846    1.000
## Genus                       3.058    1.000             4.79    0.999
## Maturity                    2.236    0.993            1.972    0.995
##                   Micromus.variegatus          Neriene.montana
##                                   Dev Pr(>Dev)             Dev Pr(>Dev)
## (Intercept)
## Julian.Day                      0.098    1.000           1.858    0.999
## MeanWeekDaylength               6.032    0.779           0.403    1.000
## Genus                           2.075    1.000           5.037    0.999
## Maturity                         2.44    0.992           0.217    0.999
##                   Nothodelphax.sp.          Oscinella.sp.
##                                Dev Pr(>Dev)          Dev Pr(>Dev)
## (Intercept)
## Julian.Day                   0.017    1.000        12.567    0.025
## MeanWeekDaylength            0.167    1.000         5.631    0.818
## Genus                        7.434    0.896        44.375    0.001
## Maturity                     2.924    0.970         1.476    0.998
##                   Pardosa.amentata          Pardosa.lugubris
##                                Dev Pr(>Dev)              Dev Pr(>Dev)
## (Intercept)
## Julian.Day                    1.04    1.000            0.029    1.000
## MeanWeekDaylength            0.248    1.000            0.459    1.000
## Genus                        3.764    1.000                0    1.000
## Maturity                     0.619    0.998            0.365    0.999
##                   Pardosa.pullata          Reticulitermes.lucifugus.lucifu
gus
##                               Dev Pr(>Dev)
Dev
## (Intercept)
## Julian.Day                  1.302    0.999                              0.
251
## MeanWeekDaylength           0.237    1.000                              8.
022
```

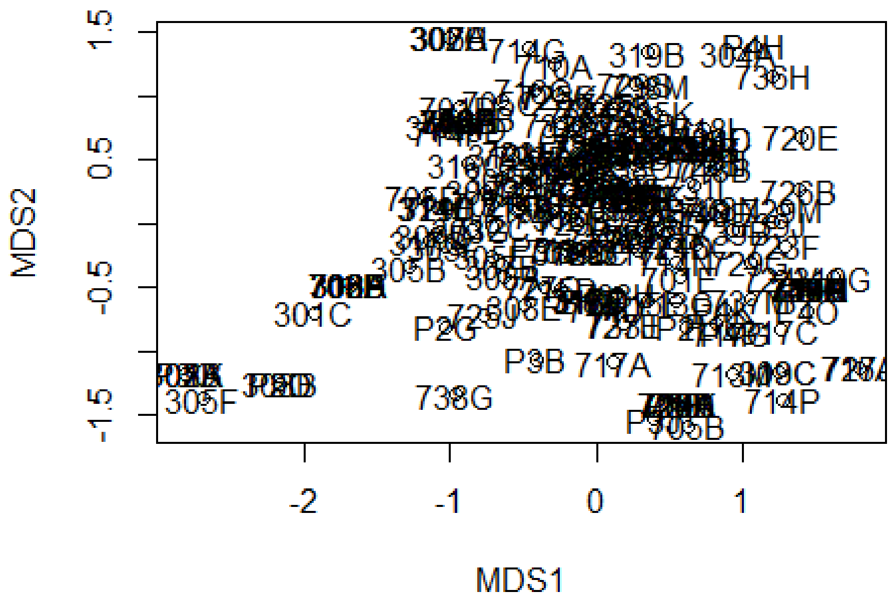
```
## Genus                       4.717    0.999                             65.
065
## Maturity                    2.014    0.995                              0.
942
##                            Rhopalosiphum.sp.          Scaptomyza.pallida
##                   Pr(>Dev)               Dev Pr(>Dev)                Dev
## (Intercept)
## Julian.Day           1.000             0.394    1.000              0.182
## MeanWeekDaylength    0.410             2.013    1.000              0.606
## Genus                0.001             7.192    0.907              2.587
## Maturity             0.998             0.153    0.999              2.293
##                            Scatopsciara.atomaria          Sipha.sp.
##                   Pr(>Dev)                   Dev Pr(>Dev)       Dev Pr(>De
v)
## (Intercept)
## Julian.Day           1.000                 0.191    1.000     1.106    1.0
00
## MeanWeekDaylength    1.000                   1.1    1.000     1.411    1.0
00
## Genus                1.000                 9.463    0.671     4.081    1.0
00
## Maturity             0.993                 1.712    0.995      1.05    0.9
98
##                   Sitobion.sp.          Sminthurinus.aureus
##                            Dev Pr(>Dev)                 Dev Pr(>Dev)
## (Intercept)
## Julian.Day               0.634    1.000              14.217    0.020
## MeanWeekDaylength         4.61    0.939               0.759    1.000
## Genus                   17.555    0.016               5.783    0.987
## Maturity                 0.764    0.998               0.004    0.999
##                   Sminthurinus.elegans          Sminthurus.viridis
##                                    Dev Pr(>Dev)                Dev Pr(>Dev
)
## (Intercept)
## Julian.Day                       8.087    0.267             33.455    0.00
1
## MeanWeekDaylength               10.203    0.149             10.679    0.12
6
## Genus                            9.094    0.719             12.205    0.25
4
## Maturity                          0.04    0.999              3.483    0.93
2
##                   Stilbus.testaceus          Tachyporus.chrysomelinus
##                                 Dev Pr(>Dev)                      Dev Pr(>
Dev)
## (Intercept)
## Julian.Day                    0.614    1.000                    0.231    1
.000
## MeanWeekDaylength             0.719    1.000                    9.167    0
.269
```

```
## Genus                         6.852    0.924                     1.95    1
.000
## Maturity                      5.404    0.580                     0.49    0
.999
##                   Tachyporus.hypnorum          Tenuiphantes.tenuis
##                                   Dev Pr(>Dev)                 Dev Pr(>Dev
)
## (Intercept)
## Julian.Day                      0.037    1.000               0.702    1.00
0
## MeanWeekDaylength               1.983    1.000               0.045    1.00
0
## Genus                           3.355    1.000               7.421    0.89
6
## Maturity                        3.076    0.965                   0    0.99
9
##                   Trombidiidae.sp.
##                                Dev Pr(>Dev)
## (Intercept)
## Julian.Day                   0.091    1.000
## MeanWeekDaylength             7.15    0.544
## Genus                        2.275    1.000
## Maturity                    14.754    0.007
## Arguments:
##  Test statistics calculated assuming uncorrelated response (for faster com
putation)
## P-value calculated using 999 iterations via parametric resampling.
```

Dietary NMDS

As in Chapter 3, we can visualise dietary differences using NMDS.

```
dietnmds <- read.csv("2018dietarydatanosingnoblanknoout.csv")
rownames(dietnmds) <- dietnmds[,1]
dietpreynmds <- dietnmds[,25:75]

diet.mds <- metaMDS(comm = dietpreynmds, distance = "jaccard", trymax=999, k=
2, trace = FALSE, autotransform = FALSE)
plot(diet.mds$points); text(diet.mds, row.names(diet.mds))
```



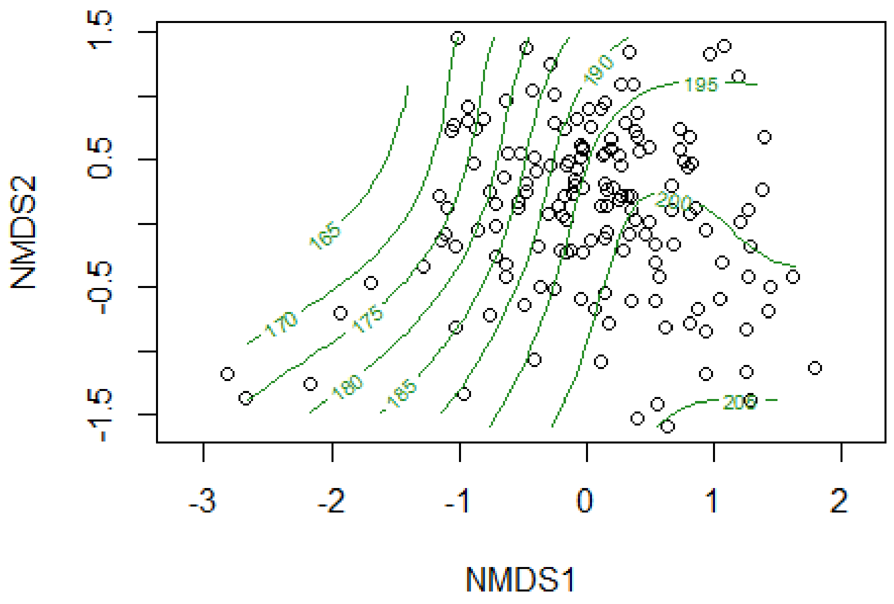
```
diet.mds$stress

## [1] 0.08754908
```

Using this, we can begin to overlay information based on our significant MGLM results.

Julian day ordisurf

We can create a surf plot to show how diets vary across time.

```
juliandiet <- ordisurf(diet.mds, dietnmds$Julian.Day, main="", col="forestgre
en")
```



And we can make this prettier.

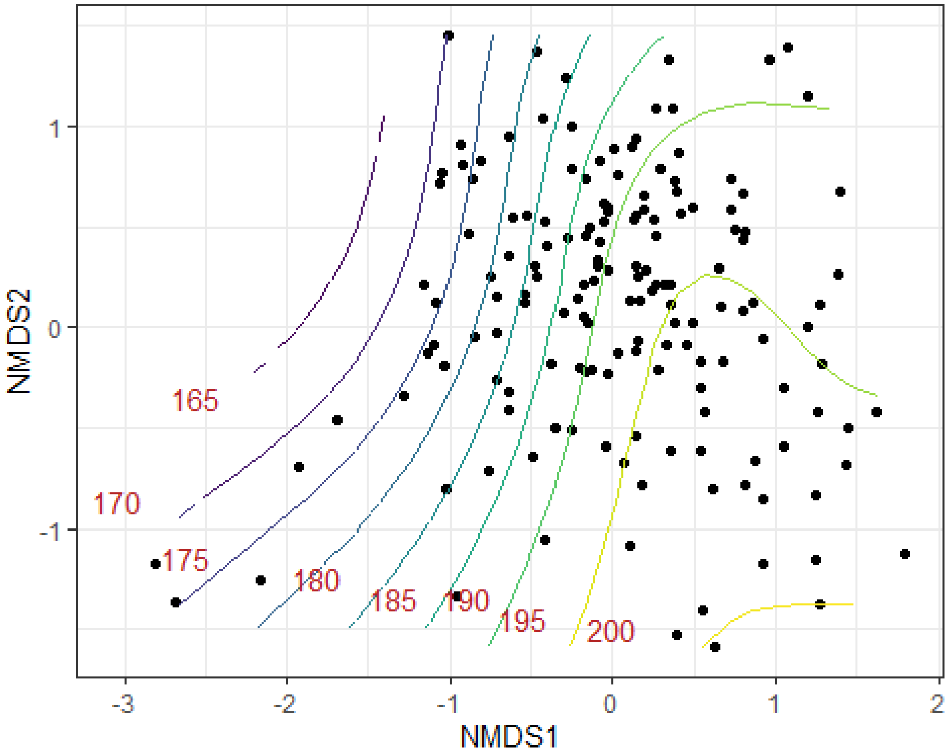
```
species.scores <- as.data.frame(scores(diet.mds, "species"))
species.scores$species <- rownames(species.scores)
names(species.scores)[c(1, 2)] <- c("x", "y")
species.scores$z <- NA

data.scores <- as.data.frame(scores(diet.mds))
data.scores$site <- rownames(data.scores)
data.scores$Julian.Day <- dietnmds$Julian.Day
head(data.scores)

##          NMDS1      NMDS2 site Julian.Day
## 301C -1.926980 -0.6937867 301C        121
## 302A -1.692315 -0.4648923 302A        121
## 302B -2.812778 -1.1754041 302B        121
## 302C -1.008527  1.4531896 302C        121
## 303B -2.163494 -1.2513534 303B        131
## 303F -1.033711 -0.1861719 303F        131

head(species.scores)

##                                   x          y                  species  z
## Acrodactyla.degener     -0.10784704 -0.1121951     Acrodactyla.degener NA
## Aeolothrips.intermedius  0.27131742  0.3855255 Aeolothrips.intermedius NA
## Agyneta.rurestris        0.05067865  0.0979651       Agyneta.rurestris NA
## Amischa.sp.              1.35695880 -0.2173725             Amischa.sp. NA
## Anagrus.sp.             -1.26314167  0.8651043             Anagrus.sp. NA
## Anaphothrips.obscurus    0.12710907  0.5273901   Anaphothrips.obscurus NA
```

```
extract.xyz <- function(obj) {
  xy <- expand.grid(x = obj$grid$x, y = obj$grid$y)
  xyz <- cbind(xy, c(obj$grid$z))
  names(xyz) <- c("x", "y", "z")
  return(xyz)
}

juliandiet.contour.vals <- extract.xyz(obj = juliandiet)
head(juliandiet.contour.vals)
```

```
##           x         y  z
## 1 -2.812778 -1.586096 NA
## 2 -2.659321 -1.586096 NA
## 3 -2.505864 -1.586096 NA
## 4 -2.352407 -1.586096 NA
## 5 -2.198950 -1.586096 NA
## 6 -2.045493 -1.586096 NA
```

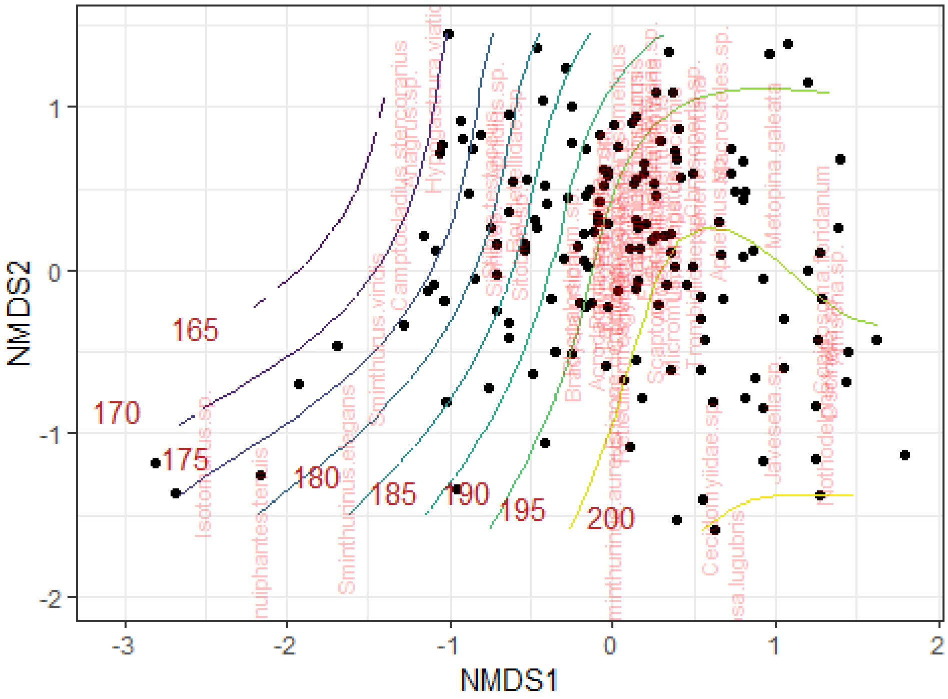
```
p <- ggplot(data=juliandiet.contour.vals, aes(x,y,z=z)) + geom_point(data=dat
a.scores, aes(x=NMDS1, y=NMDS2), inherit.aes = FALSE) + stat_contour(aes(colo
ur = ..level..), colour = viridis(306)) + theme_bw() +
  labs(x = "NMDS1", y = "NMDS2") +
  theme(panel.border = element_rect(fill = NA)) #+
  #geom_text(data=species.scores,aes(x=x,y=y,label=species), color="red", siz
e=2,alpha=0.5, angle=90) #+ coord_equal()
#geom_point(data=species.scores, aes(x=x, y=y), size=3)

labelz <- data.frame(x = c(-2.56, -3.05, -2.62, -1.81, -1.35, -0.90, -0.55, -
0.00),
                     y = c(-0.35, -0.86, -1.14, -1.25, -1.35, -1.35, -1.45, -
1.50),
                     z = NA,
                 labels = c("165", "170", "175", "180", "185", "190", "195",
"200"))

pt <- p + geom_text(data = labelz, aes(x = x, y = y, label = labels), angle =
0, color = "firebrick",
                      size = 4) + labs(x = "NMDS1", y = "NMDS2")

pt
```

```
## Warning: Removed 209 rows containing non-finite values (stat_contour).
```



As before, we can overlay the prey species.

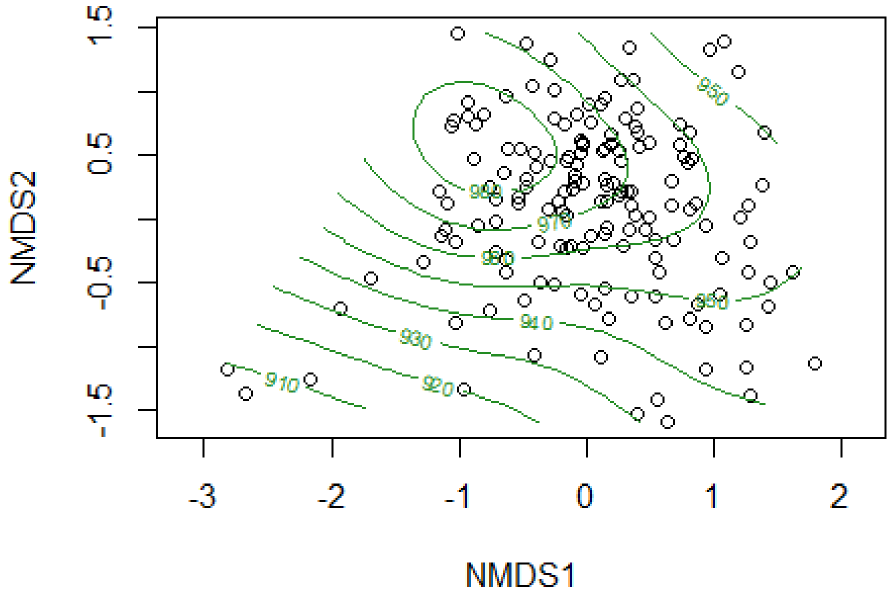
```
pts <- pt +
geom_text(data=species.scores,aes(x=x,y=y,label=species), color="red", size=3
,alpha=0.25, angle=90) + coord_equal() #+
#geom_point(data=species.scores, aes(x=x, y=y), size=3)

pts

## Warning: Removed 209 rows containing non-finite values (stat_contour).

## Warning: Removed 5 rows containing missing values (geom_text).
```



Daylength ordisurf

We can do the same for changes in daylength.

```
daylengthdiet <- ordisurf(diet.mds, dietnmds$MeanWeekDaylength, main="", col=
"forestgreen")
```



And, again, prettier.

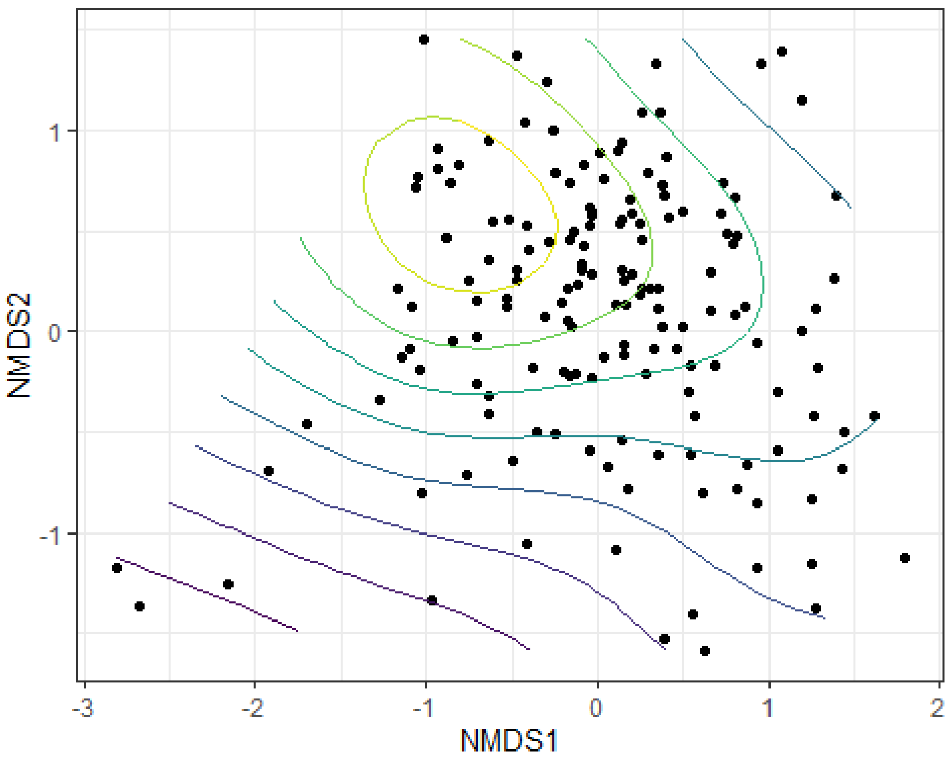
```
species.scores <- as.data.frame(scores(diet.mds, "species"))
species.scores$species <- rownames(species.scores)
names(species.scores)[c(1, 2)] <- c("x", "y")
species.scores$z <- NA

data.scores <- as.data.frame(scores(diet.mds))
data.scores$site <- rownames(data.scores)
data.scores$MeanWeekDaylength <- dietnmds$MeanWeekDaylength
head(data.scores)

##           NMDS1      NMDS2  site MeanWeekDaylength
## 301C -1.926980 -0.6937867 301C          878.1429
## 302A -1.692315 -0.4648923 302A          878.1429
## 302B -2.812778 -1.1754041 302B          878.1429
## 302C -1.008527  1.4531896 302C          878.1429
## 303B -2.163494 -1.2513534 303B          912.5714
## 303F -1.033711 -0.1861719 303F          912.5714

head(species.scores)

##                                   x          y                  species  z
## Acrodactyla.degener     -0.10784704 -0.1121951      Acrodactyla.degener NA
## Aeolothrips.intermedius  0.27131742  0.3855255 Aeolothrips.intermedius NA
## Agyneta.rurestris        0.05067865  0.0979651       Agyneta.rurestris NA
## Amischa.sp.              1.35695880 -0.2173725             Amischa.sp. NA
## Anagrus.sp.             -1.26314167  0.8651043             Anagrus.sp. NA
## Anaphothrips.obscurus    0.12710907  0.5273901   Anaphothrips.obscurus NA
```

```
extract.xyz <- function(obj) {
  xy <- expand.grid(x = obj$grid$x, y = obj$grid$y)
  xyz <- cbind(xy, c(obj$grid$z))
  names(xyz) <- c("x", "y", "z")
  return(xyz)
}

daylengthdiet.contour.vals <- extract.xyz(obj = daylengthdiet)
head(daylengthdiet.contour.vals)
```

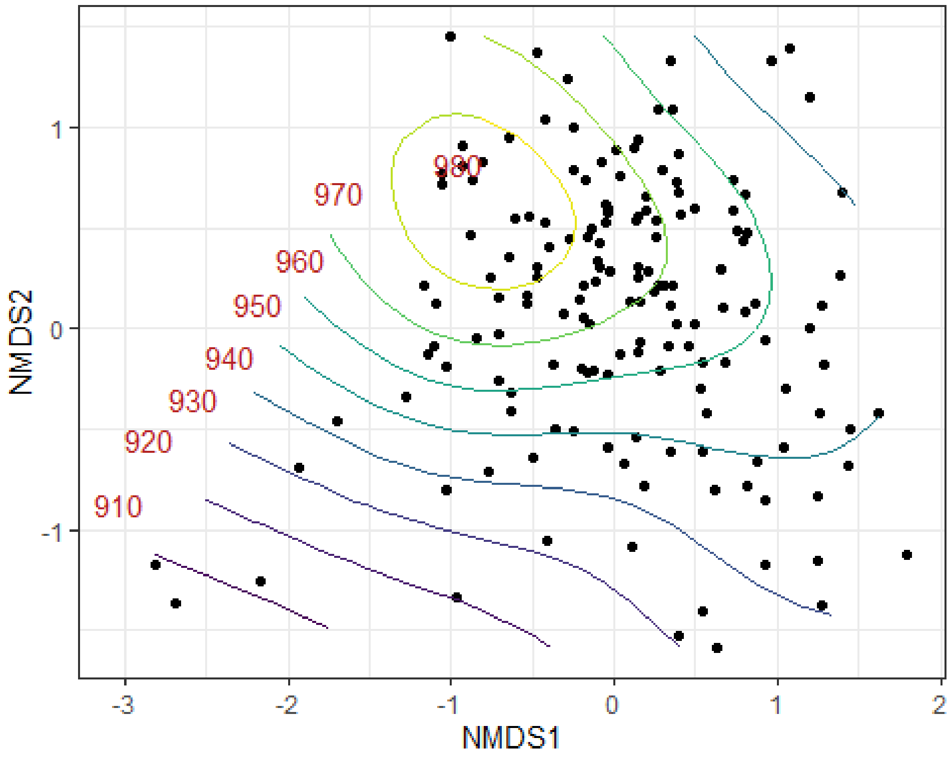
```
##           x         y  z
## 1 -2.812778 -1.586096 NA
## 2 -2.659321 -1.586096 NA
## 3 -2.505864 -1.586096 NA
## 4 -2.352407 -1.586096 NA
## 5 -2.198950 -1.586096 NA
## 6 -2.045493 -1.586096 NA
```

```
p <- ggplot(data=daylengthdiet.contour.vals, aes(x,y,z=z)) + geom_point(data=
data.scores, aes(x=NMDS1, y=NMDS2), inherit.aes = FALSE) + stat_contour(aes(c
olour = ..level..), colour = viridis(270)) + theme_bw() +
  labs(x = "NMDS1", y = "NMDS2") +
  theme(panel.border = element_rect(fill = NA)) #+
#geom_text(data=species.scores,aes(x=x,y=y,label=species), color="red", size=
2,alpha=0.5, angle=90) #+ coord_equal()
#geom_point(data=species.scores, aes(x=x, y=y), size=3)

p
```

```
## Warning: Removed 209 rows containing non-finite values (stat_contour).
```

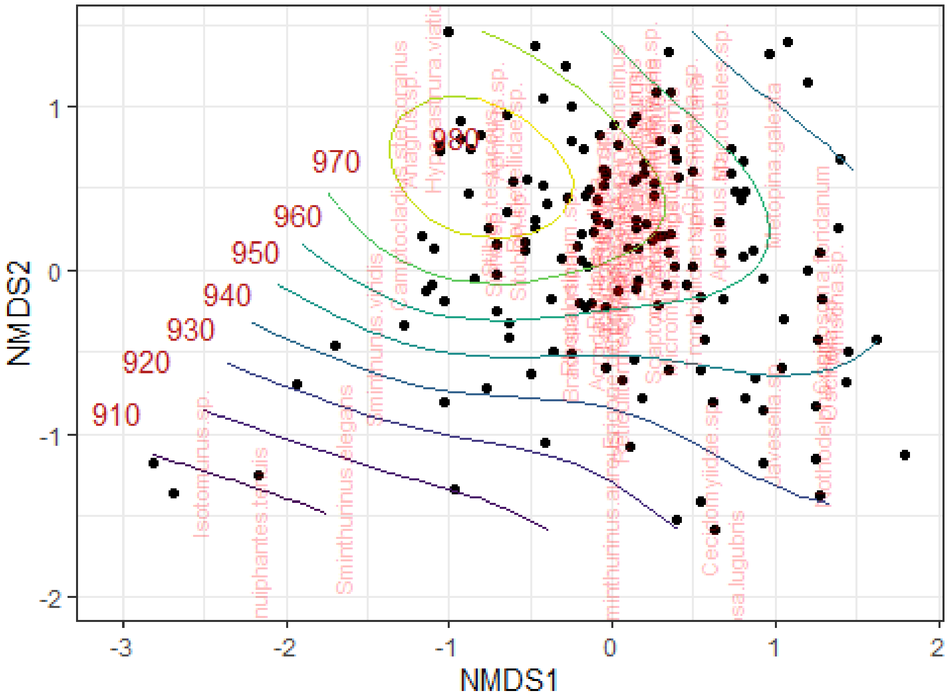


```
labelz <- data.frame(x = c(-3.04, -2.85, -2.57, -2.35, -2.18, -1.92, -1.68, -
0.95),
                     y = c(-0.87, -0.55, -0.35, -0.14, 0.120, 0.350, 0.680, 0
.820),
                     z = NA,
                 labels = c("910", "920", "930", "940", "950", "960", "970",
"980"))

pt <- p + geom_text(data = labelz, aes(x = x, y = y, label = labels), angle =
0, color = "firebrick",
                    size = 4) + labs(x = "NMDS1", y = "NMDS2")

pt

## Warning: Removed 209 rows containing non-finite values (stat_contour).
```



And with prey species labels.

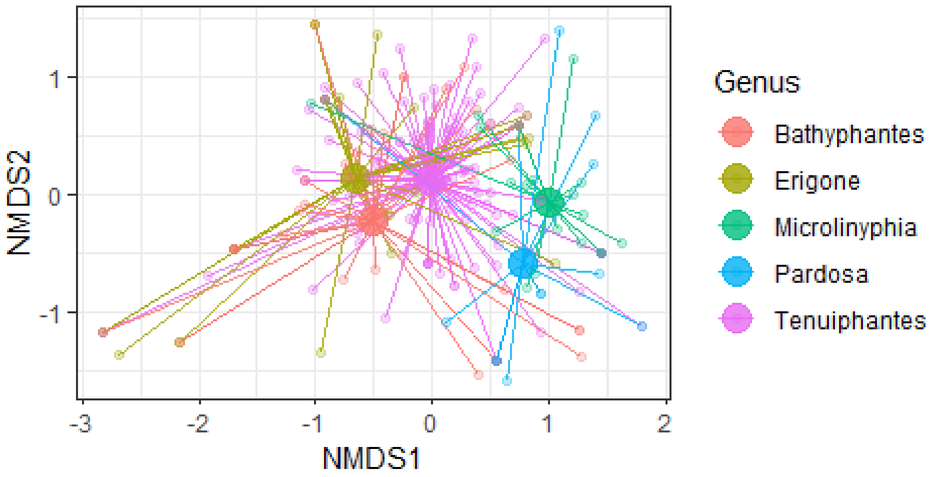
```
pts <- pt +
  geom_text(data=species.scores,aes(x=x,y=y,label=species), color="red", size
=3,alpha=0.25, angle=90) + coord_equal() #+
#geom_point(data=species.scores, aes(x=x, y=y), size=3)

pts

## Warning: Removed 209 rows containing non-finite values (stat_contour).

## Warning: Removed 5 rows containing missing values (geom_text).
```



Genus Ordispider

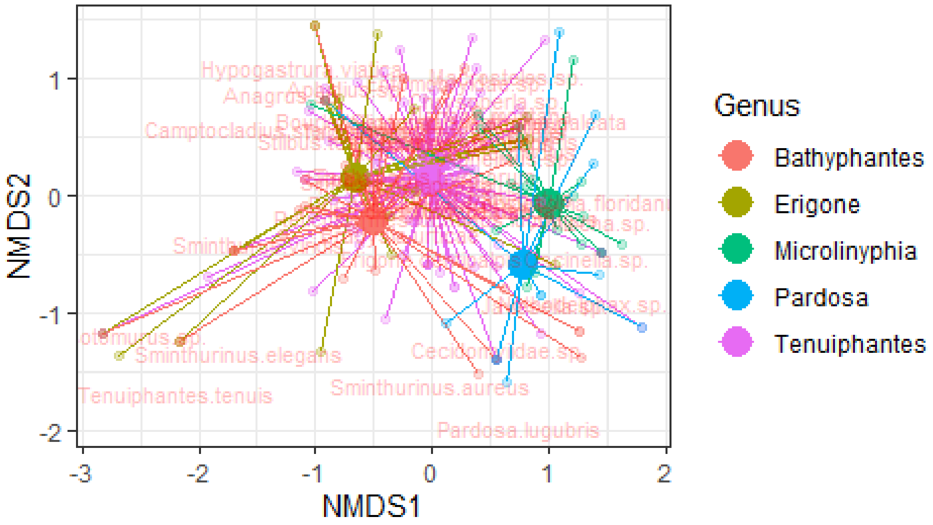
For categorical variables, we can use spider plots as in Chapter 3.

```
scrs <- scores(diet.mds, display = 'sites')
scrs <- cbind(as.data.frame(scrs), Genus = dietnmds$Genus)
cent <- aggregate(cbind(NMDS1, NMDS2) ~ Genus, data = scrs, FUN = mean)
segs <- merge(scrs, setNames(cent, c('Genus', 'oNMDS1', 'oNMDS2')), by = 'Genus', sort = FALSE)

genusspiplot <- ggplot(scrs, aes(x = NMDS1, y = NMDS2, colour = Genus)) + scale_fill_brewer(6,  "Accent") + geom_segment(data = segs, mapping = aes(xend = oNMDS1, yend = oNMDS2), alpha=0.8) + geom_point(data = cent, size = 5, alpha = 0.8) + geom_point(alpha=0.25) + coord_fixed() + theme_bw()
genusspiplot
```



Again, prey species labels can be added.

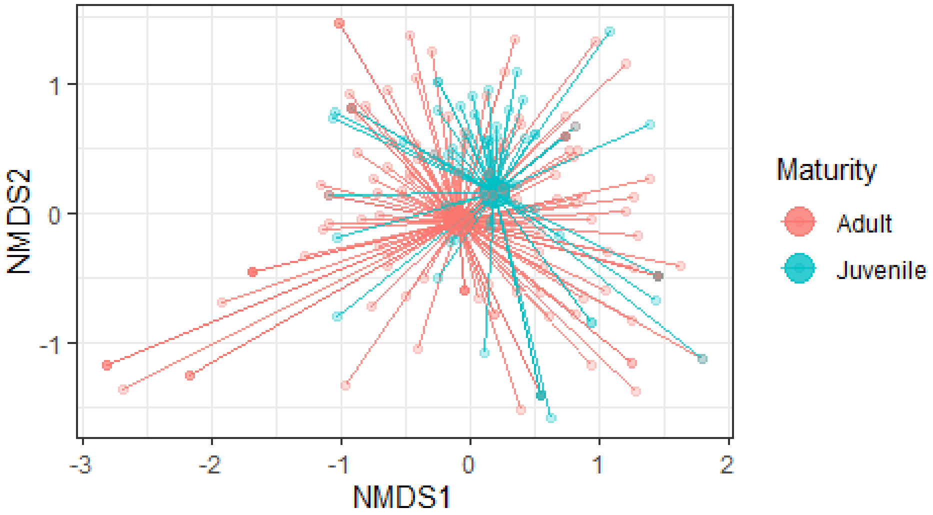
```
species.scores <- as.data.frame(scores(diet.mds, "species"))
species.scores$species <- rownames(species.scores)
names(species.scores)[c(1, 2)] <- c("x", "y")
species.scores$z <- NA

genusspiplotsp <- ggplot(species.scores, aes(x = x, y = y)) + theme_bw() +
  geom_text(data=species.scores,aes(x=x,y=y,label=species), color="black", si
ze=4,alpha=0.75, angle=0)

genusspiplotspp <- ggplot(scrs, aes(x = NMDS1, y = NMDS2, colour = Genus)) +
scale_fill_brewer(2,  "Accent") + geom_segment(data = segs, mapping = aes(xen
d = oNMDS1, yend = oNMDS2)) + geom_point(data = cent, size = 5, alpha=1) + ge
om_point(alpha=0.25) + coord_fixed() + theme_bw() +
  geom_text(data=species.scores,aes(x=x,y=y,label=species), color="red", size
=3,alpha=0.25, angle=0) #+ coord_equal()

genusspiplotspp

## Warning: Removed 5 rows containing missing values (geom_text).
```



Life stage Ordispider

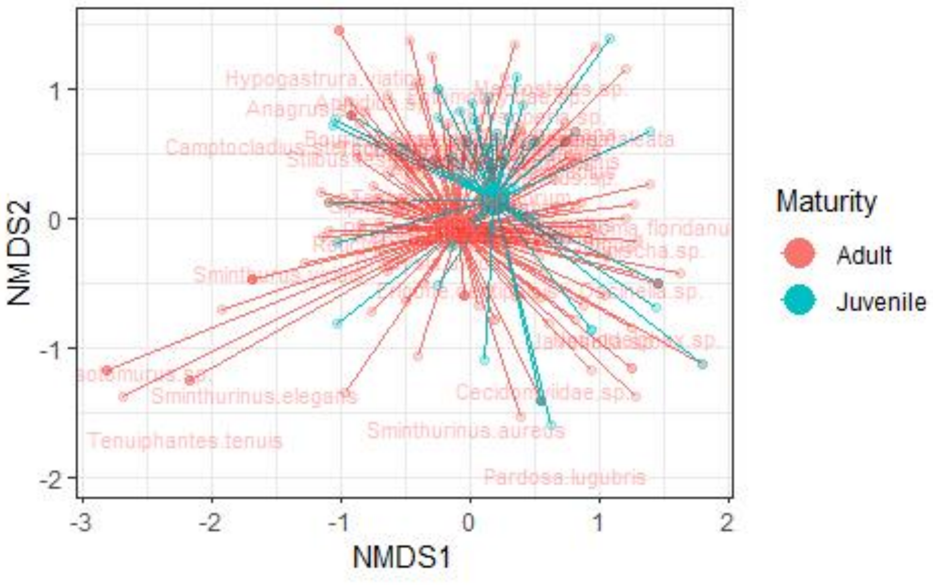
A spiderplot can also be generated for dietary differences between life stages.

```
scrs <- scores(diet.mds, display = 'sites')
scrs <- cbind(as.data.frame(scrs), Maturity = dietnmds$Maturity)
cent <- aggregate(cbind(NMDS1, NMDS2) ~ Maturity, data = scrs, FUN = mean)
segs <- merge(scrs, setNames(cent, c('Maturity', 'oNMDS1', 'oNMDS2')), by = '
Maturity', sort = FALSE)

matspiplot <- ggplot(scrs, aes(x = NMDS1, y = NMDS2, colour = Maturity)) + sc
ale_fill_brewer(6,  "Accent") + geom_segment(data = segs, mapping = aes(xend
= oNMDS1, yend = oNMDS2), alpha=0.8) + geom_point(data = cent, size = 5, alph
a = 0.8) + geom_point(alpha=0.25) + coord_fixed() + theme_bw()
matspiplot
```



And again with prey species labels.

```
species.scores <- as.data.frame(scores(diet.mds, "species"))
species.scores$species <- rownames(species.scores)
names(species.scores)[c(1, 2)] <- c("x", "y")
species.scores$z <- NA

matspiplotsp <- ggplot(species.scores, aes(x = x, y = y)) + theme_bw() +
  geom_text(data=species.scores,aes(x=x,y=y,label=species), color="black", si
ze=4,alpha=0.75, angle=0)

matspiplotspp <- ggplot(scrs, aes(x = NMDS1, y = NMDS2, colour = Maturity)) +
scale_fill_brewer(2,  "Accent") + geom_segment(data = segs, mapping = aes(xen
d = oNMDS1, yend = oNMDS2)) + geom_point(data = cent, size = 5, alpha=1) + ge
om_point(alpha=0.25) + coord_fixed() + theme_bw() +
  geom_text(data=species.scores,aes(x=x,y=y,label=species), color="red", size
=3,alpha=0.25, angle=0) #+ coord_equal()

matspiplotspp

## Warning: Removed 5 rows containing missing values (geom_text).
```



Web comparison

```
webs <- read.csv("Webdata.csv") # Microlinyphia removed to reduce leverage
```

Web height

To create a GLM for web height, we first need to check that web height meets the assumptions, which it does not for normality.

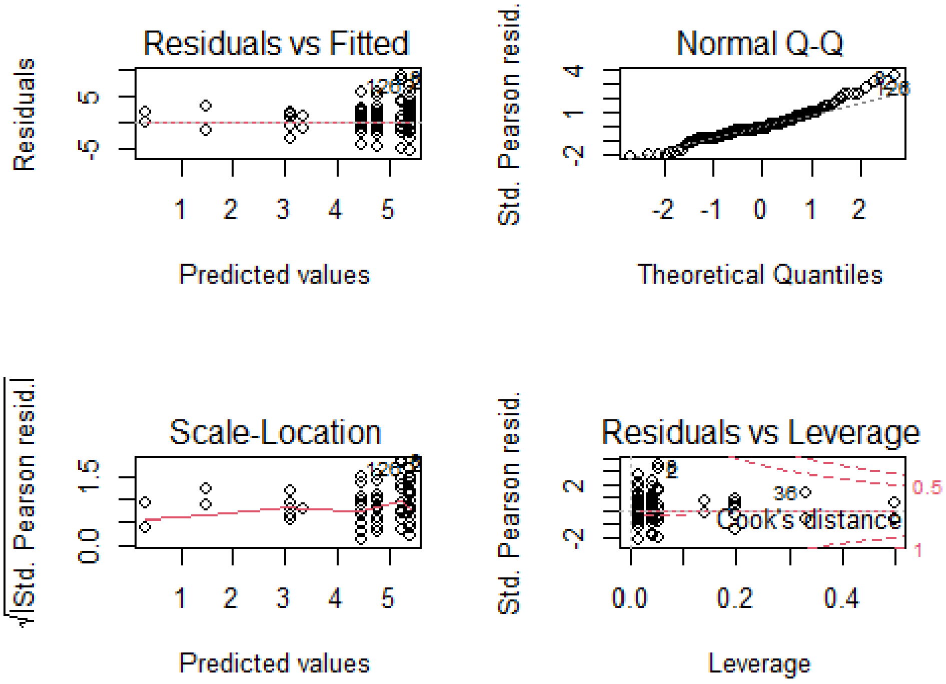
```
hist(webs$Web.height.mm)
```

We can, however, square-root transform it to make it fit.

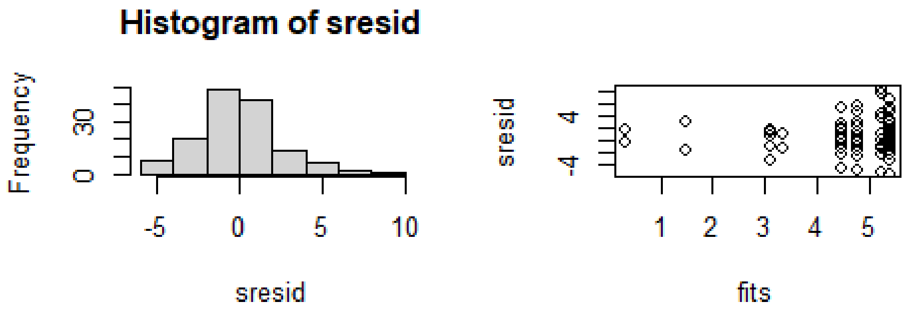
```
hist(sqrt(webs$Web.height.mm))
```

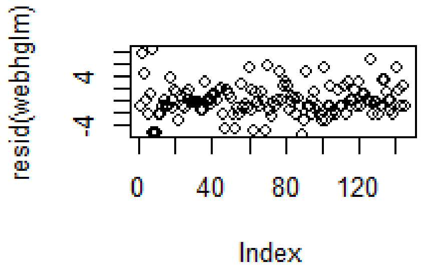
We can now create a model, check the assumptions are met, and summarise the results.

```
webhglm <- glm(sqrt(Web.height.mm) ~ Genus + Sex + Genus:Sex
               , family=gaussian, data=webs)

par(mfrow=c(2,2))
plot(webhglm)
```



```
fits <- fitted(webhglm)
sresid <- resid(webhglm, type = "pearson")
hist(sresid)
plot(sresid ~ fits)
plot(resid(webhglm))

summary(webhglm)

##
## Call:
## glm(formula = sqrt(Web.height.mm) ~ Genus + Sex + Genus:Sex,
##     family = gaussian, data = webs)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -5.3826  -1.4907  -0.3194   1.3256   8.5599
##
## Coefficients: (1 not defined because of singularities)
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)                5.2241     0.5866   8.906 2.87e-15 ***
## GenusErigone              -4.9047     1.1305  -4.338 2.76e-05 ***
## GenusTenuiphantes         -0.7748     0.7782  -0.996    0.321
## SexMale                   -2.1079     1.2852  -1.640    0.103
## SexN/A                    -1.8700     1.9008  -0.984    0.327
## GenusErigone:SexMale       3.2791     2.1829   1.502    0.135
## GenusTenuiphantes:SexMale  2.4460     1.4824   1.650    0.101
## GenusErigone:SexN/A            NA         NA      NA       NA
## GenusTenuiphantes:SexN/A    2.8033     1.9954   1.405    0.162
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 6.537782)
##
##     Null deviance: 1112.36  on 144  degrees of freedom
## Residual deviance:  895.68  on 137  degrees of freedom
## AIC: 693.51
##
## Number of Fisher Scoring iterations: 2
```



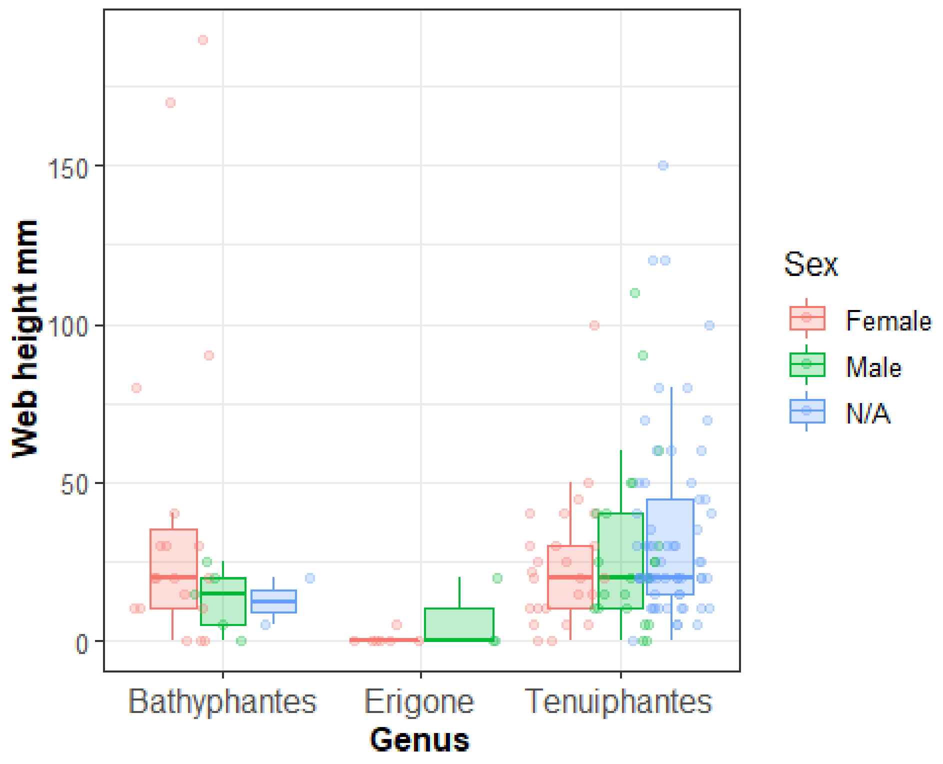


To assess every relationship between the categories, we must, however, relevel the factors and re-run it. For example:

```
webs$Genus <- relevel(webs$Genus, ref = "Bathyphantes")
webs$Sex <- relevel(webs$Sex, ref = "Female")

webhglm <- glm(sqrt(Web.height.mm) ~ Genus + Sex + Genus:Sex
               , family=gaussian, data=webs)

summary(webhglm)
```

We can then create boxplots to highlight the significant differences in web height, with points jittered and overlaid to highlight the density of points at each height.

```
Webpal <- brewer.pal(3, "Accent")

web_height <- ggplot(webs, aes(x=Genus, y=Web.height.mm, fill=Sex)) +
  geom_boxplot(alpha=0.25, aes(colour=Sex), outlier.colour = NA) + theme_bw()
+ scale_x_discrete() +
  #scale_colour_manual(values=Webpal, name = "Tropho-species") +
  geom_point(position=position_jitterdodge(dodge.width=0.8), aes(colour=Sex),
alpha=0.25)+
  theme(text = element_text(size = 12),
        axis.title = element_text(face="bold"),
        axis.text.x=element_text(size = 12)) + labs(y="Web height mm")

web_height
```



Web area

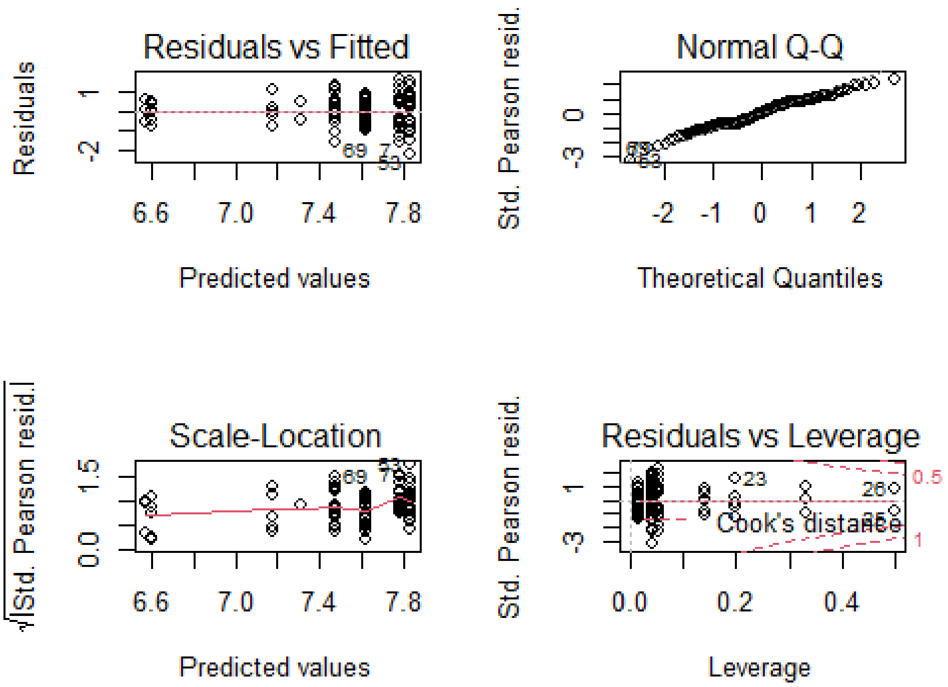
Similarly, web area is non-normal, but this is fixed with a log transformation.

```
hist(webs$Web.area)
hist(log(webs$Web.area))
```

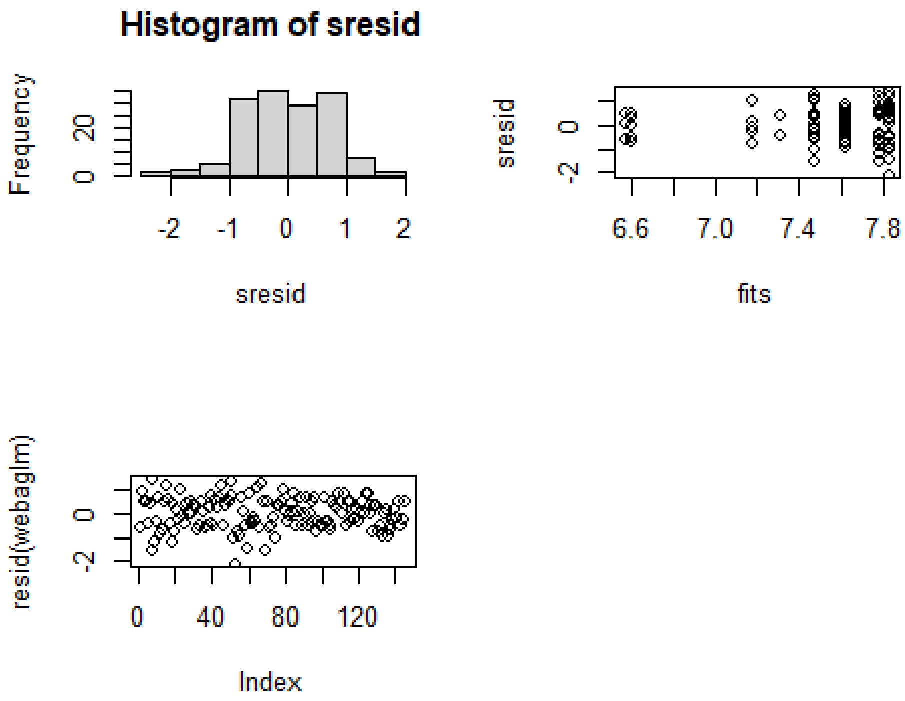
We can then create a model as before, and check assumptions before checking the output.

```
webaglm <- glm(log(Web.area) ~ Genus + Sex + Genus:Sex
               , family=gaussian, data=webs)
```

```r
par(mfrow=c(2,2))
plot(webaglm)
```



```r
fits <- fitted(webaglm)
sresid <- resid(webaglm, type = "pearson")
hist(sresid)
plot(sresid ~ fits)
plot(resid(webaglm))

summary(webaglm)

##
## Call:
## glm(formula = log(Web.area) ~ Genus + Sex + Genus:Sex, family = gaussian,
##     data = webs)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.12783  -0.51839  -0.02338   0.53624   1.51829
##
## Coefficients: (1 not defined because of singularities)
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 7.78736    0.15744  49.463  < 2e-16 ***
## GenusErigone               -1.19572    0.30342  -3.941 0.000129 ***
## GenusTenuiphantes           0.04425    0.20886   0.212 0.832526
## SexMale                    -0.60802    0.34493  -1.763 0.080171 .
## SexN/A                     -0.47414    0.51015  -0.929 0.354315
## GenusErigone:SexMale       0.58359    0.58586   0.996 0.320949
## GenusTenuiphantes:SexMale  0.25082    0.39785   0.630 0.529467
## GenusErigone:SexN/A             NA         NA      NA       NA
## GenusTenuiphantes:SexN/A   0.26681    0.53555   0.498 0.619145
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.4709427)
##
##     Null deviance: 78.128  on 144  degrees of freedom
## Residual deviance: 64.519  on 137  degrees of freedom
## AIC: 312.08
##
## Number of Fisher Scoring iterations: 2
```

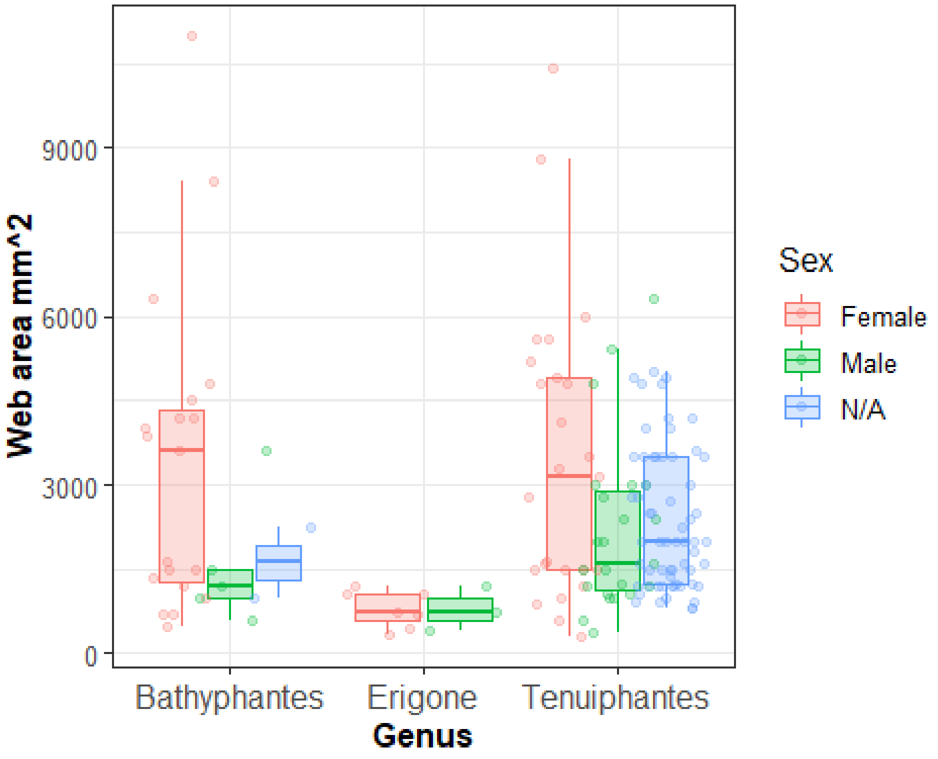


Again, we can re-level this as above to compare between categories.

We can then create a jittered boxplot again, this time for web area.

```
web_area <- ggplot(webs, aes(x=Genus, y=Web.area, fill=Sex)) +
  geom_boxplot(alpha=0.25, aes(colour=Sex), outlier.colour = NA) + theme_bw()
+ scale_x_discrete() +
  geom_point(position=position_jitterdodge(dodge.width=0.8), aes(colour=Sex),
alpha=0.25)+
  theme(text = element_text(size = 12),
        axis.title = element_text(face="bold"),
        axis.text.x=element_text(size = 12)) + labs(y="Web area mm^2")
```

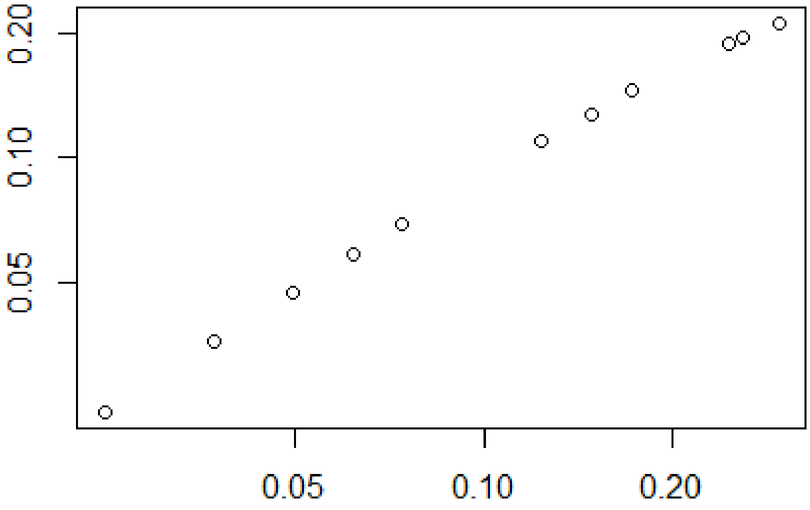
```
web_area
```



Web diet

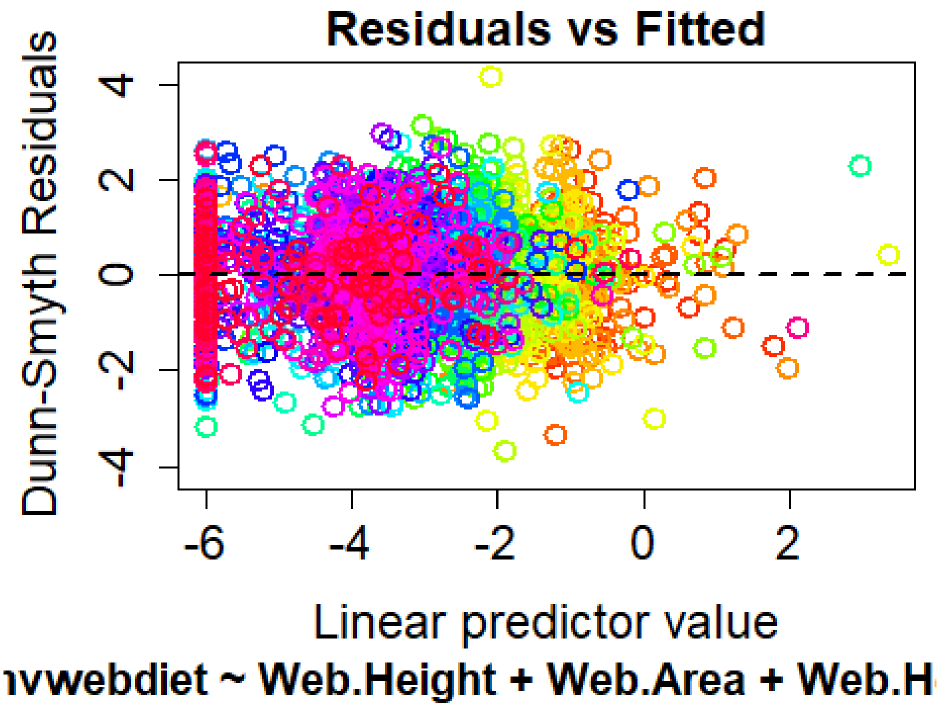
We are really interested in not just how webs differ between spiders, but also how this may affect their diets, so we can begin to analyse that via mvabund.

```
webdiet <- read.csv("2018dietarydatawebbednosing.csv")
rownames(webdiet) <- webdiet[,1]
webdietprey <- webdiet[,25:56]

mvwebdiet <- mvabund(webdiet[,25:56])
meanvar.plot(mvwebdiet)
```

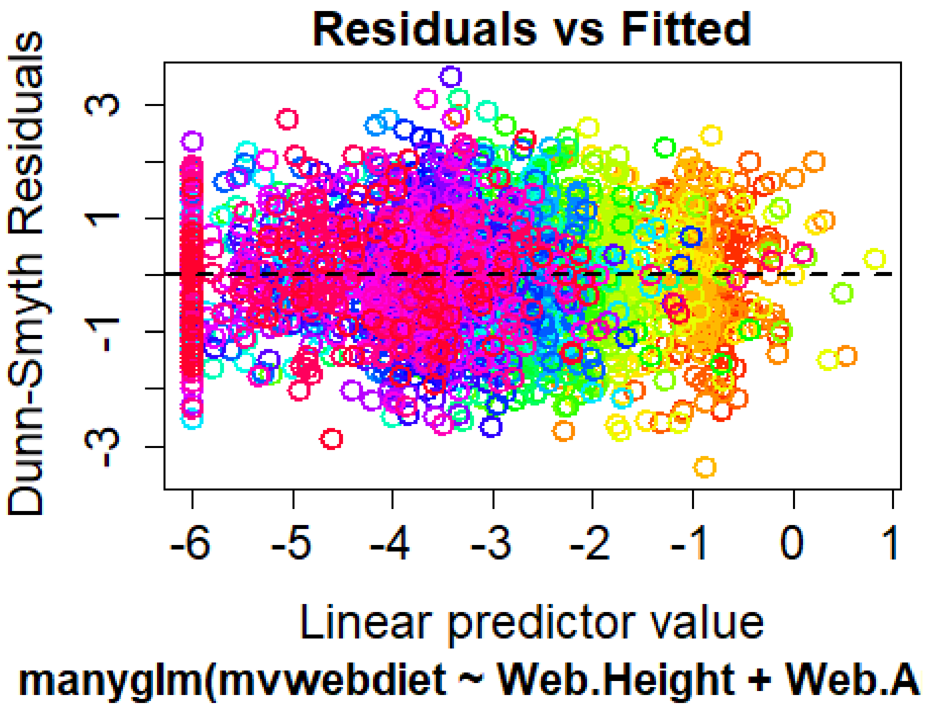


As before, we must create and simplify a model before then viewing the output.

```
webdiet1<-manyglm(mvwebdiet ~ Web.Height + Web.Area +
                    Web.Height:Web.Area
                  , data=webdiet, family="binomial(cloglog)")

plot(webdiet1)
```



```
step(webdiet1)

webdiet2<-manyglm(mvwebdiet ~ Web.Height + Web.Area
                  , data=webdiet, family="binomial(cloglog)")

plot(webdiet2)
```



```
anova(webdiet2, p.uni="adjusted", resamp="montecarlo")

## Time elapsed: 0 hr 0 min 42 sec

## Analysis of Deviance Table
##
## Model: mvwebdiet ~ Web.Height + Web.Area
##
## Multivariate test:
##             Res.Df Df.diff   Dev Pr(>Dev)
## (Intercept)     80
## Web.Height      79       1 33.83    0.417
## Web.Area        78       1 48.90    0.082 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Univariate Tests:
##             Acrodactyla.degener          Aeolothrips.intermedius
##                             Dev Pr(>Dev)                     Dev Pr(>Dev)
## (Intercept)
## Web.Height                0.001    1.000                   0.303    1.000
## Web.Area                  0.023    1.000                   1.406    1.000
##             Anaphothrips.obscurus          Aphelinus.sp.          Aphidius
.sp.
##                               Dev Pr(>Dev)           Dev Pr(>Dev)
Dev
## (Intercept)
## Web.Height                  0.189    1.000         0.082    1.000        1
.703
## Web.Area                    9.646    0.181         0.777    1.000        0
.051
##                      Bourletiellidae.sp.          Bradysia.urticae
##             Pr(>Dev)                 Dev Pr(>Dev)              Dev Pr(>Dev
)
## (Intercept)
## Web.Height     0.999               0.037    1.000            0.436    1.00
0
## Web.Area       1.000               0.089    1.000             0.75    1.00
0
##             Cecidomyiidae.sp.          Coproica.ferruginata
##                           Dev Pr(>Dev)                  Dev Pr(>Dev)
## (Intercept)
## Web.Height                1.336    0.999                0.082    1.000
## Web.Area                  0.152    1.000                1.427    1.000
##             Corynoptera.sp.          Entomobryidae.sp.          Eupodidae.
sp.
##                         Dev Pr(>Dev)               Dev Pr(>Dev)
Dev
## (Intercept)
## Web.Height              0.288    1.000             4.257    0.710         0.
006
## Web.Area                0.008    1.000             0.865    1.000         2.
737
##                      Frankliniella.tenuicornis          Hypogastrura.viati
ca
##             Pr(>Dev)                       Dev Pr(>Dev)                  D
ev
## (Intercept)
## Web.Height     1.000                     4.359    0.678                1.4
69
## Web.Area       0.954                     4.074    0.871                0.1
94
##                      Isotomurus.sp.          Javesella.sp.
##             Pr(>Dev)            Dev Pr(>Dev)           Dev Pr(>Dev)
## (Intercept)
## Web.Height     0.999          1.065    0.999         2.535    0.979
## Web.Area       1.000          0.181    1.000             0    1.000
##             Limothrips.denticornis          Macrosteles.sp.
##                                Dev Pr(>Dev)             Dev Pr(>Dev)
## (Intercept)
## Web.Height                   0.301    1.000           1.936    0.994
## Web.Area                     3.004    0.944            0.77    1.000
##             Neriene.montana          Nothodelphax.sp.          Oscinella.s
p.
##                         Dev Pr(>Dev)              Dev Pr(>Dev)           D
ev
## (Intercept)
## Web.Height            0.578    1.000              0.1    1.000         2.5
64
## Web.Area              1.766    0.997            0.056    1.000
0
##                      Pardosa.pullata
##             Pr(>Dev)             Dev Pr(>Dev)
## (Intercept)
## Web.Height     0.979           0.288    1.000
## Web.Area       1.000            0.63    1.000
##             Reticulitermes.lucifugus.lucifugus          Rhopalosiphum.sp.
##                                            Dev Pr(>Dev)               Dev
## (Intercept)
## Web.Height                                1.68    0.999             0.164
## Web.Area                                 0.235    1.000             1.079
##                      Scatopsciara.atomaria          Sipha.sp.
##             Pr(>Dev)                   Dev Pr(>Dev)       Dev Pr(>Dev)
## (Intercept)
## Web.Height     1.000                 0.018    1.000     0.584    1.000
## Web.Area       1.000                 0.877    1.000     0.713    1.000
##             Sitobion.sp.          Sminthurus.viridis          Stilbus.test
aceus
##                      Dev Pr(>Dev)                Dev Pr(>Dev)
Dev
## (Intercept)
## Web.Height         2.911    0.963              2.288    0.984
1.112
## Web.Area               0    1.000              3.445    0.929
0.288
##                      Tachyporus.chrysomelinus          Tachyporus.hypnorum
##             Pr(>Dev)                      Dev Pr(>Dev)                 Dev
## (Intercept)
## Web.Height     0.999                    1.112    0.999               0.001
## Web.Area       1.000                    6.525    0.457               3.807
##                      Trombidiidae.sp.
##             Pr(>Dev)              Dev Pr(>Dev)
## (Intercept)
## Web.Height     1.000            0.044    1.000
## Web.Area       0.892            3.324    0.930
## Arguments:
##  Test statistics calculated assuming uncorrelated response (for faster com
putation)
## P-value calculated using 999 iterations via parametric resampling.
```

Intraguild predation and biocontrol

We can analyse the incidence of intraguild and pest predation in the diets of these spiders separately to the main dietary analyses.

```
IPBC <- read.csv("IPBC.csv")
IPBC$Field <- as.factor(IPBC$Field)
```

```
IPBC$Site <- as.factor(IPBC$Site)
rownames(IPBC) <- IPBC[,1]
```

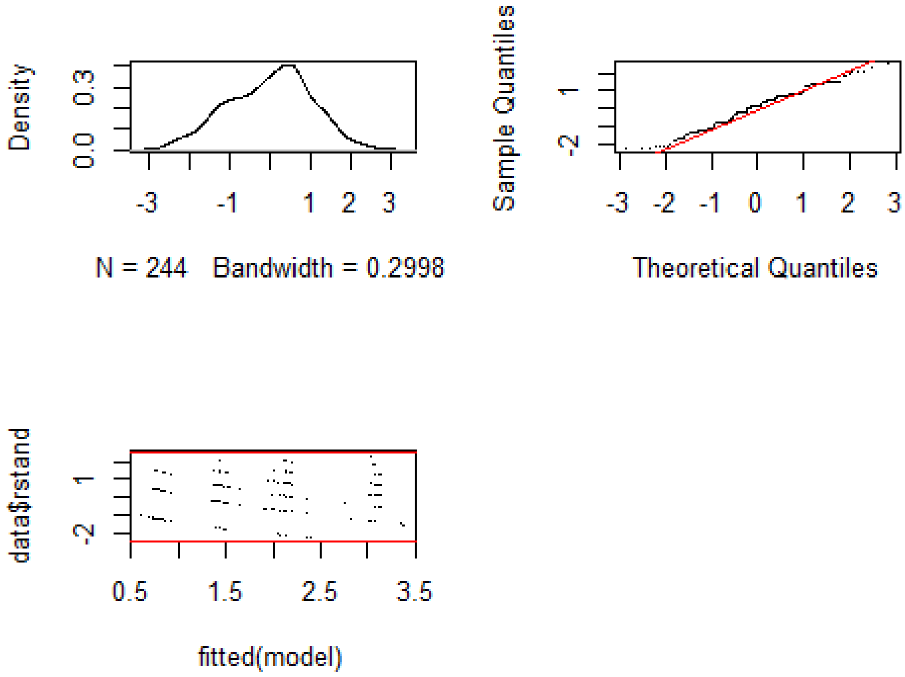
We need to scale Julian days for better model fits.

```
IPBC$Day2 <- IPBC$Julian.Day-min(IPBC$Julian.Day)
IPBC$Day2s <- scale(IPBC$Day2)
```

Analysis of pest predation (biocontrol)

We want to first check if site should be included as a random effect by creating and comparing 'glmer' and 'glm' models, and checking the assumptions fit.

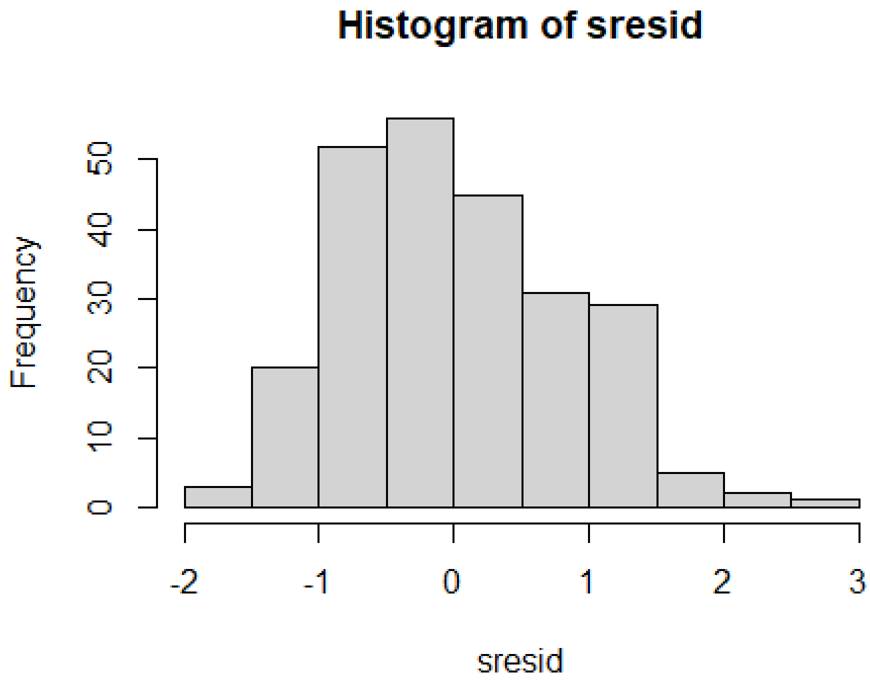
```
BCm1 <- glmer(Pest ~ Genus + Maturity + Day2s
                + (1 | Site), family=poisson, data=IPBC)

## boundary (singular) fit: see ?isSingular

BCm2 <- glm(Pest ~ Genus + Maturity + Day2s,
              family=poisson, data=IPBC)

mcp.fnc(BCm1)       # heteroscedasticity and qqplot
```



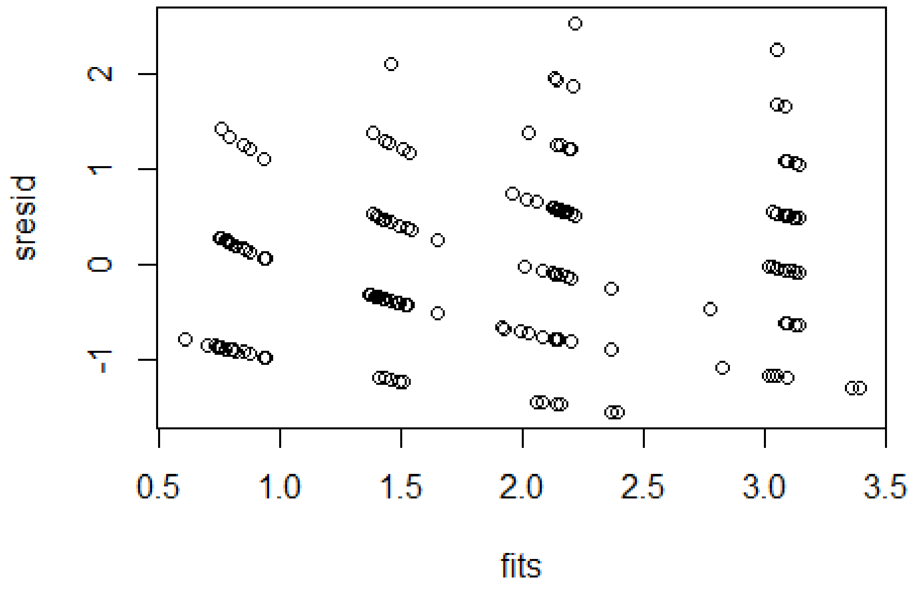
```
plotLMER.fnc(BCm1)  # Plots model relationship
```

```
## effect size (range) for  Genus is  1.284965
## effect size (range) for  Maturity is  0.3487272
## effect size (range) for  Day2s is  0.2170137
```

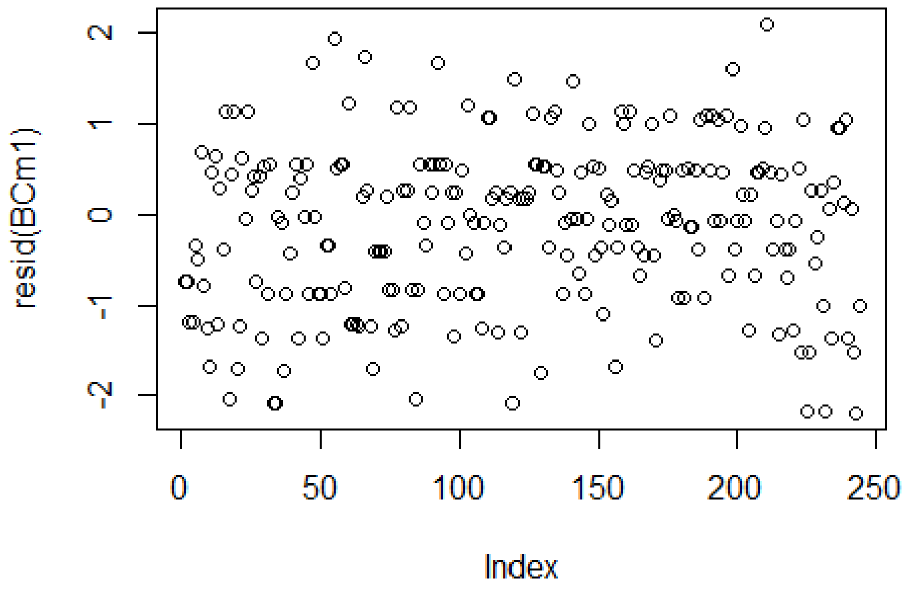
```
fits <- fitted(BCm1)
sresid <- resid(BCm1, type = "pearson")
hist(sresid)
```



```
plot(sresid ~ fits)
```



```
plot(resid(BCm1))
```



```
anova(BCm1, BCm2, test="Chisq")
```

```
## Data: IPBC
## Models:
## BCm2: Pest ~ Genus + Maturity + Day2s
## BCm1: Pest ~ Genus + Maturity + Day2s + (1 | Site)
##      npar    AIC     BIC  logLik deviance Chisq Df Pr(>Chisq)
## BCm2    7 748.59 773.07 -367.29   734.59
## BCm1    8 750.59 778.57 -367.29   734.59     0  1          1
```

```
G2 = -2 * logLik(BCm2) + 2 * logLik(BCm1)
pchisq(as.numeric(G2), df=1, lower.tail=F)
```

```
## [1] 1
```

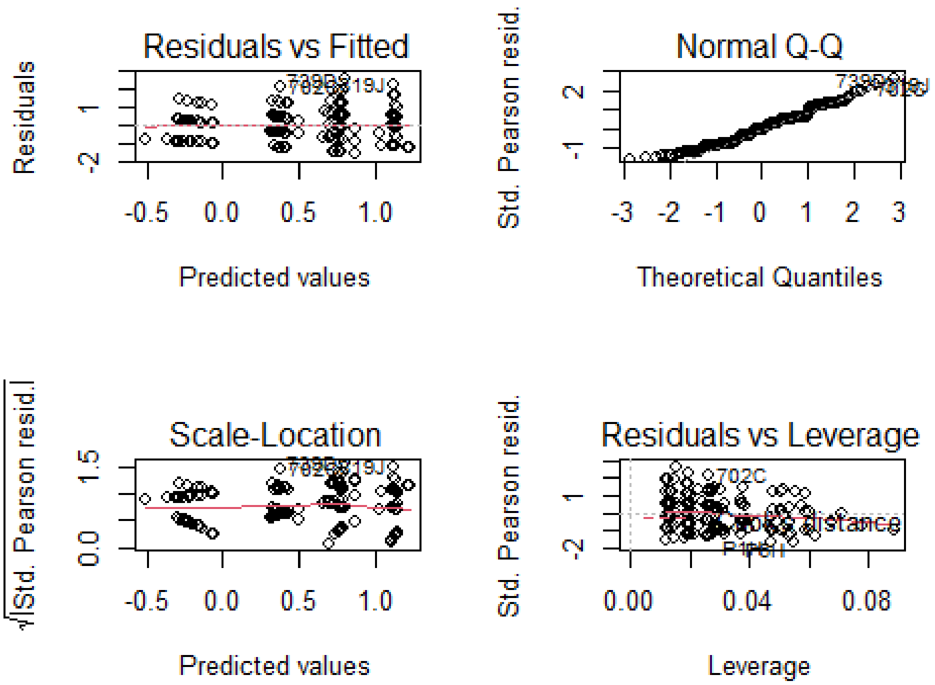
```
lrtest(BCm2, BCm1)
```

```
## Warning in modelUpdate(objects[[i - 1]], objects[[i]]): original model was
of
## class "glm", updated model is of class "glmerMod"
```

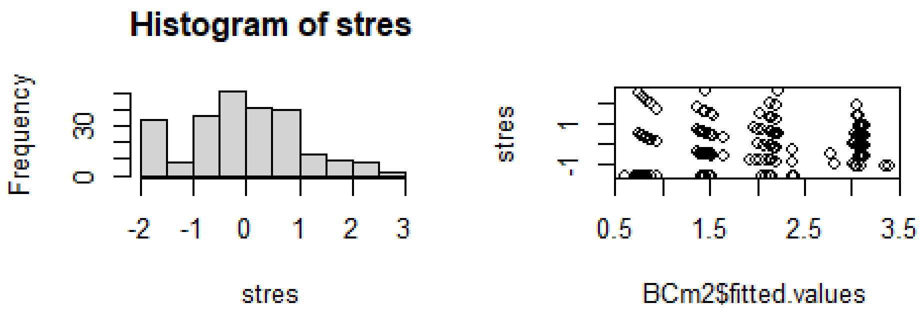
```
## Likelihood ratio test
##
## Model 1: Pest ~ Genus + Maturity + Day2s
## Model 2: Pest ~ Genus + Maturity + Day2s + (1 | Site)
##   #Df  LogLik Df Chisq Pr(>Chisq)
## 1   7 -367.29
## 2   8 -367.29  1     0          1
```

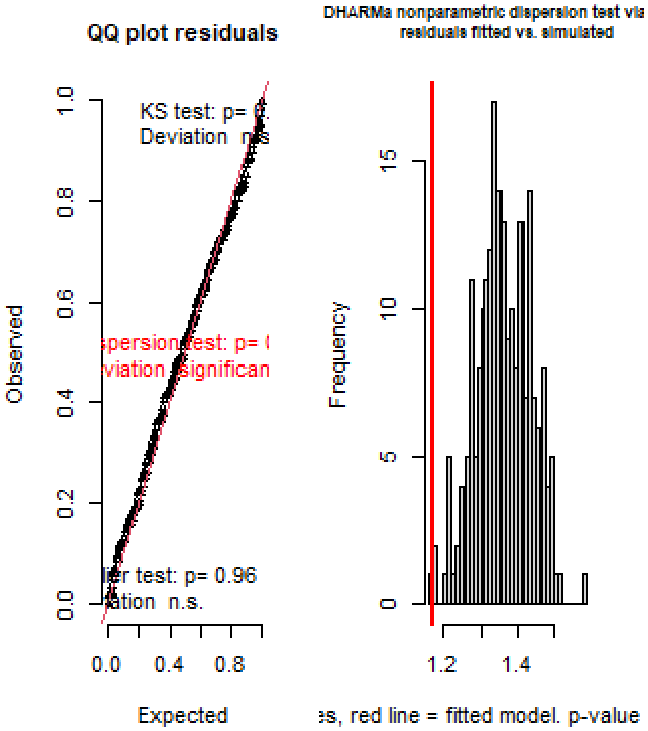
Having selected the 'glm' model, we can now check th assumptions fully and produce outputs.

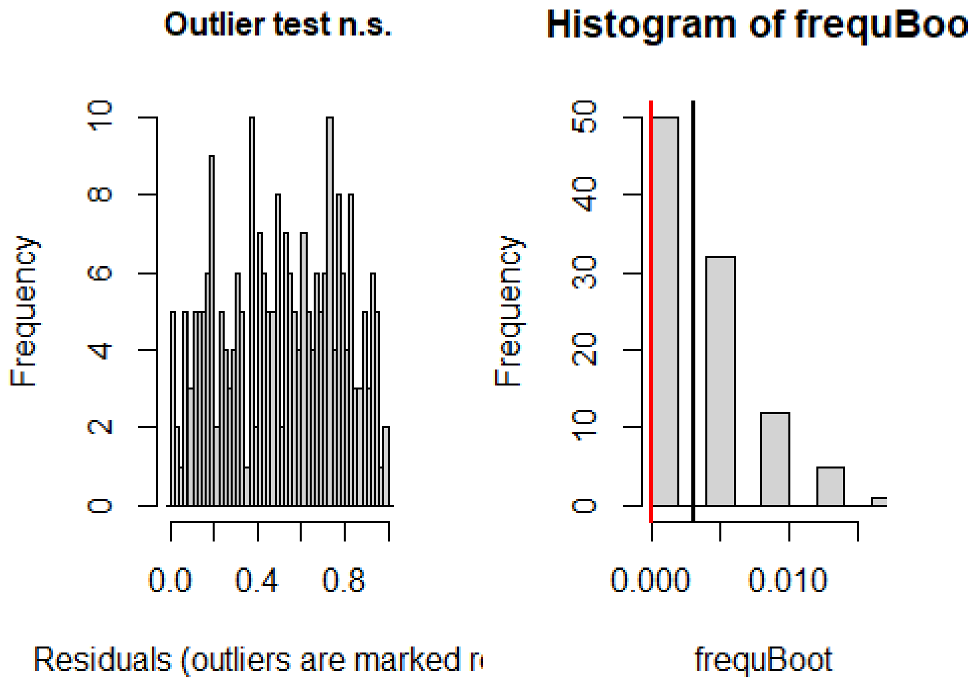
```
BCm2 <- glm(Pest ~ Genus + Maturity + Day2s,
            family=poisson, data=IPBC)

par(mfrow=c(2,2))
plot(BCm2)
```



```
stres<- (BCm2$residuals - mean(BCm2$residuals))/           sd(BCm2$residuals
)
hist(stres)
plot(stres ~ BCm2$fitted.values)
theta <- BCm2$deviance/BCm2$df.residual
theta

## [1] 0.8398758

testResiduals(BCm2, plot = T)
```


Histogram of stres
Frequency
0
30
-2
-1
0
1
2
3
stres
stres
-1
1
0.5
1.5
2.5
3.5
BCm2$fitted.values


QQ plot residuals
KS test: p= 0.
Deviation n.s
spersion test: p= (
viation significan
lier test: p= 0.96
iation n.s.
Observed
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.4
0.8
Expected
DHARMa nonparametric dispersion test via
residuals fitted vs. simulated
Frequency
0
5
10
15
1.2
1.4
es, red line = fitted model. p-value



```
## $uniformity
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  simulationOutput$scaledResiduals
## D = 0.049924, p-value = 0.5773
## alternative hypothesis: two-sided
##
##
## $dispersion
##
##  DHARMa nonparametric dispersion test via sd of residuals fitted vs.
##  simulated
##
## data:  simulationOutput
## ratioObsSim = 0.8587, p-value = 0.008
## alternative hypothesis: two.sided
##
##
## $outliers
##
##  DHARMa bootstrapped outlier test
##
## data:  simulationOutput
## outliers at both margin(s) = 0, observations = 244, p-value = 1
## alternative hypothesis: two.sided
##  percent confidence interval:
##  0.00000000 0.01229508
## sample estimates:
## outlier frequency (expected: 0.00307377049180328 )
##                                                   0
```

```
## $uniformity
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  simulationOutput$scaledResiduals
## D = 0.049924, p-value = 0.5773
## alternative hypothesis: two-sided
##
##
## $dispersion
##
##  DHARMa nonparametric dispersion test via sd of residuals fitted vs.
##  simulated
##
## data:  simulationOutput
## ratioObsSim = 0.8587, p-value = 0.008
## alternative hypothesis: two.sided
##
##
## $outliers
##
##  DHARMa bootstrapped outlier test
##
## data:  simulationOutput
## outliers at both margin(s) = 0, observations = 244, p-value = 1
## alternative hypothesis: two.sided
##  percent confidence interval:
##  0.00000000 0.01229508
## sample estimates:
## outlier frequency (expected: 0.00307377049180328 )
##                                                   0
```

```r
summary.glm(BCm2)
```

```
##
## Call:
## glm(formula = Pest ~ Genus + Maturity + Day2s, family = poisson,
##     data = IPBC)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.18796  -0.83665  -0.05838   0.49816   2.09408
##
## Coefficients:
```

```
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)          0.40955    0.13089   3.129  0.00175 **
## GenusErigone        -0.65131    0.23276  -2.798  0.00514 **
## GenusMicrolinyphia -0.07531    0.20901  -0.360  0.71861
## GenusPardosa        -0.92428    0.28203  -3.277  0.00105 **
## GenusTenuiphantes    0.36068    0.14453   2.496  0.01258 *
## MaturityJuvenile     0.34873    0.10660   3.271  0.00107 **
## Day2s                0.04202    0.05152   0.816  0.41470
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 285.09  on 243  degrees of freedom
## Residual deviance: 199.05  on 237  degrees of freedom
## AIC: 748.59
##
## Number of Fisher Scoring iterations: 5
```

```
summary(BCm2)
```

```
##
## Call:
## glm(formula = Pest ~ Genus + Maturity + Day2s, family = poisson,
##     data = IPBC)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.18796  -0.83665  -0.05838   0.49816   2.09408
##
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)          0.40955    0.13089   3.129  0.00175 **
## GenusErigone        -0.65131    0.23276  -2.798  0.00514 **
## GenusMicrolinyphia -0.07531    0.20901  -0.360  0.71861
## GenusPardosa        -0.92428    0.28203  -3.277  0.00105 **
## GenusTenuiphantes    0.36068    0.14453   2.496  0.01258 *
## MaturityJuvenile     0.34873    0.10660   3.271  0.00107 **
## Day2s                0.04202    0.05152   0.816  0.41470
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 285.09  on 243  degrees of freedom
## Residual deviance: 199.05  on 237  degrees of freedom
## AIC: 748.59
##
## Number of Fisher Scoring iterations: 5
```

```
anova(BCm2)

## Analysis of Deviance Table
##
## Model: poisson, link: log
##
## Response: Pest
##
## Terms added sequentially (first to last)
##
##
##          Df Deviance Resid. Df Resid. Dev
## NULL                       243     285.09
## Genus     4   73.420       239     211.67
## Maturity  1   11.954       238     199.71
## Day2s     1    0.662       237     199.05
```

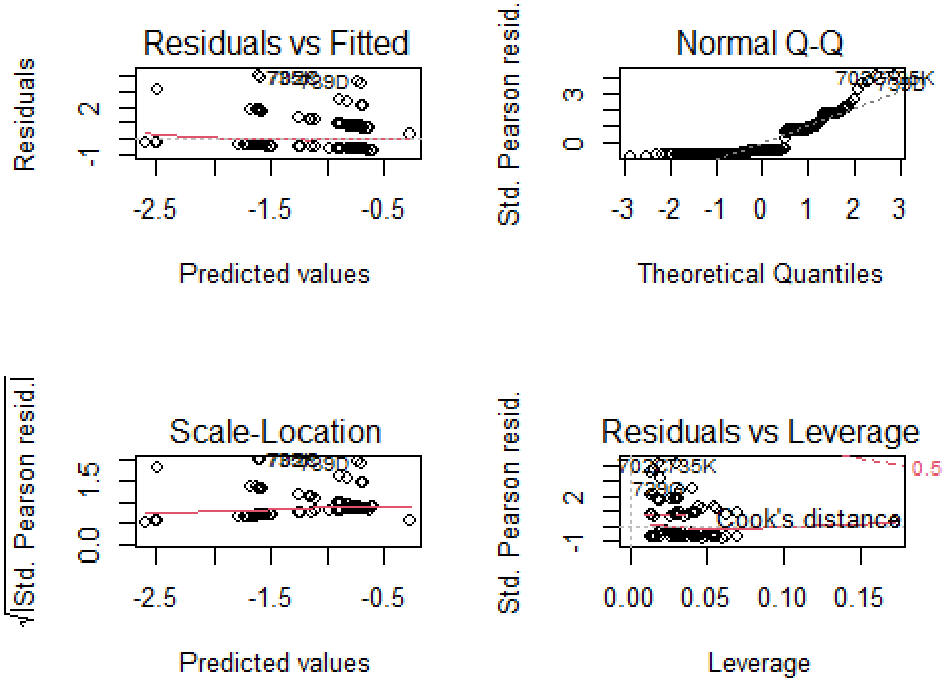
Again, we can relevel the factors to assess relationships between different categories.

```
IPBC$Genus <- relevel(IPBC$Genus, ref = "Tenuiphantes")
```

Intraguild predation

We can do the same for intraguild predation modelling.

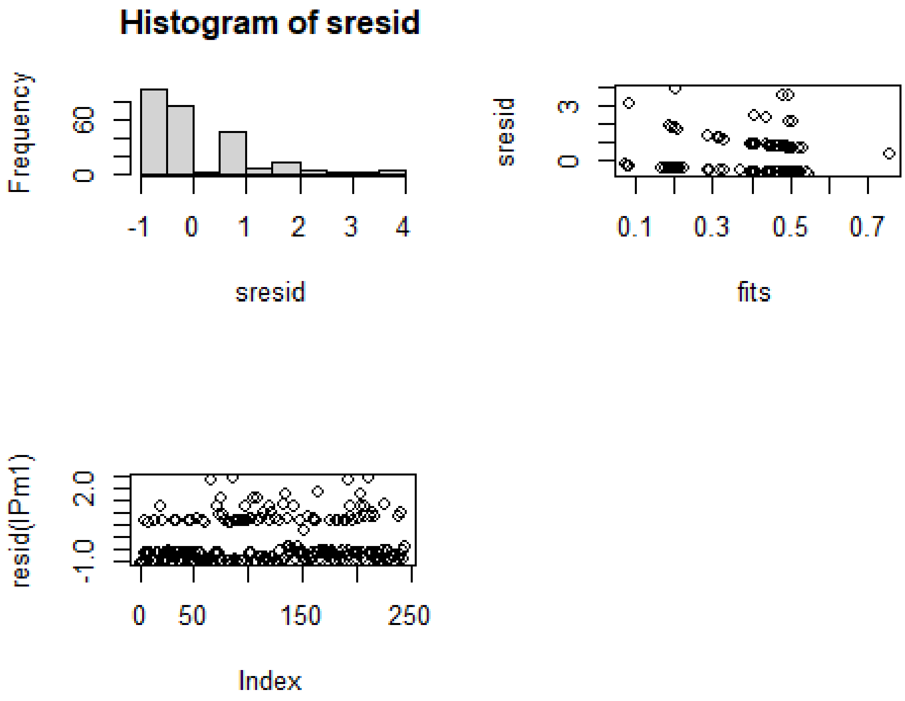
```
IPm1 <- glmer.nb(Predator ~ Genus + Maturity + Day2s
              + (1 | Site),  data=IPBC)

## Warning in theta.ml(Y, mu, weights = object@resp$weights, limit = limit, :
## iteration limit reached

## boundary (singular) fit: see ?isSingular

IPm2 <- glm(Predator ~ Genus + Maturity + Day2s ,
            family=poisson, data=IPBC)


par(mfrow=c(2,2))
plot(IPm2)
```



```
fits <- fitted(IPm1)
sresid <- resid(IPm1, type = "pearson")
hist(sresid)
plot(sresid ~ fits)
plot(resid(IPm1))
anova(IPm1, IPm2, test="Chisq")

## Data: IPBC
## Models:
## IPm2: Predator ~ Genus + Maturity + Day2s
## IPm1: Predator ~ Genus + Maturity + Day2s + (1 | Site)
##      npar   AIC    BIC  logLik deviance Chisq Df Pr(>Chisq)
## IPm2    7 370.9 395.38 -178.45    356.9
## IPm1    9 374.9 406.37 -178.45    356.9     0  2          1

G2 = -2 * logLik(IPm2) + 2 * logLik(IPm1)
pchisq(as.numeric(G2), df=1, lower.tail=F)

## [1] 1

lrtest(IPm2, IPm1)

## Warning in modelUpdate(objects[[i - 1]], objects[[i]]): original model was
of
## class "glm", updated model is of class "glmerMod"

## Likelihood ratio test
##
```

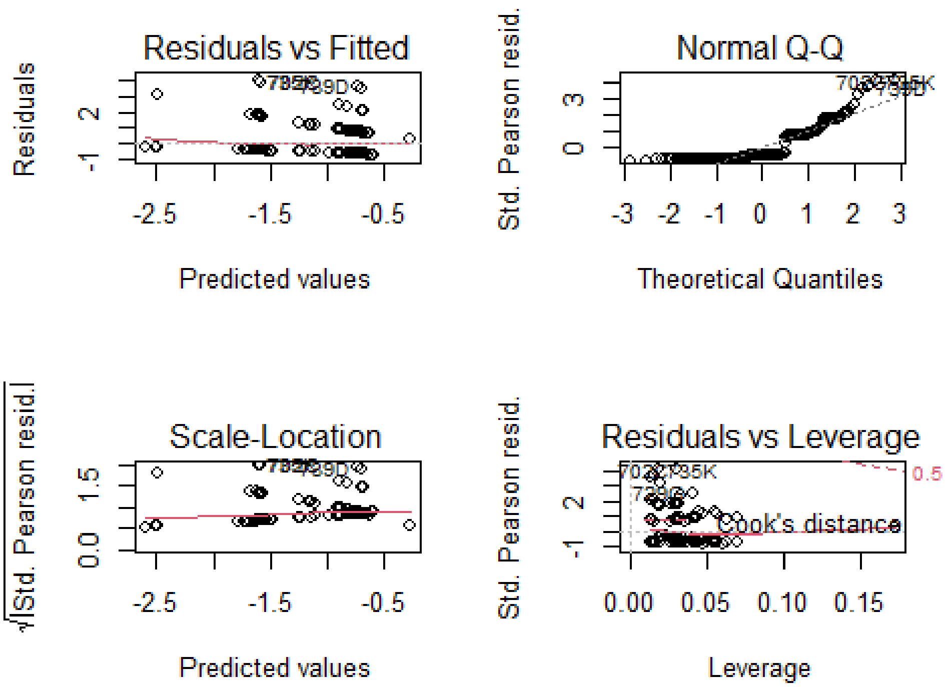
```
## Model 1: Predator ~ Genus + Maturity + Day2s
## Model 2: Predator ~ Genus + Maturity + Day2s + (1 | Site)
##   #Df  LogLik Df Chisq Pr(>Chisq)
## 1   7 -178.45
## 2   9 -178.45  2 8e-04     0.9996
```

```
par(mfrow=c(2,2))
```



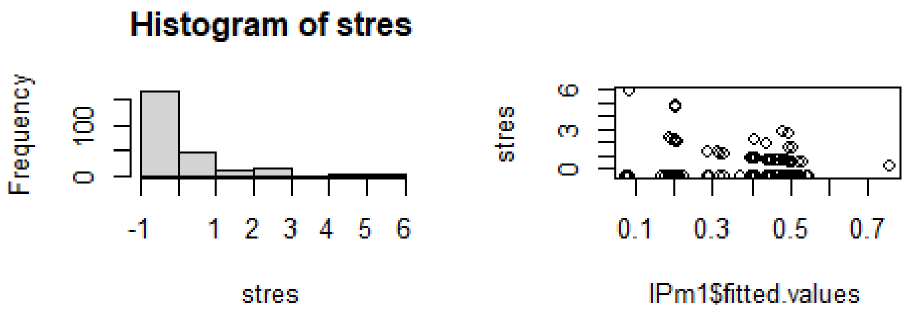
We can then test and create outputs from the final model.

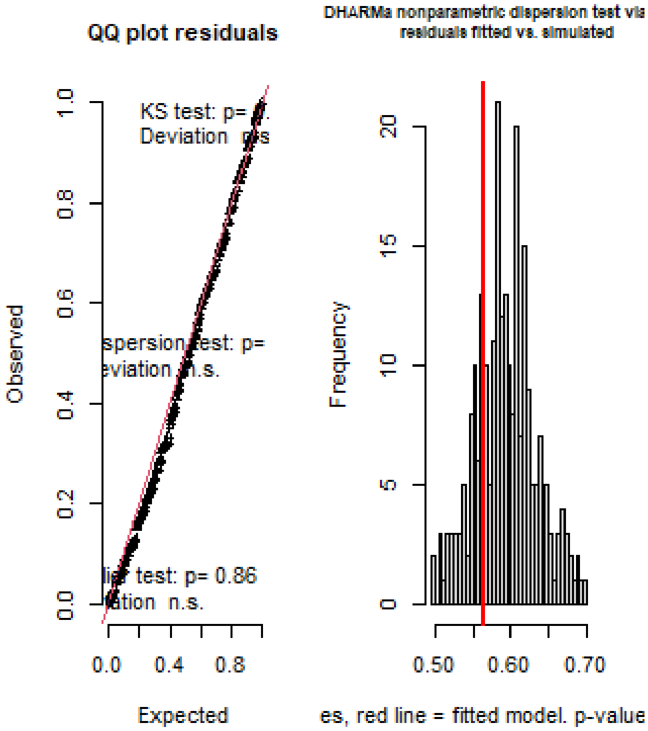
```
IPm1 <- glm(Predator ~ Genus + Day2s + Maturity,
            family=poisson, data=IPBC)

summary(IPm1)

##
## Call:
## glm(formula = Predator ~ Genus + Day2s + Maturity, family = poisson,
##     data = IPBC)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.0480  -0.9414  -0.6393   0.6317   2.4446
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)          -1.62496    0.38048  -4.271 1.95e-05 ***
## GenusErigone         0.82256    0.45387   1.812  0.06994 .
## GenusMicrolinyphia  0.72957    0.49876   1.463  0.14353
## GenusPardosa         1.35077    0.60348   2.238  0.02520 *
## GenusTenuiphantes    0.91208    0.40884   2.231  0.02569 *
## Day2s               -0.04171    0.11362  -0.367  0.71357
## MaturityJuvenile    -0.86577    0.33148  -2.612  0.00901 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 212.03  on 243  degrees of freedom
## Residual deviance: 198.62  on 237  degrees of freedom
## AIC: 370.9
##
## Number of Fisher Scoring iterations: 6

par(mfrow=c(2,2))
plot(IPm1)
```

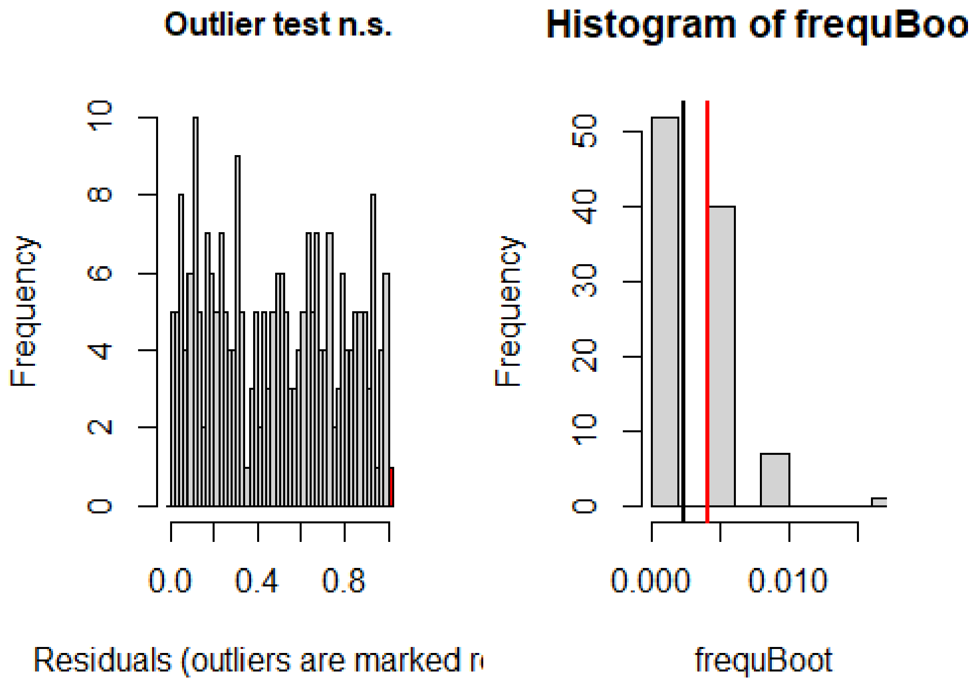


```
stres<- (IPm1$residuals - mean(IPm1$residuals))/           sd(IPm1$residuals
)
hist(stres)
plot(stres ~ IPm1$fitted.values)
theta <- IPm1$deviance/IPm1$df.residual
theta
```

```
## [1] 0.8380458

testResiduals(IPm1, plot = T)
```


Histogram of stres
Frequency
0
100
-1
1
2
3
4
5
6
stres
stres
0
3
6
0.1
0.3
0.5
0.7
IPm1$fitted.values


QQ plot residuals
KS test: p= 0.
Deviation n.s
spersion test: p=
eviation n.s.
lier test: p= 0.86
ation n.s.
Observed
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.4
0.8
Expected
DHARMa nonparametric dispersion test via
residuals fitted vs. simulated
Frequency
0
5
10
15
20
0.50
0.60
0.70
es, red line = fitted model. p-value



```
## $uniformity
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  simulationOutput$scaledResiduals
## D = 0.065158, p-value = 0.2514
## alternative hypothesis: two-sided
##
##
## $dispersion
##
##  DHARMa nonparametric dispersion test via sd of residuals fitted vs.
##  simulated
##
## data:  simulationOutput
## ratioObsSim = 0.95124, p-value = 0.48
## alternative hypothesis: two.sided
##
##
## $outliers
##
##  DHARMa bootstrapped outlier test
##
## data:  simulationOutput
## outliers at both margin(s) = 1, observations = 244, p-value = 0.96
## alternative hypothesis: two.sided
##  percent confidence interval:
##  0.000000000 0.008196721
## sample estimates:
## outlier frequency (expected: 0.00237704918032787 )
##                                         0.004098361
```

```
## $uniformity
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  simulationOutput$scaledResiduals
## D = 0.065158, p-value = 0.2514
## alternative hypothesis: two-sided
##
##
## $dispersion
##
##  DHARMa nonparametric dispersion test via sd of residuals fitted vs.
##  simulated
##
## data:  simulationOutput
## ratioObsSim = 0.95124, p-value = 0.48
## alternative hypothesis: two.sided
##
##
## $outliers
##
##  DHARMa bootstrapped outlier test
##
## data:  simulationOutput
## outliers at both margin(s) = 1, observations = 244, p-value = 0.96
## alternative hypothesis: two.sided
##  percent confidence interval:
##  0.000000000 0.008196721
## sample estimates:
## outlier frequency (expected: 0.00237704918032787 )
##                                         0.004098361
```

```
summary.glm(IPm1)
```

```
##
## Call:
## glm(formula = Predator ~ Genus + Day2s + Maturity, family = poisson,
##     data = IPBC)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.0480  -0.9414  -0.6393   0.6317   2.4446
##
## Coefficients:
```

```
##                     Estimate Std. Error z value Pr(>|z|)    
## (Intercept)         -1.62496    0.38048  -4.271 1.95e-05 ***
## GenusErigone         0.82256    0.45387   1.812  0.06994 .  
## GenusMicrolinyphia   0.72957    0.49876   1.463  0.14353    
## GenusPardosa         1.35077    0.60348   2.238  0.02520 *  
## GenusTenuiphantes    0.91208    0.40884   2.231  0.02569 *  
## Day2s               -0.04171    0.11362  -0.367  0.71357    
## MaturityJuvenile    -0.86577    0.33148  -2.612  0.00901 ** 
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for poisson family taken to be 1)
## 
##     Null deviance: 212.03  on 243  degrees of freedom
## Residual deviance: 198.62  on 237  degrees of freedom
## AIC: 370.9
## 
## Number of Fisher Scoring iterations: 6
```

```
summary(IPm1)
```

```
## 
## Call:
## glm(formula = Predator ~ Genus + Day2s + Maturity, family = poisson, 
##     data = IPBC)
## 
## Deviance Residuals: 
##     Min       1Q   Median       3Q      Max  
## -1.0480  -0.9414  -0.6393   0.6317   2.4446  
## 
## Coefficients:
##                     Estimate Std. Error z value Pr(>|z|)    
## (Intercept)         -1.62496    0.38048  -4.271 1.95e-05 ***
## GenusErigone         0.82256    0.45387   1.812  0.06994 .  
## GenusMicrolinyphia   0.72957    0.49876   1.463  0.14353    
## GenusPardosa         1.35077    0.60348   2.238  0.02520 *  
## GenusTenuiphantes    0.91208    0.40884   2.231  0.02569 *  
## Day2s               -0.04171    0.11362  -0.367  0.71357    
## MaturityJuvenile    -0.86577    0.33148  -2.612  0.00901 ** 
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for poisson family taken to be 1)
## 
##     Null deviance: 212.03  on 243  degrees of freedom
## Residual deviance: 198.62  on 237  degrees of freedom
## AIC: 370.9
## 
## Number of Fisher Scoring iterations: 6
```

```
anova(IPm1)

## Analysis of Deviance Table
##
## Model: poisson, link: log
##
## Response: Predator
##
## Terms added sequentially (first to last)
##
##
##          Df Deviance Resid. Df Resid. Dev
## NULL                       243     212.03
## Genus     4   5.0241       239     207.01
## Day2s     1   0.6258       238     206.38
## Maturity  1   7.7640       237     198.62
```

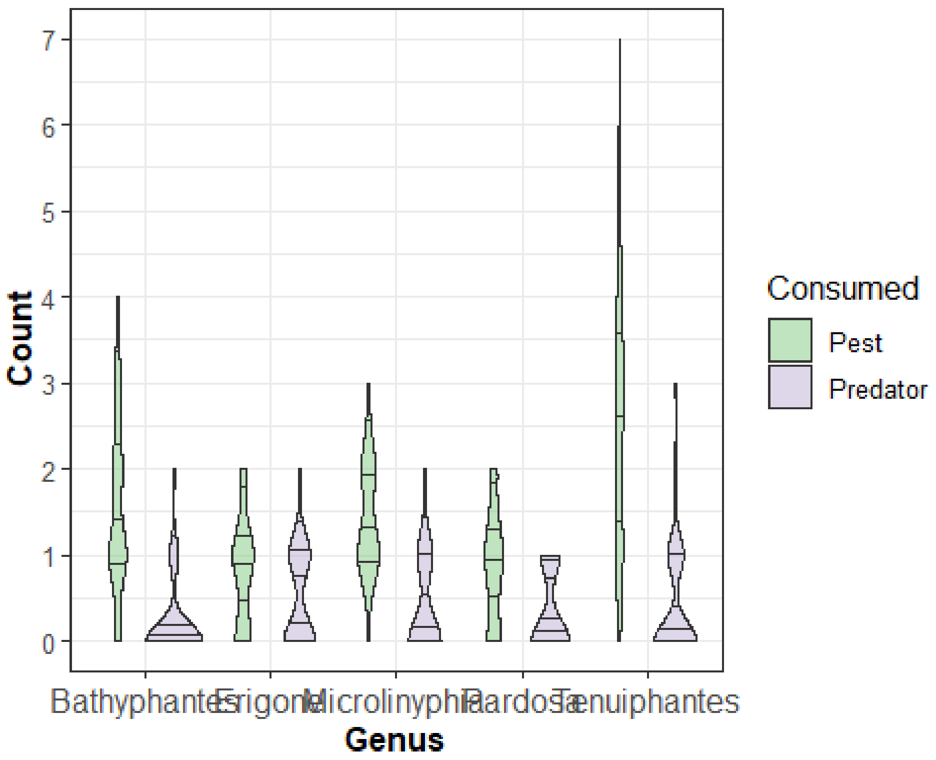
Again, we can relevel for a comprehensive understanding of the relationships between groups.

Visualising intraguild predation and biocontrol.

To highlight not only differences in the extent of intraguild predation and biocontrol between groups, but also how this dynamically changes (i.e. how many predators are eating how many pests), violin plots were used.

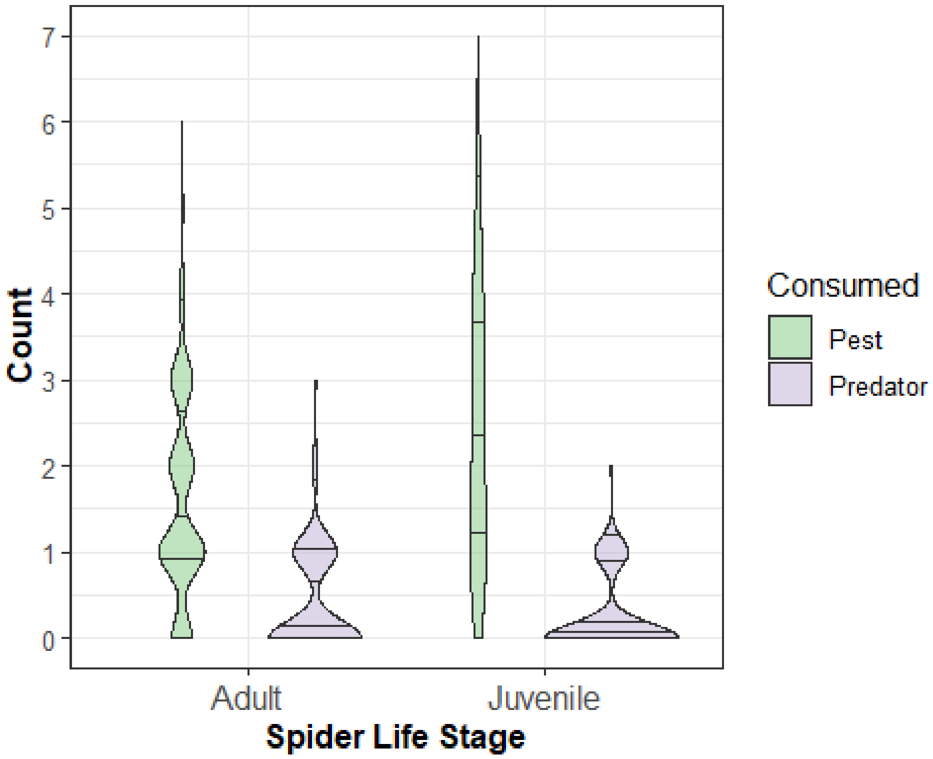
First for genera:

```
IPBCplot <- read.csv("IPBCcombiplot.csv")

violin_genus <- ggplot(IPBCplot, aes(x=Genus, y=Count, fill=Consumed)) +
  geom_violin(alpha=0.5, draw_quantiles = c(.25, .5, .75, .95)) + theme_bw()
+ scale_y_continuous(name="Count", breaks=seq(0,7,1), limits = c(0,7)) + scal
e_x_discrete(name="Genus")+
  #geom_jitter(shape=16, position=position_jitter(0.5), aes(alpha=0.01, colou
r=Consumed, fill=Consumed))+
  theme(text = element_text(size = 12),
        axis.title = element_text(face="bold"),
        axis.text.x=element_text(size = 12)) +
  scale_fill_brewer(palette = "Accent")

violin_genus
```



Then for life stage:

```
violin_life <- ggplot(IPBCplot, aes(x=Maturity, y=Count, fill=Consumed)) +
  geom_violin(alpha=0.5, draw_quantiles = c(.25, .5, .75, .95)) + theme_bw()
+ scale_y_continuous(name="Count", breaks=seq(0,7,1), limits = c(0,7)) + scal
e_x_discrete(name="Spider Life Stage")+
  #geom_jitter(shape=16, position=position_jitter(0.5), aes(alpha=0.01, colou
r=Consumed, fill=Consumed))+
  theme(text = element_text(size = 12),
        axis.title = element_text(face="bold"),
        axis.text.x=element_text(size = 12)) +
  scale_fill_brewer(palette = "Accent")

violin_life
```



Co-occurrence analysis

For co-occurrece analysis, we need to first create a co-occurrence matrix.

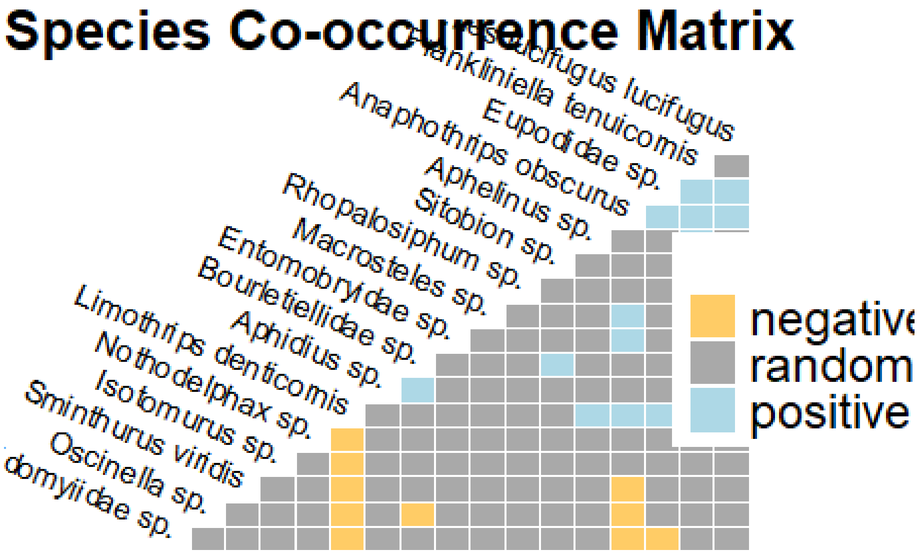
```
cooccurdiet <- read.csv("CooccurenceDiet.csv")
rownames(cooccurdiet) <- cooccurdiet[,1]
coocdiet <- cooccurdiet[,-1]

coocmat <- create.N.matrix(coocdiet)
```

We can then calculate the probabilities of co-occurrences based on this matrix using a null model.

```
diet.cooccur <- cooccur(coocdiet, type = "spp_site", spp_names = TRUE, true_r
and_classifier = 0.1, prob = "hyper",  site_mask = NULL, only_effects = FALSE
, eff_standard = TRUE, eff_matrix = FALSE, thresh=TRUE)

cooceff <- effect.sizes(diet.cooccur)
coocpro <- prob.table(diet.cooccur)

## Warning in prob.table(diet.cooccur): The co-occurrence model was run using
## 'thresh = TRUE.' The probability table may not include all species pairs
```

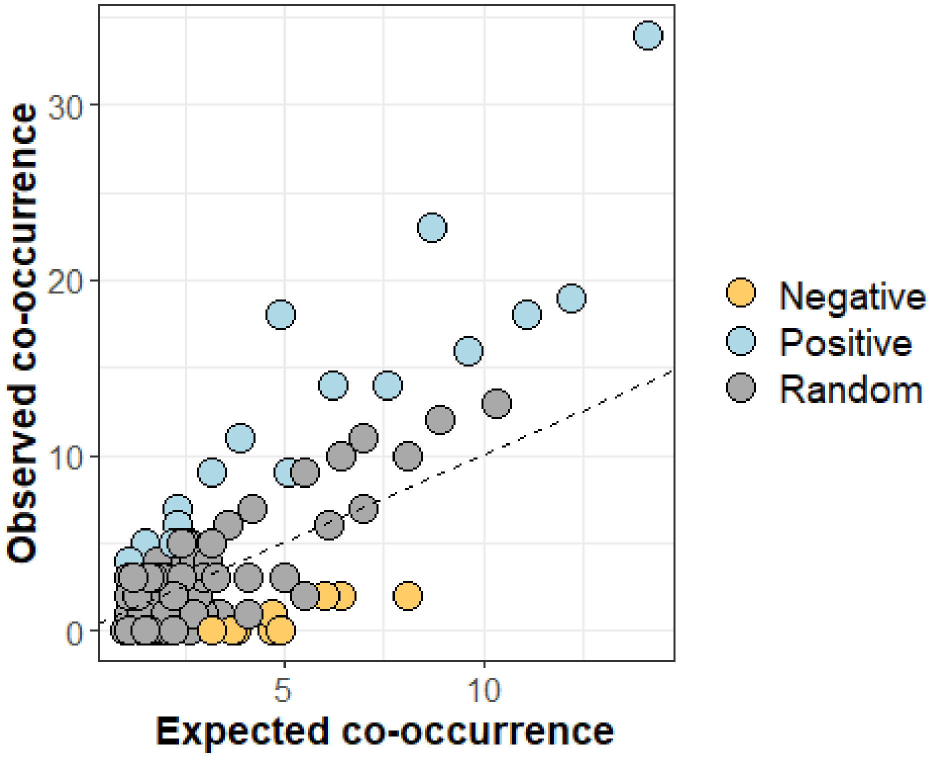
We can plot this as a matrix.

```
plot(diet.cooccur)
```



Or as the relationship between expected and observed co-occurrences.

```
df = diet.cooccur$results
df$type = "Random"
df$type[df$p_lt<0.05] = "Negative"
df$type[df$p_gt<0.05] = "Positive"

ove.co <- ggplot(df,aes(x=exp_cooccur,y=obs_cooccur)) +
  geom_point(aes(fill=type), pch=21, lwd=5) + geom_abline(linetype="dashed")
+
  #geom_label_repel(data=subset(df,sp1_name=="Geospiza magnirostris"),
                   # aes(label=paste(sp1_name,sp2_name,sep="\n")),
                   # size=2,nudge_x=-1,nudge_y=-1) +
  scale_fill_manual(values=c("#FFCC66","light blue","dark gray")) +
  theme_bw() + theme(axis.text=element_text(size=12), axis.title=element_text
(size=14, face="bold"), legend.title=element_blank(), legend.text=element_tex
t(size=14)) +
  labs(x = "Expected co-occurrence", y = "Observed co-occurrence")

ove.co
```



Dietary niche comparison

There are several ways of characterising a species' dietary niche, such as Levins niche breadth and Pianka niche overlap.

Levins niche breadth

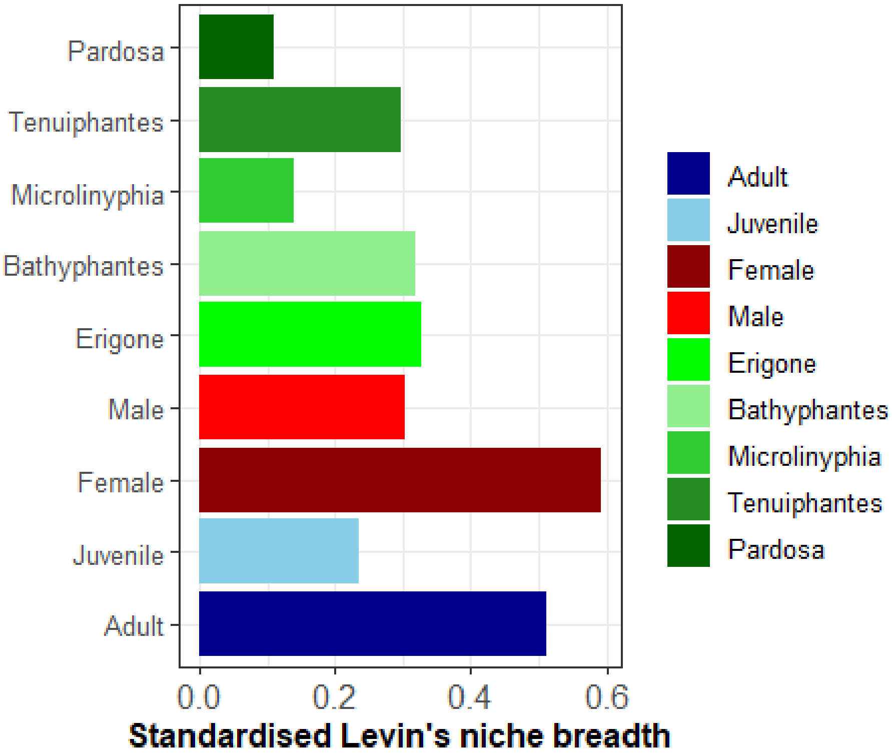
We can calculate niche breath for every group of spiders that we are interested in.

```
genniche <- read.csv("GenusNiche.csv")

rownames(genniche) <- genniche[,1]

genniche <- genniche[,-1]

genbreadth <- niche.width(genniche, method="levins")
genbreadth

##   Bathyphantes  Erigone Microlinyphia  Pardosa Tenuiphantes
## 1     16.95349 17.37433      7.943765 6.451327     15.85659

sexniche <- read.csv("SexNiche.csv")

rownames(sexniche) <- sexniche[,1]

sexniche <- sexniche[,-1]
```

```
sexbreadth <- niche.width(sexniche, method="levins")
sexbreadth

##      Female      Male     N.A
## 1 30.60531 16.15975 11.9011

matniche <- read.csv("MaturityNiche.csv")

rownames(matniche) <- matniche[,1]

matniche <- matniche[,-1]

matbreadth <- niche.width(matniche, method="levins")
matbreadth

##       Adult Juvenile
## 1 26.60966  12.7049
```

Once we have calculated these, we can standardise them in excel following equation one from Razgour et al. (2011), and then plot them.

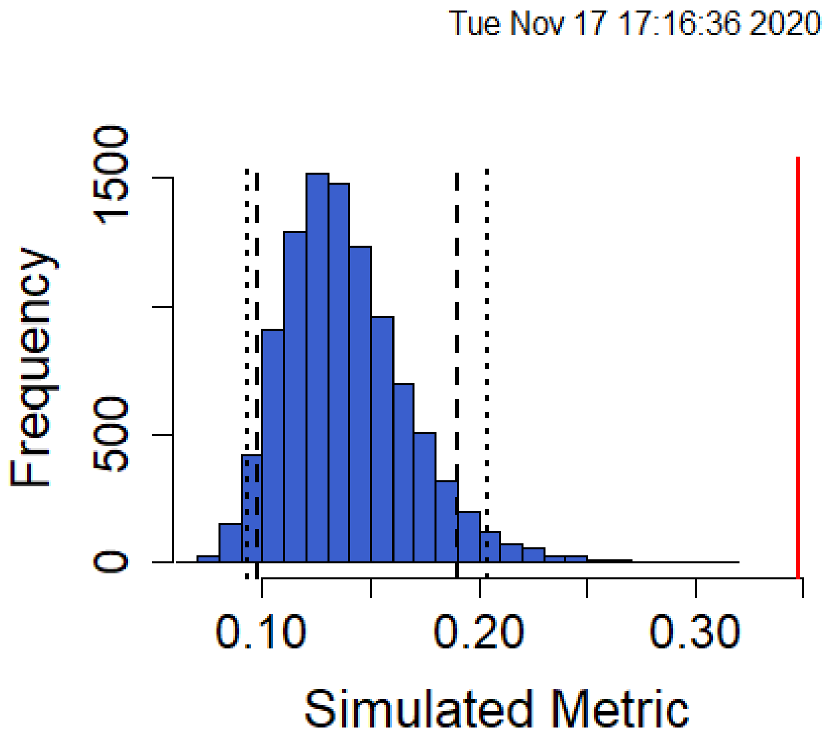
```
nichebreadth <- read.csv("NicheBreadth.csv")
nichebreadth <- nichebreadth[1:9,]

# re-order the levels in the order of appearance in the data.frame
nichebreadth$Spider <- factor(nichebreadth$Spiders, as.character(nichebreadth
$Spiders))
# same as
nichebreadth$Spider <- factor(nichebreadth$Spiders, c('Adult','Juvenile','Fem
ale','Male','Erigone','Bathyphantes','Microlinyphia','Tenuiphantes','Pardosa'
))

nwideplot <- ggplot() + geom_bar(data=nichebreadth, aes(x=Spider, y=StBreadth
, fill=Spiders), stat='identity') + coord_flip() +theme_bw() +
  scale_fill_manual("",
                    values=c('Adult'='darkblue','Juvenile'='skyblue','Female'
='darkred','Male'='red','Erigone'='green','Bathyphantes'='lightgreen','Microl
inyphia'='limegreen','Tenuiphantes'='forestgreen','Pardosa'='darkgreen'),
                    breaks=c('Adult','Juvenile','Female','Male','Erigone','Ba
thyphantes','Microlinyphia','Tenuiphantes','Pardosa'),
                    labels=c('Adult','Juvenile','Female','Male','Erigone','Ba
thyphantes','Microlinyphia','Tenuiphantes','Pardosa')) +
                    labs(y="Standardised Levin's niche breadth", x = "") +
  theme(text = element_text(size = 12),
        axis.title = element_text(face="bold"),
        axis.text.x=element_text(size = 12))

nwideplot
```



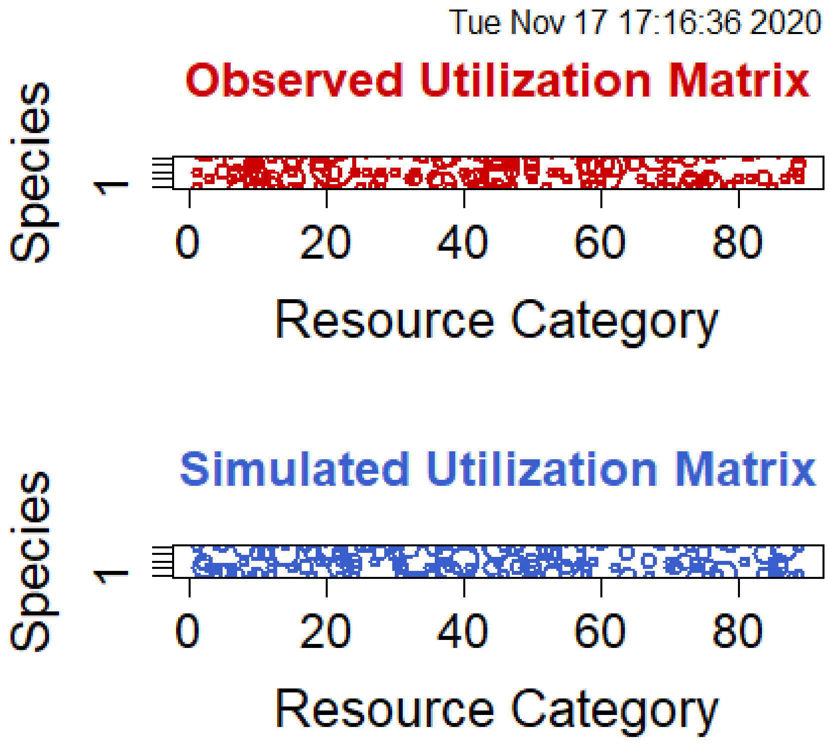
Pianka niche overlap

We can then assess niche overlap between groups of spiders, such as genera.

```
genover <- read.csv("GenusNicheFlip.csv")
rownames(genover) <- genover[,1]

genpianka <- niche_null_model(genover,
                              metric="pianka",
                              suppressProg=TRUE,nReps=9999)

plot(genpianka,type="hist")
```

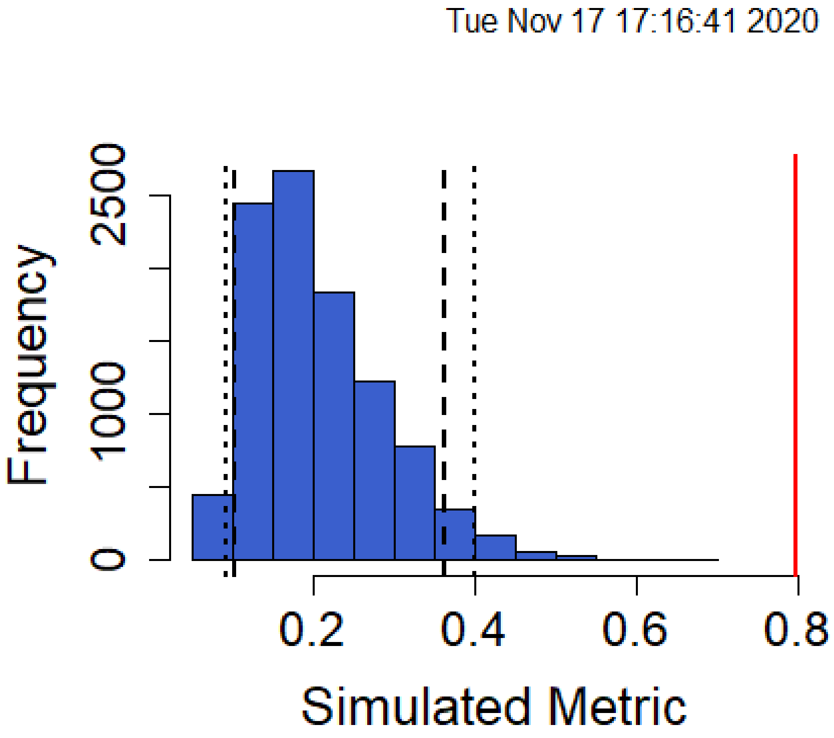


```
plot(genpianka,type="niche", ylab=Genus)
```

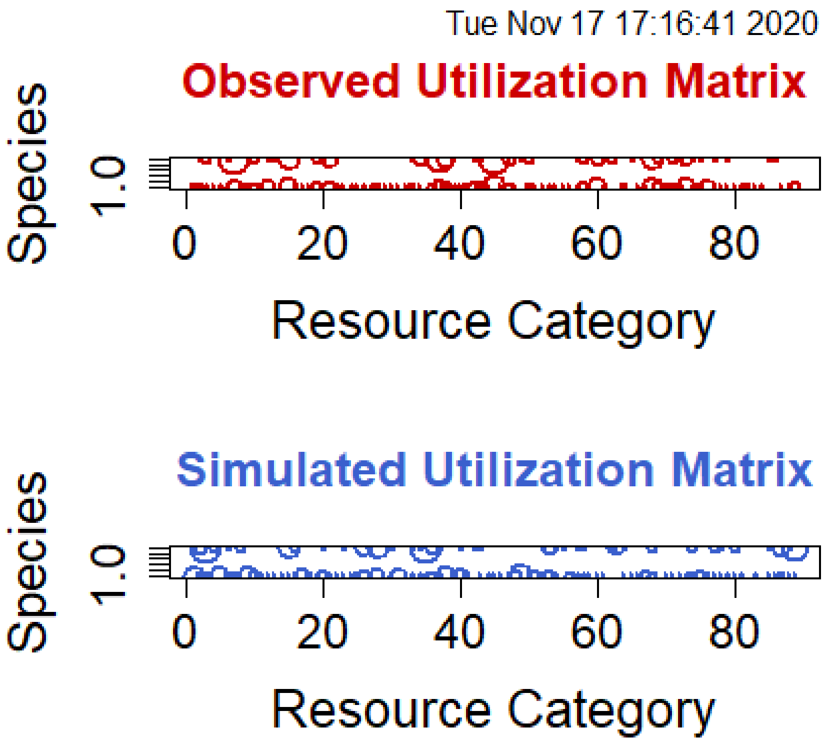


Or life stages.

```
matover <- read.csv("MaturityNicheFlip.csv")

matpianka <- niche_null_model(matover,
                              metric="pianka",
                              suppressProg=TRUE,nReps=9999)

plot(matpianka,type="hist")
```



```
plot(matpianka,type="niche")
```

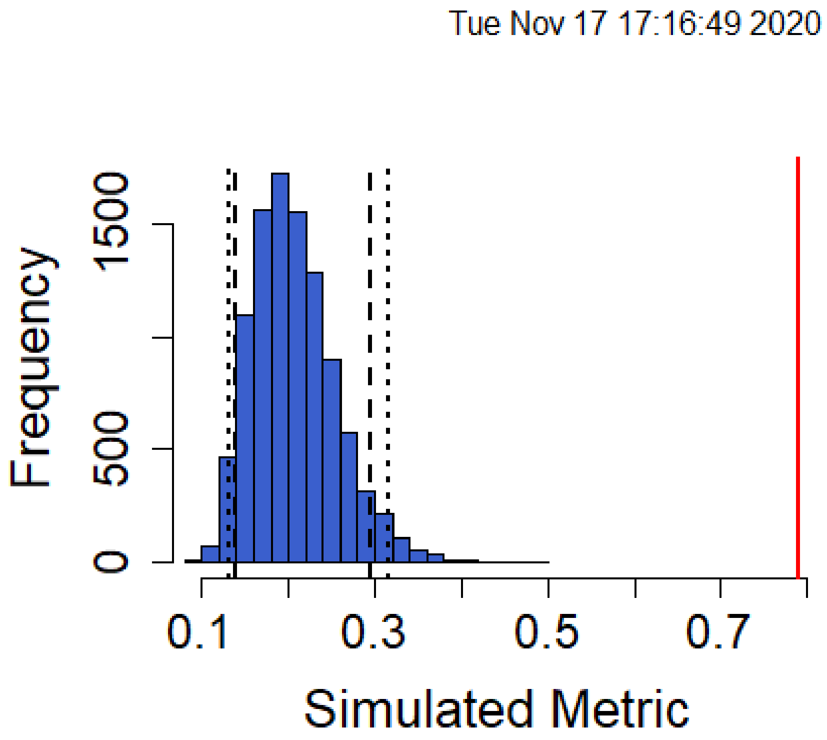

Tue Nov 17 17:16:41 2020

Observed Utilization Matrix

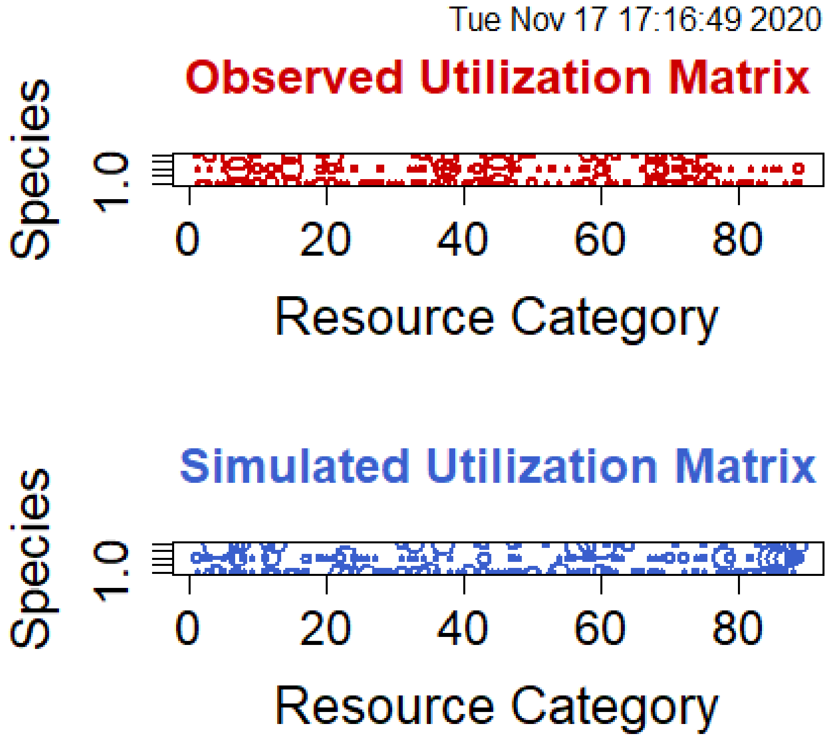
Simulated Utilization Matrix

Or sexes.

```
sexover <- read.csv("SexNicheFlip.csv")

sexpianka <- niche_null_model(sexover,
                              metric="pianka",
                              suppressProg=TRUE,nReps=9999)

plot(sexpianka,type="hist")
```



```
plot(sexpianka,type="niche")
```



Chapter 5 R Markdown

J. P. Cuff

17 November 2020

Chapter 5

Libraries

```
library('mvabund')
library('ggtern')

## Loading required package: ggplot2

## Registered S3 methods overwritten by 'ggtern':
##   method           from
##   grid.draw.ggplot ggplot2
##   plot.ggplot      ggplot2
##   print.ggplot     ggplot2

## --
## Remember to cite, run citation(package = 'ggtern') for further info.
## --

##
## Attaching package: 'ggtern'

## The following objects are masked from 'package:ggplot2':
##
##     aes, annotate, ggplot, ggplot_build, ggplot_gtable, ggplotGrob,
##     ggsave, layer_data, theme_bw, theme_classic, theme_dark,
##     theme_gray, theme_light, theme_linedraw, theme_minimal, theme_void

library('vegan')

## Loading required package: permute

## Loading required package: lattice

## This is vegan 2.5-6

library("flashClust")

##
## Attaching package: 'flashClust'

## The following object is masked from 'package:stats':
##
##     hclust
```

```
library("dendextend")

## Registered S3 method overwritten by 'dendextend':
##   method      from
##   rev.hclust vegan

##
## ---------------------
## Welcome to dendextend version 1.14.0
## Type citation('dendextend') for how to cite the package.
##
## Type browseVignettes(package = 'dendextend') for the package vignette.
## The github page is: https://github.com/talgalili/dendextend/
##
## Suggestions and bug-reports can be submitted at: https://github.com/talgal
ili/dendextend/issues
## Or contact: <tal.galili@gmail.com>
##
##  To suppress this message use:  suppressPackageStartupMessages(library(den
dextend))
## ---------------------

##
## Attaching package: 'dendextend'

## The following object is masked from 'package:permute':
##
##     shuffle

## The following object is masked from 'package:stats':
##
##     cutree

library("dplyr")

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library("ggplot2")
library("viridis")

## Loading required package: viridisLite
```

```
library("RColorBrewer")
library("cluster")
library("cooccur")
library("ggrepel")
library("clValid")
library("econullnetr")
library("gplots")

##
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
##
##     lowess

library("DHARMa")

## This is DHARMa 0.3.3.0. For overview type '?DHARMa'. For recent changes, t
ype news(package = 'DHARMa') Note: Syntax of plotResiduals has changed in 0.3
.0, see ?plotResiduals for details
```

Taxonomic ENNR

For 'econullnetr', we need to have matrices of prey at the same taxonomic levels for diet and prey community abundance.

ENNR family aggregation

We first need to aggregate the dietary data at family level.

```
InFamd_to_Agg <- read.csv("Diet_Fam_agg.csv")

Aggd <- aggregate(.~Taxon, data=InFamd_to_Agg, sum)

write.table(Aggd, "Diet_Fam_agged.csv")

InFami_to_Agg <- read.csv("FamInvertENNR agg.csv")

Aggi <- aggregate(.~Taxon, data=InFami_to_Agg, sum)

write.table(Aggi, "Invert_Fam_agged.csv")
```

Taxonomic ENNR for Genera

We can compare prey preferences between spider genera using a null model.

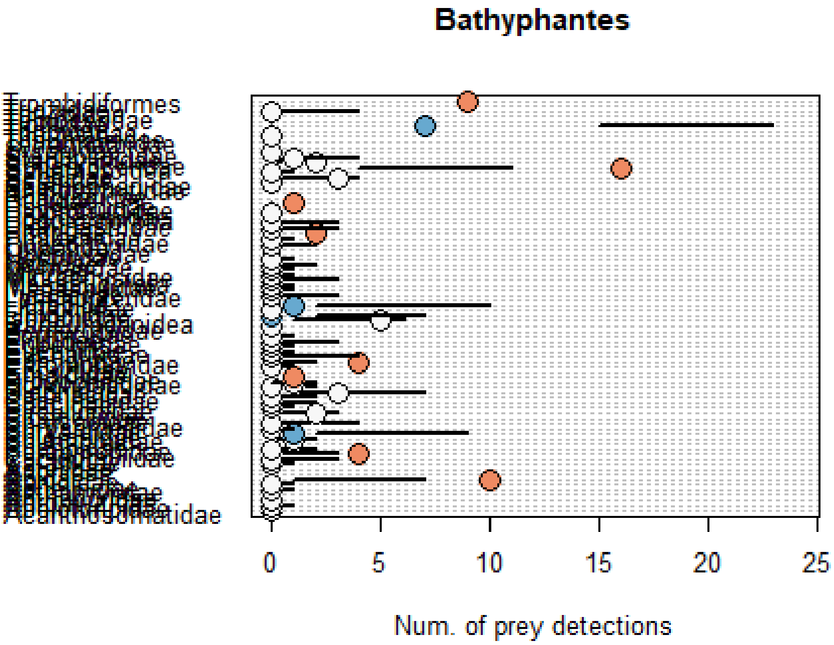
```
ennr <- read.csv("Fam_ENNR_Diet_Genusbin.csv")
invertsennr <- read.csv("Fam_ENNR_Inverts.csv")
ENNR.fl <- read.csv("Fam_ENNR_Diet.fl_Genus.csv")

genus.null <- generate_null_net(ennr[,2:83], invertsennr[,2:82],
                                sims = 999, data.type = "names",
                                summary.type = "sum",
                                r.samples = invertsennr[,1],
                                c.samples = ennr[,1],
                                r.weights = ENNR.fl)
```

```
## Warning in generate_null_net(ennr[, 2:83], invertsennr[, 2:82], sims = 999
, : One or more instances detected where a consumer interacted with a
##        resource that has zero abundance in 'resources'
```

We can then plot the overall outputs for each genus.

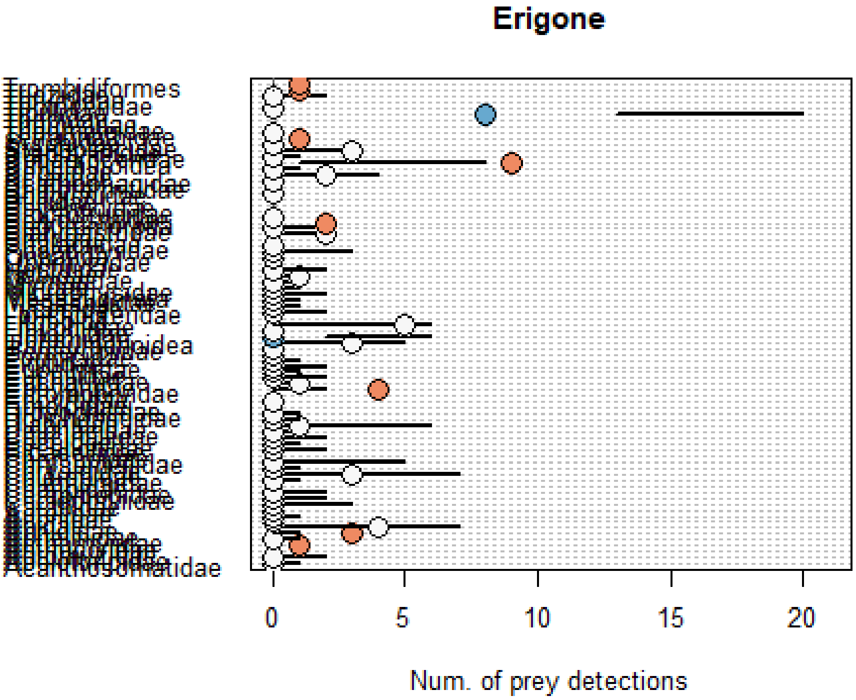
```
plot_preferences(genus.null, "Bathyphantes", signif.level = 0.95, type = "cou
nts",
                 xlab = "Num. of prey detections", p.cex = 1.5, l.cex = 0.8,
lwd = 2)
```

```
## Warning in test_interactions(nullnet, signif.level = signif.level): Be car
eful
## of Type I errors due to the large number of tests
```

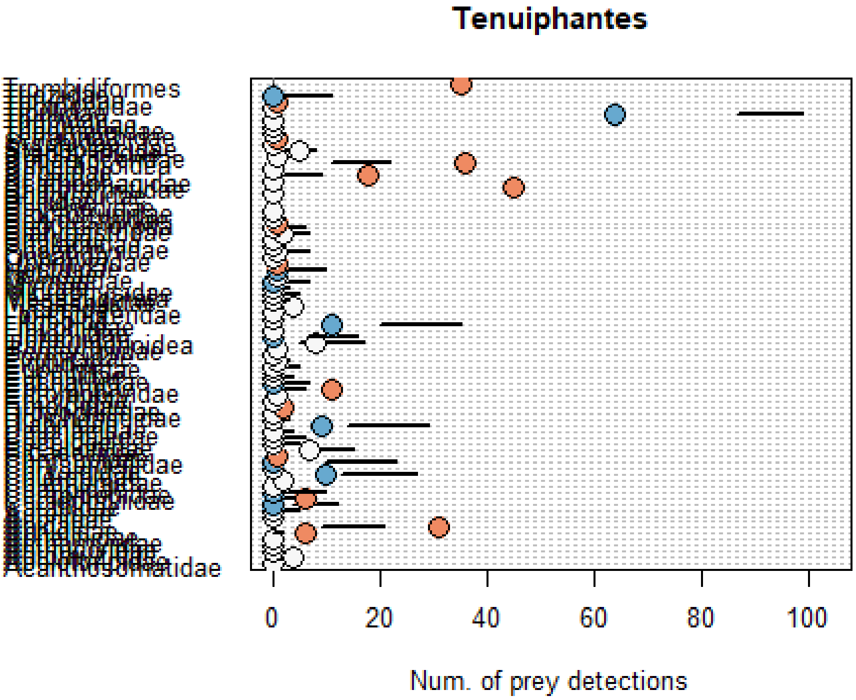


```
plot_preferences(genus.null, "Erigone", signif.level = 0.95, type = "counts",
                 xlab = "Num. of prey detections", p.cex = 1.5, l.cex = 0.8,
lwd = 2)
```

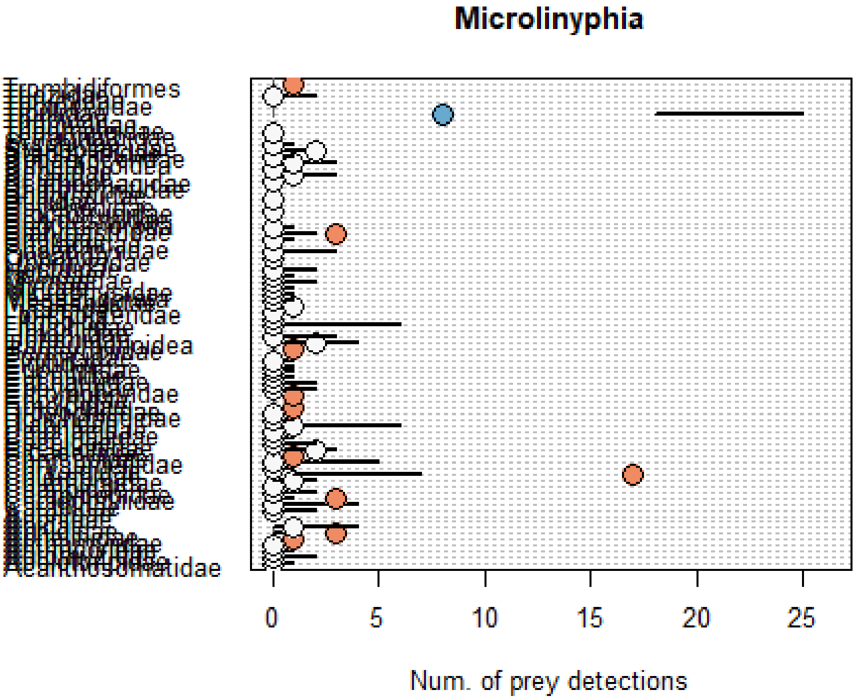
```
## Warning in test_interactions(nullnet, signif.level = signif.level): Be car
eful
## of Type I errors due to the large number of tests
```



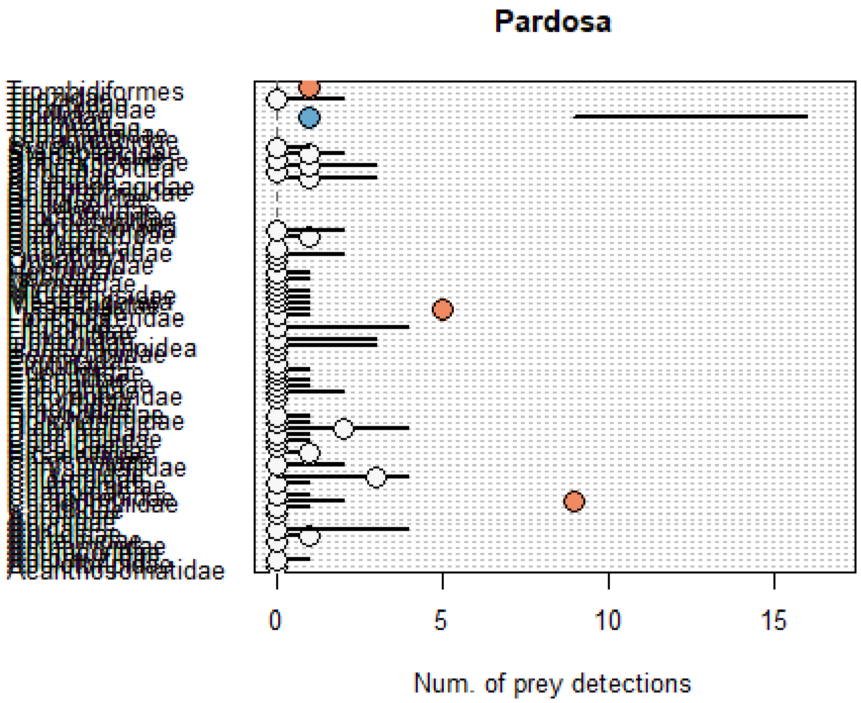
```
plot_preferences(genus.null, "Tenuiphantes", signif.level = 0.95, type = "cou
nts",
                 xlab = "Num. of prey detections", p.cex = 1.5, l.cex = 0.8,
lwd = 2)

## Warning in test_interactions(nullnet, signif.level = signif.level): Be car
eful
## of Type I errors due to the large number of tests
```



```
plot_preferences(genus.null, "Microlinyphia", signif.level = 0.95, type = "co
unts",
                 xlab = "Num. of prey detections", p.cex = 1.5, l.cex = 0.8,
lwd = 2)

## Warning in test_interactions(nullnet, signif.level = signif.level): Be car
eful
## of Type I errors due to the large number of tests
```



```
plot_preferences(genus.null, "Pardosa", signif.level = 0.95, type = "counts",
                 xlab = "Num. of prey detections", p.cex = 1.5, l.cex = 0.8,
lwd = 2)

## Warning in test_interactions(nullnet, signif.level = signif.level): Be car
eful
## of Type I errors due to the large number of tests
```



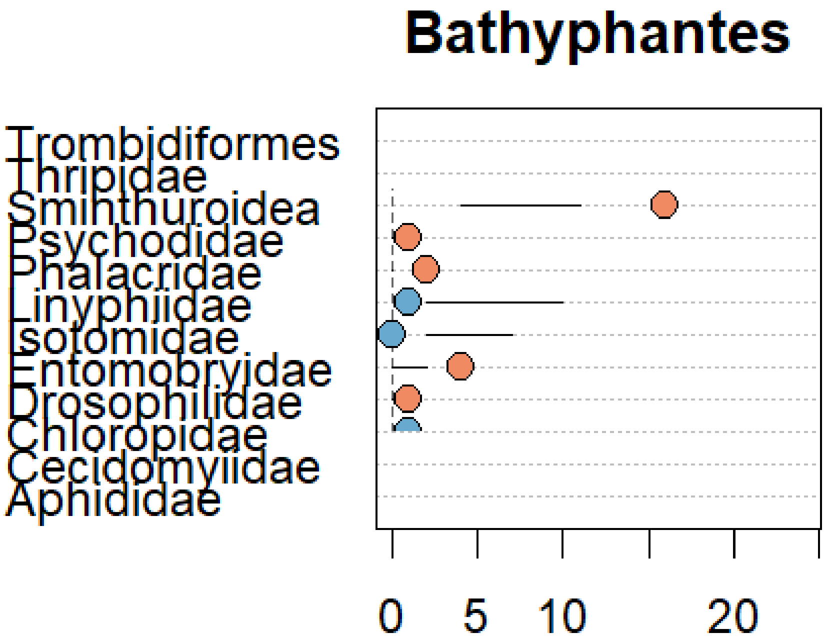
We can then extract the data output.

```
gen.links <- test_interactions(genus.null, signif.level = 0.95)

## Warning in test_interactions(genus.null, signif.level = 0.95): Be careful
of
## Type I errors due to the large number of tests
```

And then produce plots of just the significant results for each genus.

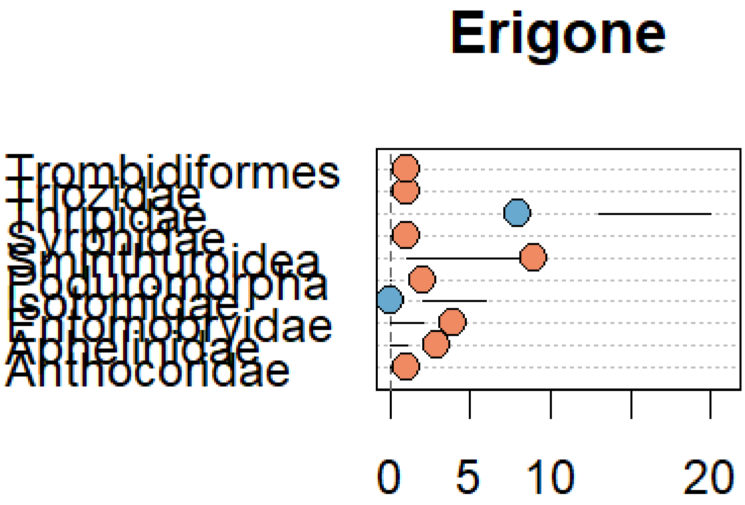
```
# Bathyphantes

gbti <- test_interactions(genus.null, signif.level = 0.95)

## Warning in test_interactions(genus.null, signif.level = 0.95): Be careful
of
## Type I errors due to the large number of tests

gbti <- gbti[gbti$Consumer == "Bathyphantes", ]
gbti[, 3] <- ifelse(rowSums(gbti[, 3:6]) == 0, NA, gbti[, 3])
gbti[, 4] <- ifelse(rowSums(gbti[, 3:6]) == 0, NA, gbti[, 4])
gbti[, 5] <- ifelse(rowSums(gbti[, 3:6]) == 0, NA, gbti[, 5])
gbti[, 6] <- ifelse(rowSums(gbti[, 3:6]) == 0, NA, gbti[, 6])

# EDIT 'ti' - to just the prey taxa that you want to show on the plot

gbti <- gbti[c(7,12,16,27,30,39,41,55,61,68,76,81),]
```

```
# Set up maximum x-axis value for xlim. Add an additional 5%
gbmin.x <- min(gbti[, 3:6], na.rm = TRUE)
gbmin.x <- max(0, gbmin.x, na.rm = TRUE)
gbmax.x <- max(gbti[, 3:6], na.rm = TRUE)
gbmax.x <- gbmax.x * 1.05
gbti$Setup <- seq(gbmin.x, gbmax.x, length.out = nrow(gbti))

# Plot built up in 2 stages: i) using min and max values to set the
#   y-axis range without having to use ylim (so this can be customised
#   by the user), ii) the main dbarplot and label, and iii) the error
graphics::dotchart(gbti$Setup, labels = paste(gbti$Resource, " ", sep = ""),
                   col = 1, pt.cex = 0, cex = 1.5, main = "Bathyphantes")
graphics::abline(v = 0, lty = 2, col = "dimgrey")


res.col <- c("#67A9CF", "#F7F7F7", "#EF8A62")

for (i in 1:nrow(gbti)){
  eval(parse(text = paste("lines(x = c(gbti$Lower.", 0.95 * 100,
                          ".CL[i], gbti$Upper.", 0.95 * 100,
                          ".CL[i]), y = c(i, i))", sep = "")))
  if(gbti$Test[i] == "Weaker") p.col <- res.col[1]
  if(gbti$Test[i] == "ns" | is.na(gbti$Test[i])) p.col <- res.col[2]
  if(gbti$Test[i] == "Stronger") p.col <- res.col[3]
  graphics::points(gbti$Observed[i], i, pch = 21, col = "black",
                   bg = p.col, cex = 2)
}
```



```
# Erigone

geti <- test_interactions(genus.null, signif.level = 0.95)

## Warning in test_interactions(genus.null, signif.level = 0.95): Be careful
of
## Type I errors due to the large number of tests

geti <- geti[geti$Consumer == "Erigone", ]
geti[, 3] <- ifelse(rowSums(geti[, 3:6]) == 0, NA, geti[, 3])
geti[, 4] <- ifelse(rowSums(geti[, 3:6]) == 0, NA, geti[, 4])
geti[, 5] <- ifelse(rowSums(geti[, 3:6]) == 0, NA, geti[, 5])
geti[, 6] <- ifelse(rowSums(geti[, 3:6]) == 0, NA, geti[, 6])

# EDIT 'ti' - to just the prey taxa that you want to show on the plot

geti <- geti[c(4,6,30,39,58,68,72,76,80,81),]


# Set up maximum x-axis value for xlim. Add an additional 5%
gemin.x <- min(geti[, 3:6], na.rm = TRUE)
gemin.x <- max(0, gemin.x, na.rm = TRUE)
gemax.x <- max(geti[, 3:6], na.rm = TRUE)
gemax.x <- gemax.x * 1.05
geti$Setup <- seq(gemin.x, gemax.x, length.out = nrow(geti))
```

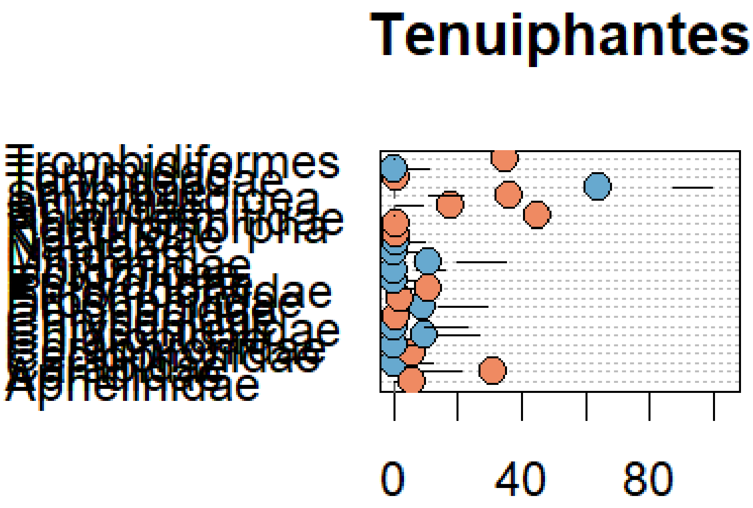
```
# Plot built up in 2 stages: i) using min and max values to set the
#   y-axis range without having to use ylim (so this can be customised
#   by the user), ii) the main dbarplot and label, and iii) the error
graphics::dotchart(geti$Setup, labels = paste(geti$Resource, " ", sep = ""),
                   col = 1, pt.cex = 0, cex = 1.5, main = "Erigone")
graphics::abline(v = 0, lty = 2, col = "dimgrey")


res.col <- c("#67A9CF", "#F7F7F7", "#EF8A62")

for (i in 1:nrow(geti)){
  eval(parse(text = paste("lines(x = c(geti$Lower.", 0.95 * 100,
                          ".CL[i], geti$Upper.", 0.95 * 100,
                          ".CL[i]), y = c(i, i))", sep = "")))
  if(geti$Test[i] == "Weaker") p.col <- res.col[1]
  if(geti$Test[i] == "ns" | is.na(geti$Test[i])) p.col <- res.col[2]
  if(geti$Test[i] == "Stronger") p.col <- res.col[3]
  graphics::points(geti$Observed[i], i, pch = 21, col = "black",
                   bg = p.col, cex = 2)
}
```



```
# Tenuiphantes

gtti <- test_interactions(genus.null, signif.level = 0.95)
```

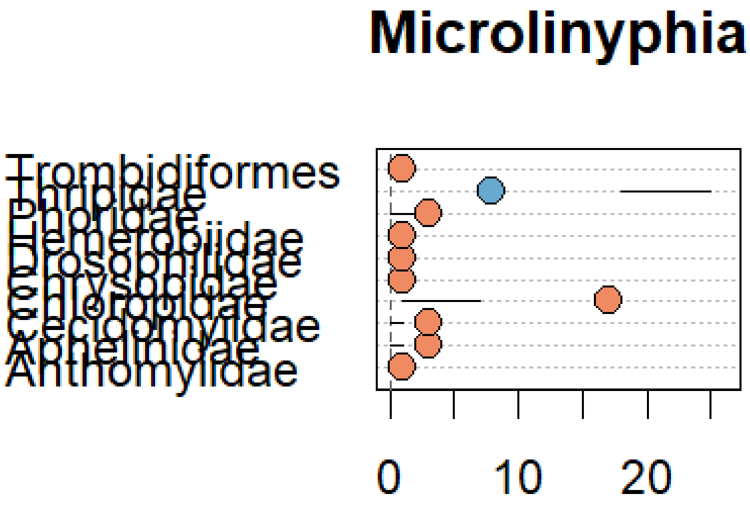
```
## Warning in test_interactions(genus.null, signif.level = 0.95): Be careful
of
## Type I errors due to the large number of tests
```

```r
gtti <- gtti[gtti$Consumer == "Tenuiphantes", ]
gtti[, 3] <- ifelse(rowSums(gtti[, 3:6]) == 0, NA, gtti[, 3])
gtti[, 4] <- ifelse(rowSums(gtti[, 3:6]) == 0, NA, gtti[, 4])
gtti[, 5] <- ifelse(rowSums(gtti[, 3:6]) == 0, NA, gtti[, 5])
gtti[, 6] <- ifelse(rowSums(gtti[, 3:6]) == 0, NA, gtti[, 6])

# EDIT 'ti' - to just the prey taxa that you want to show on the plot

gtti <- gtti[c(6,7,11,12,13,16,18,19,24,27,30,31,39,41,48,50,51,58,64,66,68,7
6,78,79,81),]


# Set up maximum x-axis value for xlim. Add an additional 5%
gtmin.x <- min(gtti[, 3:6], na.rm = TRUE)
gtmin.x <- max(0, gtmin.x, na.rm = TRUE)
gtmax.x <- max(gtti[, 3:6], na.rm = TRUE)
gtmax.x <- gtmax.x * 1.05
gtti$Setup <- seq(gtmin.x, gtmax.x, length.out = nrow(gtti))

# Plot built up in 2 stages: i) using min and max values to set the
#   y-axis range without having to use ylim (so this can be customised
#   by the user), ii) the main dbarplot and label, and iii) the error
graphics::dotchart(gtti$Setup, labels = paste(gtti$Resource, " ", sep = ""),
                   col = 1, pt.cex = 0, cex = 1.5, main = "Tenuiphantes")
graphics::abline(v = 0, lty = 2, col = "dimgrey")


res.col <- c("#67A9CF", "#F7F7F7", "#EF8A62")

for (i in 1:nrow(gtti)){
  eval(parse(text = paste("lines(x = c(gtti$Lower.", 0.95 * 100,
                          ".CL[i], gtti$Upper.", 0.95 * 100,
                          ".CL[i]), y = c(i, i))", sep = "")))
  if(gtti$Test[i] == "Weaker") p.col <- res.col[1]
  if(gtti$Test[i] == "ns" | is.na(gtti$Test[i])) p.col <- res.col[2]
  if(gtti$Test[i] == "Stronger") p.col <- res.col[3]
  graphics::points(gtti$Observed[i], i, pch = 21, col = "black",
                   bg = p.col, cex = 2)
}
```



```
# Microlinyphia

gmti <- test_interactions(genus.null, signif.level = 0.95)

## Warning in test_interactions(genus.null, signif.level = 0.95): Be careful
of
## Type I errors due to the large number of tests

gmti <- gmti[gmti$Consumer == "Microlinyphia", ]
gmti[, 3] <- ifelse(rowSums(gmti[, 3:6]) == 0, NA, gmti[, 3])
gmti[, 4] <- ifelse(rowSums(gmti[, 3:6]) == 0, NA, gmti[, 4])
gmti[, 5] <- ifelse(rowSums(gmti[, 3:6]) == 0, NA, gmti[, 5])
gmti[, 6] <- ifelse(rowSums(gmti[, 3:6]) == 0, NA, gmti[, 6])

# EDIT 'ti' - to just the prey taxa that you want to show on the plot

gmti <- gmti[c(5,6,12,16,19,27,37,56,76,81),]


# Set up maximum x-axis value for xlim. Add an additional 5%
gmmin.x <- min(gmti[, 3:6], na.rm = TRUE)
gmmin.x <- max(0, gmmin.x, na.rm = TRUE)
gmmax.x <- max(gmti[, 3:6], na.rm = TRUE)
gmmax.x <- gmmax.x * 1.05
gmti$Setup <- seq(gmmin.x, gmmax.x, length.out = nrow(gmti))
```

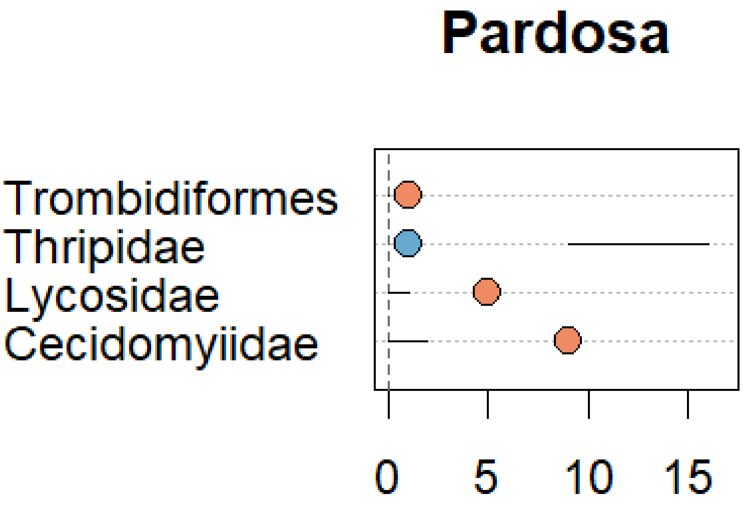
```
# Plot built up in 2 stages: i) using min and max values to set the
#   y-axis range without having to use ylim (so this can be customised
#   by the user), ii) the main dbarplot and label, and iii) the error
graphics::dotchart(gmti$Setup, labels = paste(gmti$Resource, " ", sep = ""),
                   col = 1, pt.cex = 0, cex = 1.5, main = "Microlinyphia")
graphics::abline(v = 0, lty = 2, col = "dimgrey")


res.col <- c("#67A9CF", "#F7F7F7", "#EF8A62")

for (i in 1:nrow(gmti)){
  eval(parse(text = paste("lines(x = c(gmti$Lower.", 0.95 * 100,
                          ".CL[i], gmti$Upper.", 0.95 * 100,
                          ".CL[i]), y = c(i, i))", sep = "")))
  if(gmti$Test[i] == "Weaker") p.col <- res.col[1]
  if(gmti$Test[i] == "ns" | is.na(gmti$Test[i])) p.col <- res.col[2]
  if(gmti$Test[i] == "Stronger") p.col <- res.col[3]
  graphics::points(gmti$Observed[i], i, pch = 21, col = "black",
                   bg = p.col, cex = 2)
}
```



```
# Pardosa

gpti <- test_interactions(genus.null, signif.level = 0.95)
```

```
## Warning in test_interactions(genus.null, signif.level = 0.95): Be careful
of
## Type I errors due to the large number of tests
```

```r
gpti <- gpti[gpti$Consumer == "Pardosa", ]
gpti[, 3] <- ifelse(rowSums(gpti[, 3:6]) == 0, NA, gpti[, 3])
gpti[, 4] <- ifelse(rowSums(gpti[, 3:6]) == 0, NA, gpti[, 4])
gpti[, 5] <- ifelse(rowSums(gpti[, 3:6]) == 0, NA, gpti[, 5])
gpti[, 6] <- ifelse(rowSums(gpti[, 3:6]) == 0, NA, gpti[, 6])

# EDIT 'ti' - to just the prey taxa that you want to show on the plot

gpti <- gpti[c(12,44,76,81),]


# Set up maximum x-axis value for xlim. Add an additional 5%
gpmin.x <- min(gpti[, 3:6], na.rm = TRUE)
gpmin.x <- max(0, gpmin.x, na.rm = TRUE)
gpmax.x <- max(gpti[, 3:6], na.rm = TRUE)
gpmax.x <- gpmax.x * 1.05
gpti$Setup <- seq(gpmin.x, gpmax.x, length.out = nrow(gpti))

# Plot built up in 2 stages: i) using min and max values to set the
#   y-axis range without having to use ylim (so this can be customised
#   by the user), ii) the main dbarplot and label, and iii) the error
graphics::dotchart(gpti$Setup, labels = paste(gpti$Resource, " ", sep = ""),
                   col = 1, pt.cex = 0, cex = 1.5, main = "Pardosa")
graphics::abline(v = 0, lty = 2, col = "dimgrey")


res.col <- c("#67A9CF", "#F7F7F7", "#EF8A62")

for (i in 1:nrow(gpti)){
  eval(parse(text = paste("lines(x = c(gpti$Lower.", 0.95 * 100,
                          ".CL[i], gpti$Upper.", 0.95 * 100,
                          ".CL[i]), y = c(i, i))", sep = "")))
  if(gpti$Test[i] == "Weaker") p.col <- res.col[1]
  if(gpti$Test[i] == "ns" | is.na(gpti$Test[i])) p.col <- res.col[2]
  if(gpti$Test[i] == "Stronger") p.col <- res.col[3]
  graphics::points(gpti$Observed[i], i, pch = 21, col = "black",
                   bg = p.col, cex = 2)
}
```



Taxonomic ENNR for Sexes

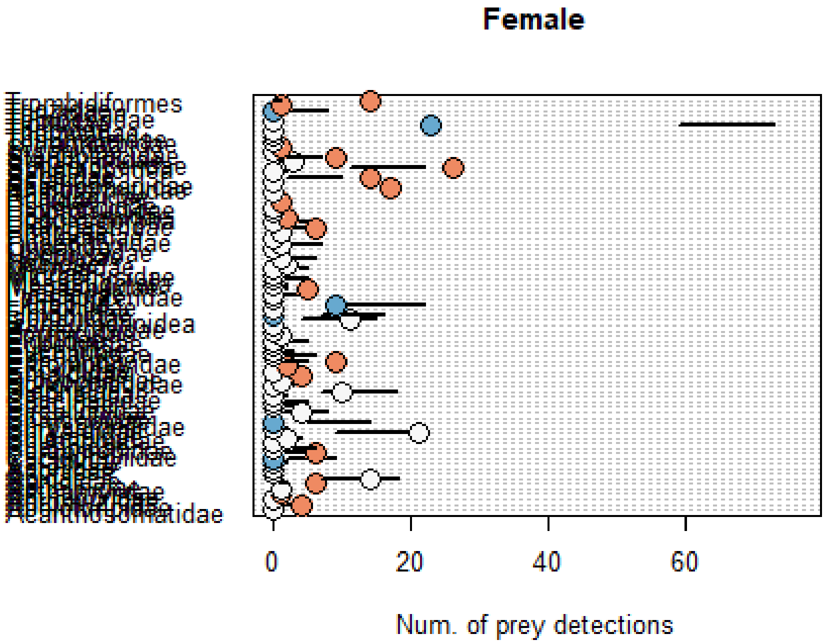
Again, we need to create the model.

```
sexennr <- read.csv("Fam_ENNR_Diet_Sexbin.csv")
invertsennr <- read.csv("Fam_ENNR_Inverts.csv")
sexENNR.fl <- read.csv("Fam_ENNR_Diet.fl_Sex.csv")

sex.null <- generate_null_net(sexennr[,2:83], invertsennr[,2:82],
                              sims = 999, data.type = "names",
                              summary.type = "sum",
                              r.samples = invertsennr[,1],
                              c.samples = sexennr[,1],
                              r.weights = sexENNR.fl)

## Warning in generate_null_net(sexennr[, 2:83], invertsennr[, 2:82], sims =
999, : One or more instances detected where a consumer interacted with a
##       resource that has zero abundance in 'resources'
```

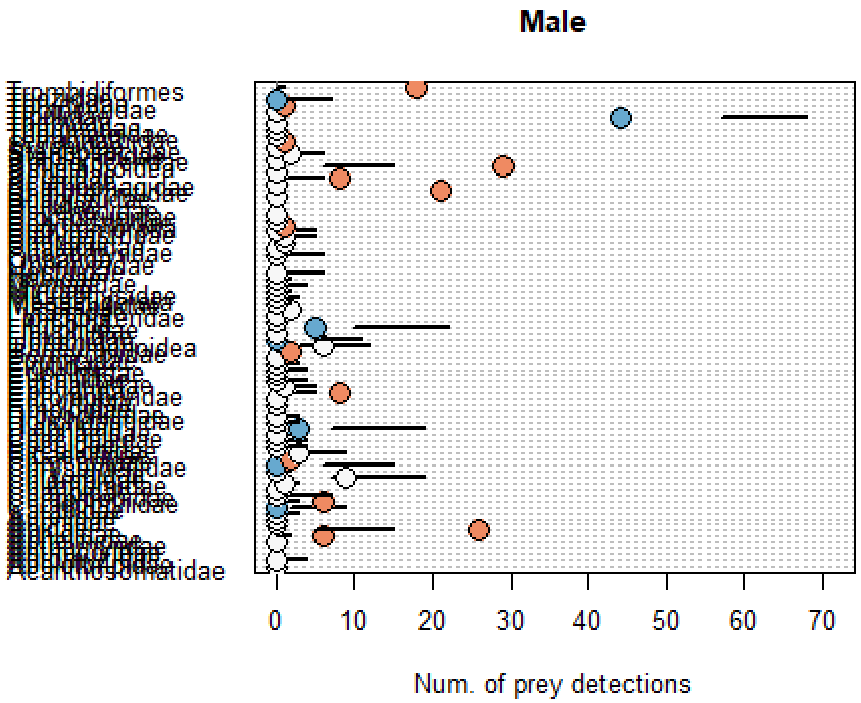
Then plot the overall preferences.

```
plot_preferences(sex.null, "Female", signif.level = 0.95, type = "counts",
                 xlab = "Num. of prey detections", p.cex = 1.5, l.cex = 0.8,
lwd = 2)
```

```
## Warning in test_interactions(nullnet, signif.level = signif.level): Be car
eful
## of Type I errors due to the large number of tests
```



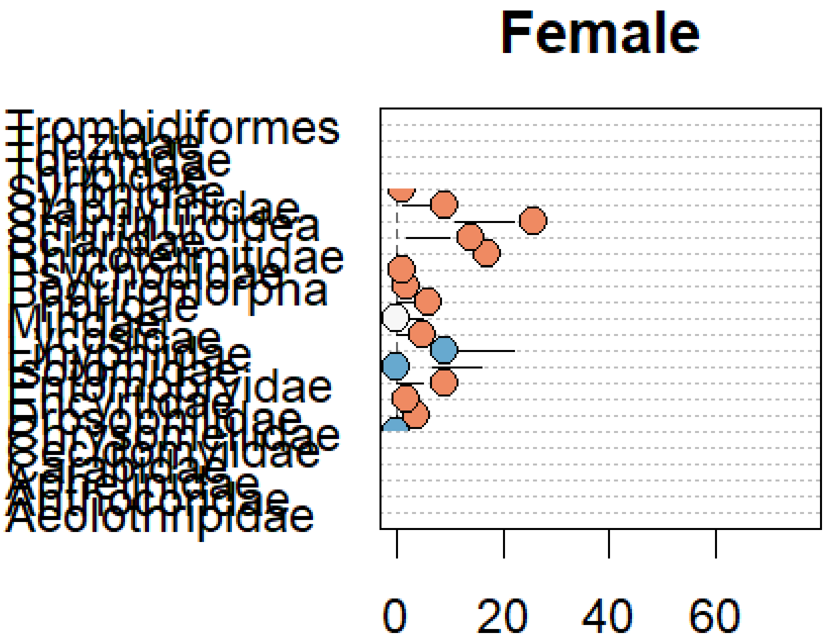
```
plot_preferences(sex.null, "Male", signif.level = 0.95, type = "counts",
                 xlab = "Num. of prey detections", p.cex = 1.5, l.cex = 0.8,
lwd = 2)

## Warning in test_interactions(nullnet, signif.level = signif.level): Be car
eful
## of Type I errors due to the large number of tests
```



Then the significant ones.

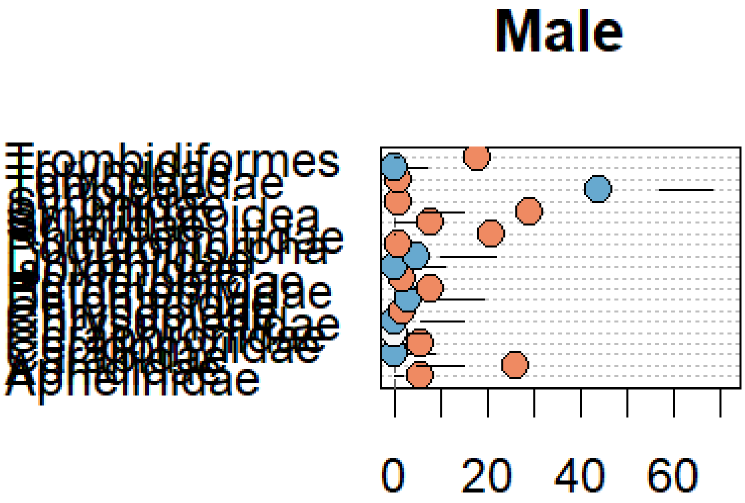
```
sex.links <- test_interactions(sex.null, signif.level = 0.95)

## Warning in test_interactions(sex.null, signif.level = 0.95): Be careful of
Type
## I errors due to the large number of tests

# Female

sfti <- test_interactions(sex.null, signif.level = 0.95)

## Warning in test_interactions(sex.null, signif.level = 0.95): Be careful of
Type
## I errors due to the large number of tests

sfti <- sfti[sfti$Consumer == "Female", ]
sfti[, 3] <- ifelse(rowSums(sfti[, 3:6]) == 0, NA, sfti[, 3])
sfti[, 4] <- ifelse(rowSums(sfti[, 3:6]) == 0, NA, sfti[, 4])
sfti[, 5] <- ifelse(rowSums(sfti[, 3:6]) == 0, NA, sfti[, 5])
sfti[, 6] <- ifelse(rowSums(sfti[, 3:6]) == 0, NA, sfti[, 6])

# EDIT 'ti' - to just the prey taxa that you want to show on the plot

sfti <- sfti[c(2,4,6,11,12,18,27,29,30,39,41,44,48,56,58,61,64,66,68,70,72,76
,79,80,81),]
```

```r
# Set up maximum x-axis value for xlim. Add an additional 5%
sfmin.x <- min(sfti[, 3:6], na.rm = TRUE)
sfmin.x <- max(0, sfmin.x, na.rm = TRUE)
sfmax.x <- max(sfti[, 3:6], na.rm = TRUE)
sfmax.x <- sfmax.x * 1.05
sfti$Setup <- seq(sfmin.x, sfmax.x, length.out = nrow(sfti))

# Plot built up in 2 stages: i) using min and max values to set the
#   y-axis range without having to use ylim (so this can be customised
#   by the user), ii) the main dbarplot and label, and iii) the error
graphics::dotchart(sfti$Setup, labels = paste(sfti$Resource, " ", sep = ""),
                   col = 1, pt.cex = 0, cex = 1.5, main = "Female")
graphics::abline(v = 0, lty = 2, col = "dimgrey")


res.col <- c("#67A9CF", "#F7F7F7", "#EF8A62")

for (i in 1:nrow(sfti)){
  eval(parse(text = paste("lines(x = c(sfti$Lower.", 0.95 * 100,
                          ".CL[i], sfti$Upper.", 0.95 * 100,
                          ".CL[i]), y = c(i, i))", sep = "")))
  if(sfti$Test[i] == "Weaker") p.col <- res.col[1]
  if(sfti$Test[i] == "ns" | is.na(sfti$Test[i])) p.col <- res.col[2]
  if(sfti$Test[i] == "Stronger") p.col <- res.col[3]
  graphics::points(sfti$Observed[i], i, pch = 21, col = "black",
                   bg = p.col, cex = 2)
}
```



```
# Male

smti <- test_interactions(sex.null, signif.level = 0.95)

## Warning in test_interactions(sex.null, signif.level = 0.95): Be careful of
Type
## I errors due to the large number of tests

smti <- smti[smti$Consumer == "Male", ]
smti[, 3] <- ifelse(rowSums(smti[, 3:6]) == 0, NA, smti[, 3])
smti[, 4] <- ifelse(rowSums(smti[, 3:6]) == 0, NA, smti[, 4])
smti[, 5] <- ifelse(rowSums(smti[, 3:6]) == 0, NA, smti[, 5])
smti[, 6] <- ifelse(rowSums(smti[, 3:6]) == 0, NA, smti[, 6])

# EDIT 'ti' - to just the prey taxa that you want to show on the plot

smti <- smti[c(6,7,11,12,13,18,19,24,30,37,39,41,58,64,66,68,72,76,78,79,81),
]


# Set up maximum x-axis value for xlim. Add an additional 5%
smmin.x <- min(smti[, 3:6], na.rm = TRUE)
smmin.x <- max(0, smmin.x, na.rm = TRUE)
smmax.x <- max(smti[, 3:6], na.rm = TRUE)
smmax.x <- smmax.x * 1.05
smti$Setup <- seq(smmin.x, smmax.x, length.out = nrow(smti))
```

```
# Plot built up in 2 stages: i) using min and max values to set the
#   y-axis range without having to use ylim (so this can be customised
#   by the user), ii) the main dbarplot and label, and iii) the error
graphics::dotchart(smti$Setup, labels = paste(smti$Resource, " ", sep = ""),
                   col = 1, pt.cex = 0, cex = 1.5, main = "Male")
graphics::abline(v = 0, lty = 2, col = "dimgrey")


res.col <- c("#67A9CF", "#F7F7F7", "#EF8A62")

for (i in 1:nrow(smti)){
  eval(parse(text = paste("lines(x = c(smti$Lower.", 0.95 * 100,
                          ".CL[i], smti$Upper.", 0.95 * 100,
                          ".CL[i]), y = c(i, i))", sep = "")))
  if(smti$Test[i] == "Weaker") p.col <- res.col[1]
  if(smti$Test[i] == "ns" | is.na(smti$Test[i])) p.col <- res.col[2]
  if(smti$Test[i] == "Stronger") p.col <- res.col[3]
  graphics::points(smti$Observed[i], i, pch = 21, col = "black",
                   bg = p.col, cex = 2)
}
```



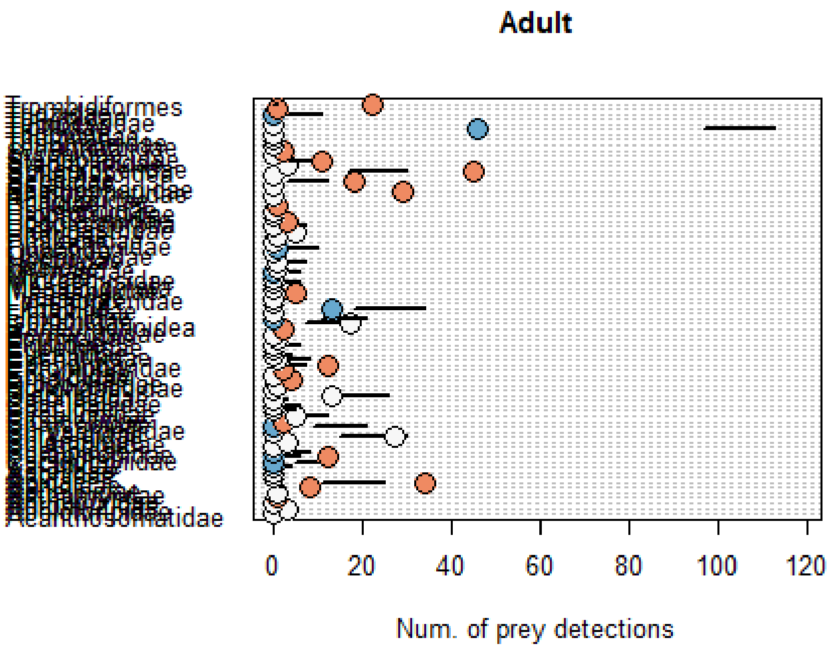
Taxonomic ENNR for Life stages

Again, model:

```
lifeennr <- read.csv("Fam_ENNR_Diet_Lifebin.csv")
invertsennr <- read.csv("Fam_ENNR_Inverts.csv")
lifeENNR.fl <- read.csv("Fam_ENNR_Diet.fl_Life.csv")

life.null <- generate_null_net(lifeennr[,2:83], invertsennr[,2:82],
                               sims = 999, data.type = "names",
                               summary.type = "sum",
                               r.samples = invertsennr[,1],
                               c.samples = lifeennr[,1],
                               r.weights = lifeENNR.fl)

## Warning in generate_null_net(lifeennr[, 2:83], invertsennr[, 2:82], sims =
999, : One or more instances detected where a consumer interacted with a
##        resource that has zero abundance in 'resources'
```

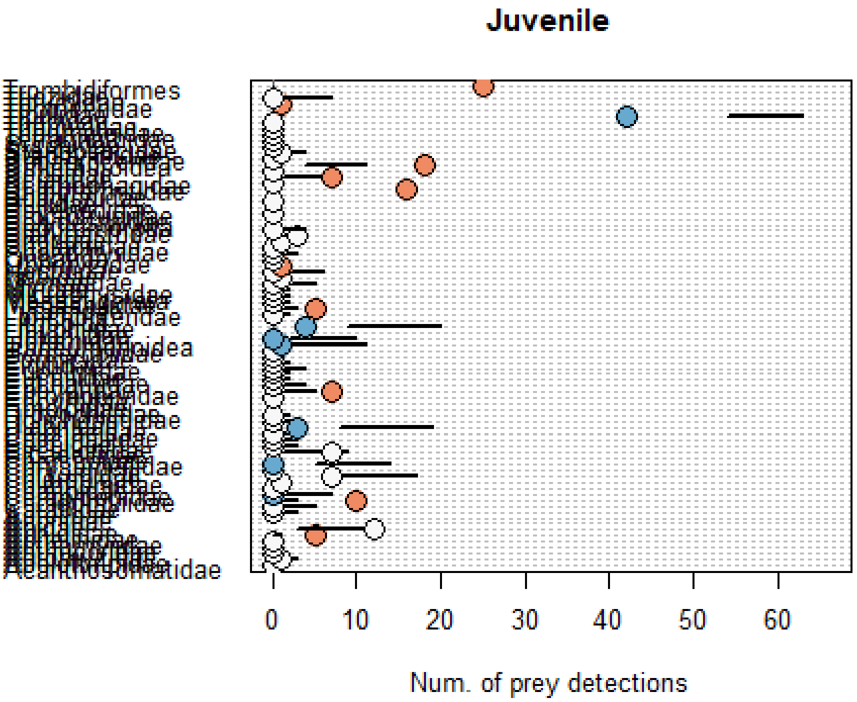
Overall plots:

```
plot_preferences(life.null, "Adult", signif.level = 0.95, type = "counts",
                 xlab = "Num. of prey detections", p.cex = 1.5, l.cex = 0.8,
lwd = 2)

## Warning in test_interactions(nullnet, signif.level = signif.level): Be car
eful
## of Type I errors due to the large number of tests
```



```
plot_preferences(life.null, "Juvenile", signif.level = 0.95, type = "counts",
                 xlab = "Num. of prey detections", p.cex = 1.5, l.cex = 0.8,
lwd = 2)
```

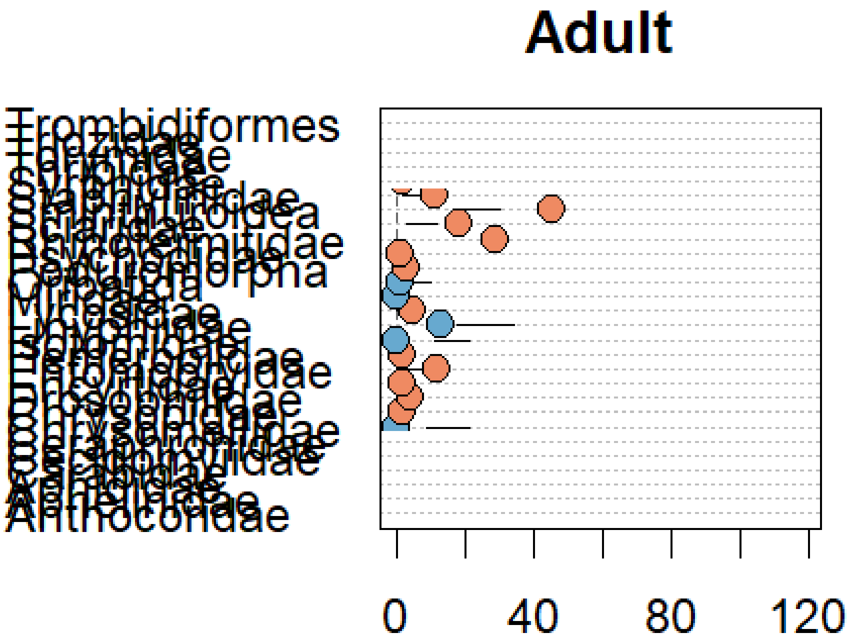
```
## Warning in test_interactions(nullnet, signif.level = signif.level): Be car
eful
## of Type I errors due to the large number of tests
```



And significant plots:

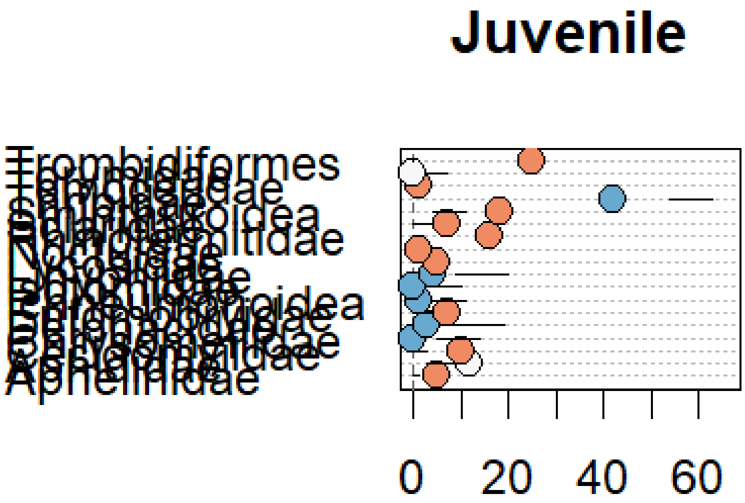
```
life.links <- test_interactions(life.null, signif.level = 0.95)

## Warning in test_interactions(life.null, signif.level = 0.95): Be careful o
f Type
## I errors due to the large number of tests

# Adult

lati <- test_interactions(life.null, signif.level = 0.95)

## Warning in test_interactions(life.null, signif.level = 0.95): Be careful o
f Type
## I errors due to the large number of tests

lati <- lati[lati$Consumer == "Adult", ]
lati[, 3] <- ifelse(rowSums(lati[, 3:6]) == 0, NA, lati[, 3])
lati[, 4] <- ifelse(rowSums(lati[, 3:6]) == 0, NA, lati[, 4])
lati[, 5] <- ifelse(rowSums(lati[, 3:6]) == 0, NA, lati[, 5])
lati[, 6] <- ifelse(rowSums(lati[, 3:6]) == 0, NA, lati[, 6])

# EDIT 'ti' - to just the prey taxa that you want to show on the plot
```

```r
lati <- lati[c(4,6,7,11,12,13,18,19,27,29,30,37,39,41,44,48,53,58,61,64,66,68
,70,72,76,79,80,81),]


# Set up maximum x-axis value for xlim. Add an additional 5%
lamin.x <- min(lati[, 3:6], na.rm = TRUE)
lamin.x <- max(0, lamin.x, na.rm = TRUE)
lamax.x <- max(lati[, 3:6], na.rm = TRUE)
lamax.x <- lamax.x * 1.05
lati$Setup <- seq(lamin.x, lamax.x, length.out = nrow(lati))

# Plot built up in 2 stages: i) using min and max values to set the
#   y-axis range without having to use ylim (so this can be customised
#   by the user), ii) the main dbarplot and label, and iii) the error
graphics::dotchart(lati$Setup, labels = paste(lati$Resource, " ", sep = ""),
                   col = 1, pt.cex = 0, cex = 1.5, main = "Adult")
graphics::abline(v = 0, lty = 2, col = "dimgrey")


res.col <- c("#67A9CF", "#F7F7F7", "#EF8A62")

for (i in 1:nrow(lati)){
  eval(parse(text = paste("lines(x = c(lati$Lower.", 0.95 * 100,
                          ".CL[i], lati$Upper.", 0.95 * 100,
                          ".CL[i]), y = c(i, i))", sep = "")))
  if(lati$Test[i] == "Weaker") p.col <- res.col[1]
  if(lati$Test[i] == "ns" | is.na(lati$Test[i])) p.col <- res.col[2]
  if(lati$Test[i] == "Stronger") p.col <- res.col[3]
  graphics::points(lati$Observed[i], i, pch = 21, col = "black",
                   bg = p.col, cex = 2)
}
```



```
# Juvenile

ljti <- test_interactions(life.null, signif.level = 0.95)

## Warning in test_interactions(life.null, signif.level = 0.95): Be careful o
f Type
## I errors due to the large number of tests

ljti <- ljti[ljti$Consumer == "Juvenile", ]
ljti[, 3] <- ifelse(rowSums(ljti[, 3:6]) == 0, NA, ljti[, 3])
ljti[, 4] <- ifelse(rowSums(ljti[, 3:6]) == 0, NA, ljti[, 4])
ljti[, 5] <- ifelse(rowSums(ljti[, 3:6]) == 0, NA, ljti[, 5])
ljti[, 6] <- ifelse(rowSums(ljti[, 3:6]) == 0, NA, ljti[, 6])

# EDIT 'ti' - to just the prey taxa that you want to show on the plot

ljti <- ljti[c(6,7,12,18,24,30,38,39,41,44,51,64,66,68,76,78,79,81),]


# Set up maximum x-axis value for xlim. Add an additional 5%
ljmin.x <- min(ljti[, 3:6], na.rm = TRUE)
ljmin.x <- max(0, ljmin.x, na.rm = TRUE)
ljmax.x <- max(ljti[, 3:6], na.rm = TRUE)
ljmax.x <- ljmax.x * 1.05
ljti$Setup <- seq(ljmin.x, ljmax.x, length.out = nrow(ljti))
```

```
# Plot built up in 2 stages: i) using min and max values to set the
#   y-axis range without having to use ylim (so this can be customised
#   by the user), ii) the main dbarplot and label, and iii) the error
graphics::dotchart(ljti$Setup, labels = paste(ljti$Resource, " ", sep = ""),
                   col = 1, pt.cex = 0, cex = 1.5, main = "Juvenile")
graphics::abline(v = 0, lty = 2, col = "dimgrey")


res.col <- c("#67A9CF", "#F7F7F7", "#EF8A62")

for (i in 1:nrow(ljti)){
  eval(parse(text = paste("lines(x = c(ljti$Lower.", 0.95 * 100,
                          ".CL[i], ljti$Upper.", 0.95 * 100,
                          ".CL[i]), y = c(i, i))", sep = "")))
  if(ljti$Test[i] == "Weaker") p.col <- res.col[1]
  if(ljti$Test[i] == "ns" | is.na(ljti$Test[i])) p.col <- res.col[2]
  if(ljti$Test[i] == "Stronger") p.col <- res.col[3]
  graphics::points(ljti$Observed[i], i, pch = 21, col = "black",
                   bg = p.col, cex = 2)
}
```



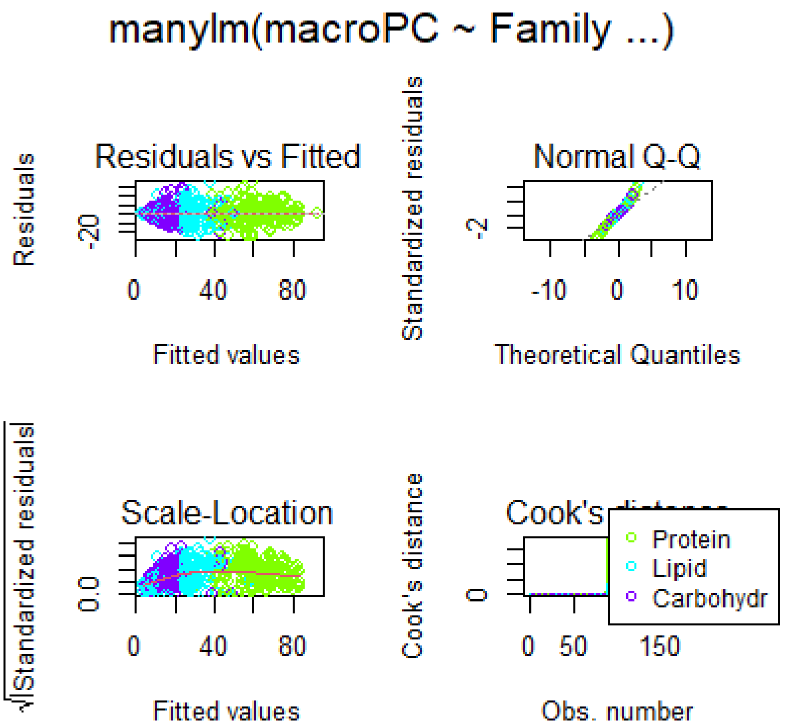
Macronutrient differences

We can compare macronutrient content between taxa as we did in Chapter 2 - using ternary plots and MLMs.

```
macro <- read.csv("macros.csv")
macro$Family <- as.factor(macro$Family)
macro$Order <- as.factor(macro$Order)
macro$Class <- as.factor(macro$Class)
```

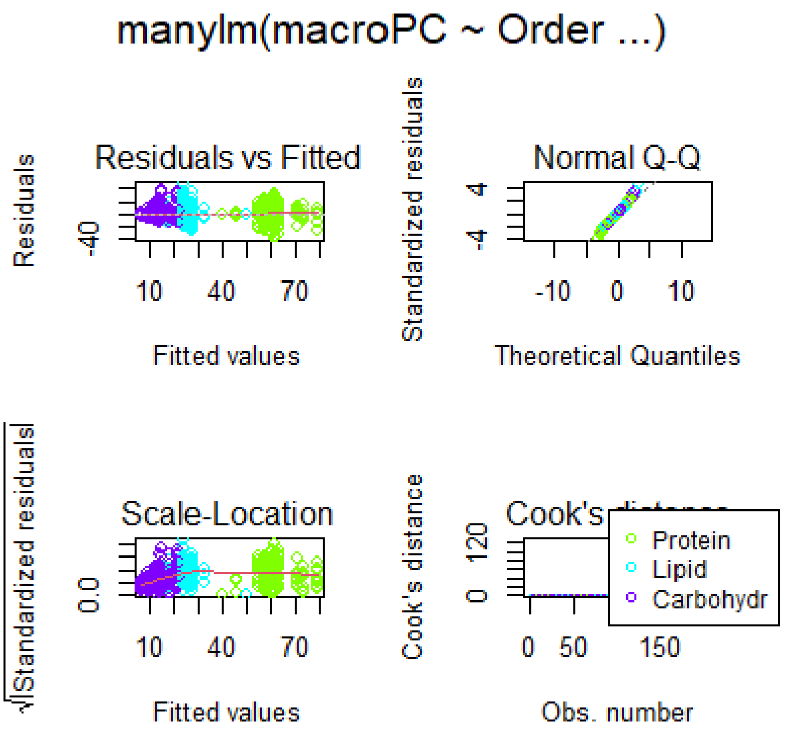
We need to create an 'mvabund' 'manylm' model as before, but this time for family, order and class levels.

```
macroPC <- mvabund((macro[,6:8]))

modf<-manylm(macroPC~Family, data=macro)
plot(modf)
```



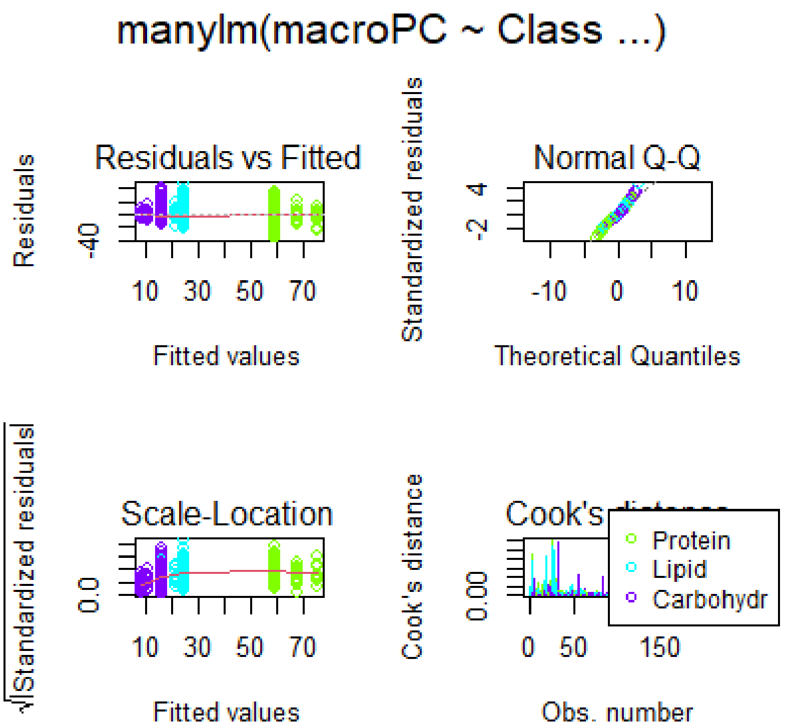
```
anova(modf, p.uni="adjusted")

## Analysis of Variance Table
##
## Model: manylm(formula = macroPC ~ Family, data = macro)
##
## Overall test for all response variables
## Test statistics:
##             Res.Df Df.diff val(F) Pr(>F)
## (Intercept)    200
## Family         137      63  8.673  0.002 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Univariate Tests
## Test statistics:
##             Carbohydrate          Lipid        Protein
##                   F value Pr(>F) F value Pr(>F) F value Pr(>F)
## (Intercept)
## Family               2.456  0.003    3.22  0.002   2.997  0.002
##
## Arguments: with 999 resampling iterations using residual (without replacem
ent) resampling and response assumed to be uncorrelated

modo<-manylm(macroPC~Order, data=macro)
plot(modo)
```



```
anova(modo, p.uni="adjusted")

## Analysis of Variance Table
##
## Model: manylm(formula = macroPC ~ Order, data = macro)
##
## Overall test for all response variables
## Test statistics:
##             Res.Df Df.diff val(F) Pr(>F)
## (Intercept)    200
## Order          190      10   8.47  0.002 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Univariate Tests
## Test statistics:
##                Carbohydrate          Lipid         Protein
##                     F value Pr(>F) F value Pr(>F) F value Pr(>F)
## (Intercept)
## Order                 2.814  0.013   2.374  0.016   3.282  0.008
##
## Arguments: with 999 resampling iterations using residual (without replacem
ent) resampling and response assumed to be uncorrelated

modc<-manylm(macroPC~Class, data=macro)
plot(modc)
```

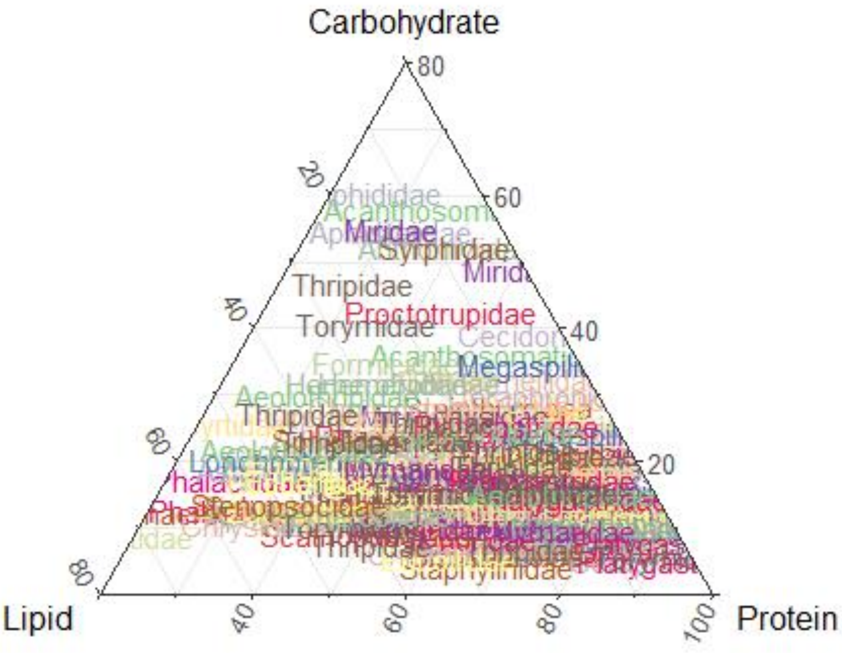


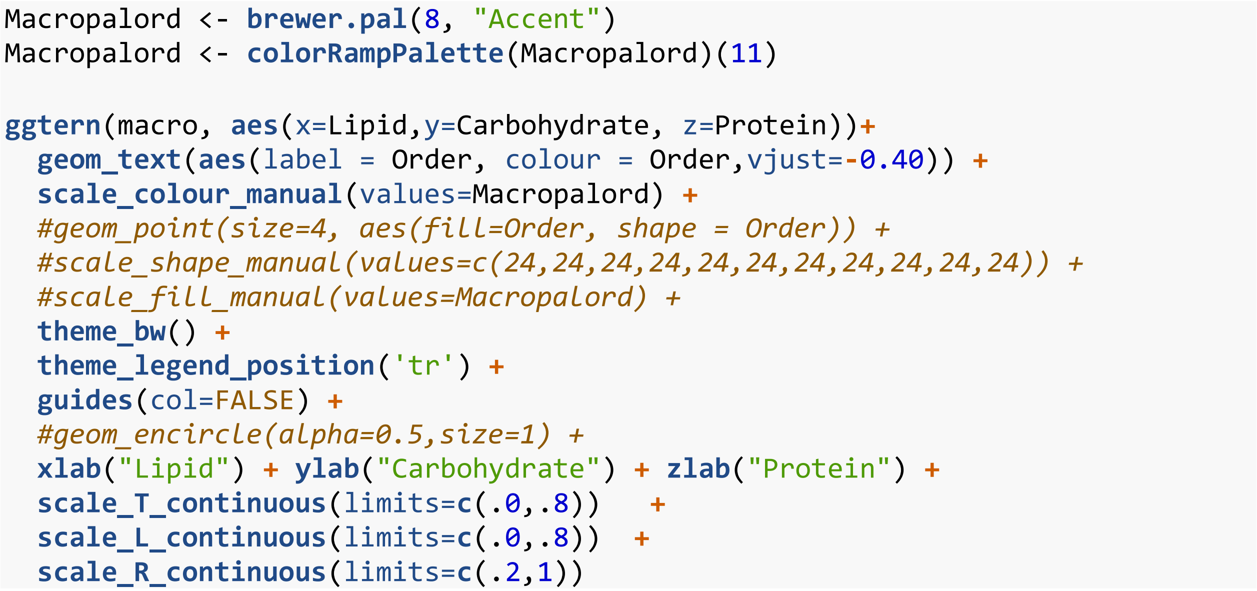
```
anova(modc, p.uni="adjusted")

## Analysis of Variance Table
##
## Model: manylm(formula = macroPC ~ Class, data = macro)
##
## Overall test for all response variables
## Test statistics:
##             Res.Df Df.diff val(F) Pr(>F)
## (Intercept)    200
## Class          198       2  18.84  0.002 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Univariate Tests
## Test statistics:
##             Carbohydrate          Lipid        Protein        
##                  F value Pr(>F) F value Pr(>F) F value Pr(>F)
## (Intercept)                                                  
## Class              5.456  0.017   3.292  0.034  10.092  0.002
## 
## Arguments: with 999 resampling iterations using residual (without replacem
ent) resampling and response assumed to be uncorrelated
```

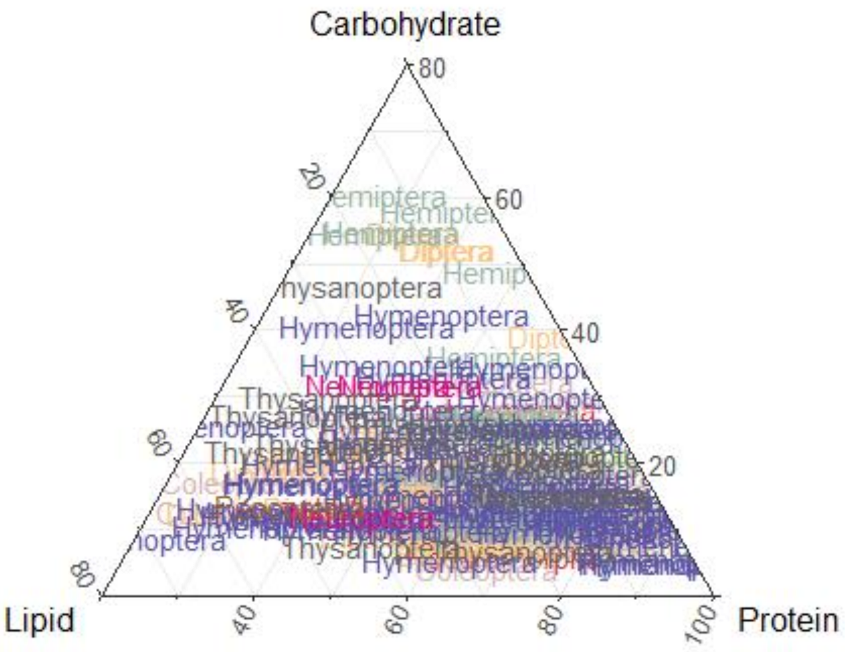
We can then plot this a ternary plots.

```
Macropalfam <- brewer.pal(8, "Accent")
Macropalfam <- colorRampPalette(Macropalfam)(64)

ggtern(macro, aes(x=Lipid,y=Carbohydrate, z=Protein))+
  geom_text(aes(label = Family, colour = Family,vjust=-0.40)) +
  scale_colour_manual(values=Macropalfam) +
  #geom_point(size=4, aes(fill=Order, shape = Order)) +
  #scale_shape_manual(values=c(24,24,24,24,24,24,24,24,24,24,24)) +
  #scale_fill_manual(values=Macropalord) +
  theme_bw() +
  theme_legend_position('tr') +
  guides(col=FALSE) +
  #geom_encircle(alpha=0.5,size=1) +
  xlab("Lipid") + ylab("Carbohydrate") + zlab("Protein") +
  scale_T_continuous(limits=c(.0,.8))   +
  scale_L_continuous(limits=c(.0,.8))  +
  scale_R_continuous(limits=c(.2,1))
```

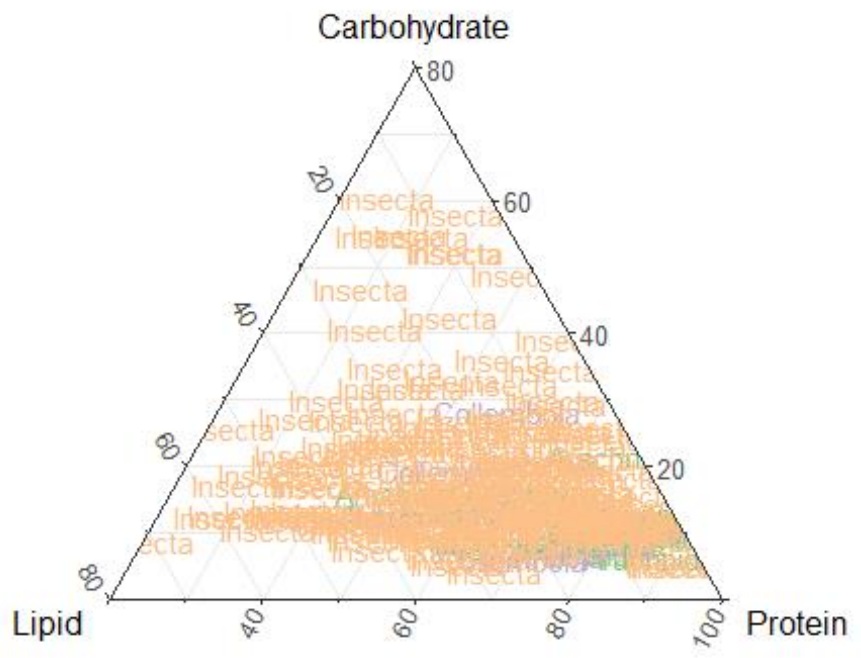




```
Macropalord <- brewer.pal(8, "Accent")
Macropalord <- colorRampPalette(Macropalord)(11)

ggtern(macro, aes(x=Lipid,y=Carbohydrate, z=Protein))+
  geom_text(aes(label = Order, colour = Order,vjust=-0.40)) +
  scale_colour_manual(values=Macropalord) +
  #geom_point(size=4, aes(fill=Order, shape = Order)) +
  #scale_shape_manual(values=c(24,24,24,24,24,24,24,24,24,24,24)) +
  #scale_fill_manual(values=Macropalord) +
  theme_bw() +
  theme_legend_position('tr') +
  guides(col=FALSE) +
  #geom_encircle(alpha=0.5,size=1) +
  xlab("Lipid") + ylab("Carbohydrate") + zlab("Protein") +
  scale_T_continuous(limits=c(.0,.8))   +
  scale_L_continuous(limits=c(.0,.8))  +
  scale_R_continuous(limits=c(.2,1))
```



```
Macropalcla <- brewer.pal(3, "Accent")

ggtern(macro, aes(x=Lipid,y=Carbohydrate, z=Protein))+
  geom_text(aes(label = Class, colour = Class,vjust=-0.40)) +
  scale_colour_manual(values=Macropalcla) +
  #geom_point(size=4, aes(fill=Order, shape = Order)) +
  #scale_shape_manual(values=c(24,24,24,24,24,24,24,24,24,24,24)) +
  #scale_fill_manual(values=Macropalord) +
  theme_bw() +
  theme_legend_position('tr') +
  guides(col=FALSE) +
  #geom_encircle(alpha=0.5,size=1) +
  xlab("Lipid") + ylab("Carbohydrate") + zlab("Protein") +
  scale_T_continuous(limits=c(.0,.8))   +
  scale_L_continuous(limits=c(.0,.8))  +
  scale_R_continuous(limits=c(.2,1))
```



Tropho-species

We can now cluster the taxa above based on their macronutrient content.

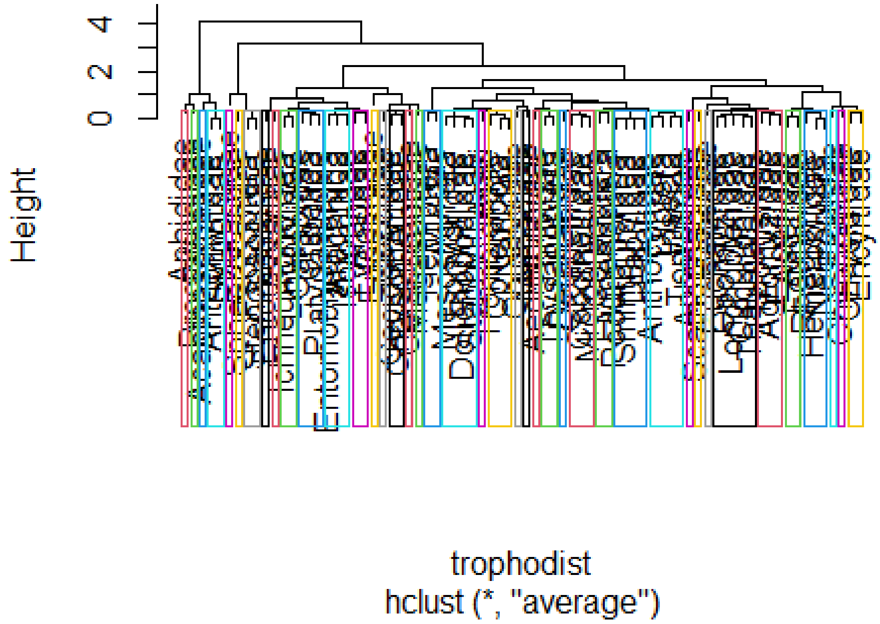
Clustering method determination

We must first prepare a scaled dissimilarity matrix.

```
tropho <- read.csv("TaxAvgMacros2.csv")
rownames(tropho) <- tropho[,1]
tropho_taxon <- tropho$Taxon
tropho <- tropho[2:4]
summary(tropho)

##   Carbohydrate        Lipid           Protein
##  Min.   : 5.169   Min.   : 4.222   Min.   :30.92
##  1st Qu.:10.367   1st Qu.:16.404   1st Qu.:50.82
##  Median :12.858   Median :24.428   Median :59.68
##  Mean   :15.561   Mean   :24.790   Mean   :59.65
##  3rd Qu.:18.020   3rd Qu.:32.063   3rd Qu.:68.50
##  Max.   :49.142   Max.   :57.260   Max.   :83.08

tropho_sc <- as.data.frame(scale(tropho))
summary(tropho_sc)

##   Carbohydrate         Lipid             Protein
##  Min.   :-1.1315   Min.   :-1.79901   Min.   :-2.284429
##  1st Qu.:-0.5655   1st Qu.:-0.73348   1st Qu.:-0.702347  
##  Median :-0.2943   Median :-0.03164   Median : 0.002436  
##  Mean   : 0.0000   Mean   : 0.00000   Mean   : 0.000000  
##  3rd Qu.: 0.2678   3rd Qu.: 0.63618   3rd Qu.: 0.703560  
##  Max.   : 3.6565   Max.   : 2.84011   Max.   : 1.863155

trophodist<- dist(tropho_sc, method = "euclidean")
```

Next, we can compare the clustering methods available based on their Dunn index. Optimal cluster numbers are defined as the first instance of a Dunn index before the first decrease in Dunn index. First, we will try the 'average' method.

```
trophotreeAVG <- hclust(trophodist, method = "average")
plot(trophotreeAVG, main="")

x <- c(3:43)
for (i in x) {
  trophocut_avg <- cutree(trophotreeAVG, k = i )
  trophodunn <- dunn(distance= trophodist, clusters = trophocut_avg, method=
'euclidean')
  print(trophodunn)
}

## [1] 0.1156344
## [1] 0.08270121
## [1] 0.1100611
## [1] 0.1147324
## [1] 0.129964
## [1] 0.1324432
## [1] 0.1324432
## [1] 0.1324432
## [1] 0.1324432
## [1] 0.1768612
## [1] 0.1768612
## [1] 0.1768612
## [1] 0.1768612
## [1] 0.2065349
## [1] 0.2588326
## [1] 0.2588326
## [1] 0.2719344
## [1] 0.3699144
## [1] 0.3699144
## [1] 0.3785205
## [1] 0.3971739
## [1] 0.3971739
## [1] 0.3971739
## [1] 0.3971739
## [1] 0.3971739
## [1] 0.3971739
## [1] 0.3971739
## [1] 0.4098497
## [1] 0.4393966
## [1] 0.4393966
## [1] 0.5245227
## [1] 0.5245227
## [1] 0.5245227
## [1] 0.5245227
## [1] 0.6890748
## [1] 0.6890748
## [1] 0.6890748
## [1] 0.6890748
## [1] 0.6912715
## [1] 0.6912715
## [1] 0.6357246

plot(trophotreeAVG, main="")
rect.hclust(trophotreeAVG, k = 41, border = 2:28)
```



```
trophocut_avg41 <- cutree(trophotreeAVG, k = 41)
trophodunn_avg41 <- dunn(distance= trophodist, clusters = trophocut_avg41, me
thod= 'euclidean')
trophodunn_avg41

## [1] 0.6912715
```

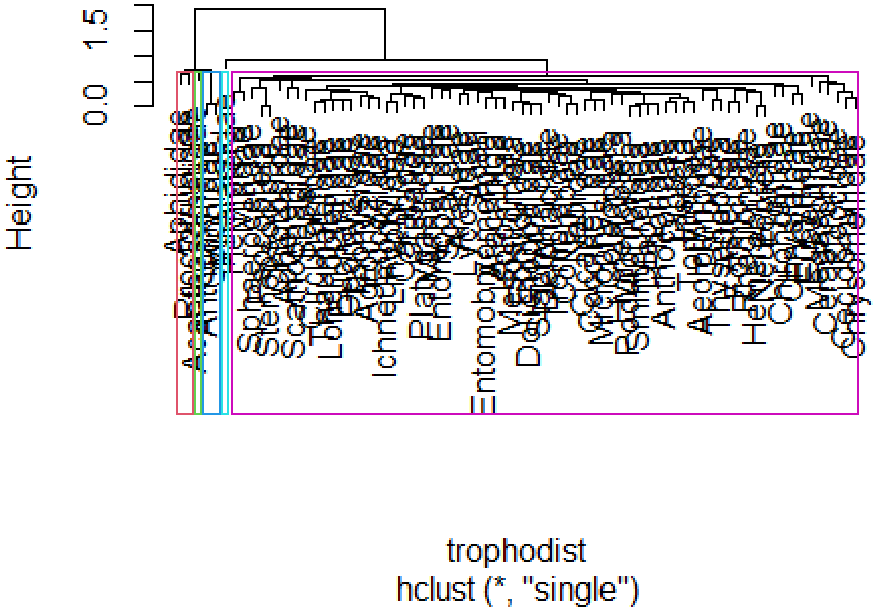
The optimal cluster number was determined as 41.

Next, the 'single' method.

```
trophotreeSIN <- hclust(trophodist, method = "single")
plot(trophotreeSIN, main="")

x <- c(5:50)
for (i in x) {
  trophocut_sin <- cutree(trophotreeSIN, k = i )
  trophodunn <- dunn(distance= trophodist, clusters = trophocut_sin, method=
'euclidean')
  print(trophodunn)
}
```

```
## [1] 0.1384584
## [1] 0.1241447
## [1] 0.1259029
## [1] 0.1222589
## [1] 0.11531
## [1] 0.1185188
## [1] 0.1174049
## [1] 0.114596
## [1] 0.1083651
## [1] 0.1141812
## [1] 0.1124977
## [1] 0.1054711
## [1] 0.1017403
## [1] 0.1017121
## [1] 0.1092853
## [1] 0.1034176
## [1] 0.10309
## [1] 0.1692886
## [1] 0.165458
## [1] 0.1625322
## [1] 0.1584304
## [1] 0.1852456
## [1] 0.1844471
## [1] 0.1830679
## [1] 0.1822089
## [1] 0.2794304
## [1] 0.2701018
## [1] 0.3774001
## [1] 0.3676041
## [1] 0.3611366
## [1] 0.3591513
## [1] 0.3478249
## [1] 0.4375518
## [1] 0.4239411
## [1] 0.4102352
## [1] 0.4098497
## [1] 0.4393966
## [1] 0.5245227
## [1] 0.6912715
## [1] 0.6859947
## [1] 0.6660239
## [1] 0.663041
## [1] 0.6357246
## [1] 0.6207389
## [1] 0.5549374
## [1] 0.5154569
```

```
rect.hclust(trophotreeSIN, k = 5, border = 2:28)
```

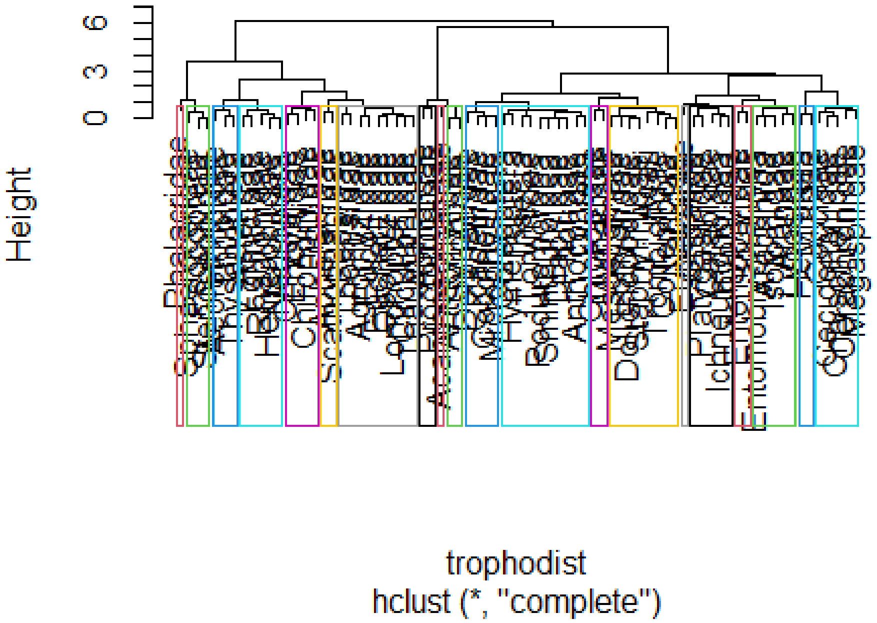


```
trophocut_sin5 <- cutree(trophotreeSIN, k = 5)
trophodunn_sin5 <- dunn(distance= trophodist, clusters = trophocut_sin5, meth
od= 'euclidean')
trophodunn_sin5

## [1] 0.1384584
```

The optimal cluster number is unclear, technically appearing to be 5 (which is fewer than would be ideal for this analysis) and just about every number following it.

Next, the 'complete' method.

```
trophotreeCOM <- hclust(trophodist, method = "complete")
plot(trophotreeCOM, main="")

x <- c(5:21)
for (i in x) {
  trophocut_com <- cutree(trophotreeCOM, k = i )
  trophodunn <- dunn(distance= trophodist, clusters = trophocut_com, method=
'euclidean')
  print(trophodunn)
}

## [1] 0.09967108
## [1] 0.1100611
## [1] 0.1571924
## [1] 0.1607845
## [1] 0.1794517
## [1] 0.181262
## [1] 0.2065349
## [1] 0.2183413
## [1] 0.2185233
## [1] 0.220858
## [1] 0.2394086
## [1] 0.240049
## [1] 0.2719344
## [1] 0.2932035
## [1] 0.3615931
## [1] 0.3699144
## [1] 0.3071432

rect.hclust(trophotreeCOM, k = 20, border = 2:28)
```



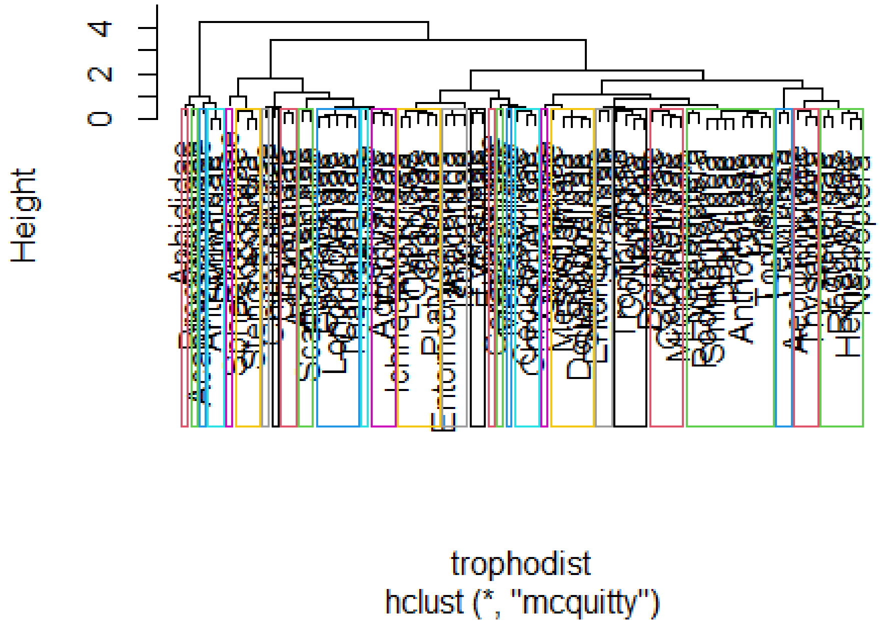
```
trophocut_com20 <- cutree(trophotreeCOM, k = 20)
trophodunn_com20 <- dunn(distance= trophodist, clusters = trophocut_com20, me
thod= 'euclidean')
trophodunn_com20

## [1] 0.3699144
```

The optimal cluster number seems to be 20.

Next the 'mcquitty' method.

```
trophotreeMCQ <- hclust(trophodist, "mcquitty")
plot(trophotreeMCQ, main="")

x <- c(5:50)
for (i in x) {
  trophocut_mcq <- cutree(trophotreeMCQ, k = i )
  trophodunn <- dunn(distance= trophodist, clusters = trophocut_mcq, method=
'euclidean')
  print(trophodunn)
}

## [1] 0.09094023
## [1] 0.129964
## [1] 0.129964
## [1] 0.1483101
## [1] 0.1483101
## [1] 0.1545601
## [1] 0.1545601
## [1] 0.1545601
## [1] 0.2048922
## [1] 0.2048922
## [1] 0.2183413
## [1] 0.2588326
## [1] 0.2588326
## [1] 0.2719344
## [1] 0.2719344
## [1] 0.32448
## [1] 0.32448
## [1] 0.32448
## [1] 0.32448
## [1] 0.32448
## [1] 0.32448
## [1] 0.3954614
## [1] 0.3954614
## [1] 0.3954614
## [1] 0.4098497
## [1] 0.4098497
## [1] 0.4098497
## [1] 0.3890009
## [1] 0.5245227
## [1] 0.5245227
## [1] 0.5245227
## [1] 0.5245227
## [1] 0.6890748
## [1] 0.6890748
## [1] 0.6890748
## [1] 0.6890748
## [1] 0.6912715
## [1] 0.6912715
## [1] 0.6357246
## [1] 0.6357246
## [1] 0.6207389
## [1] 0.5092077
## [1] 0.5092077
## [1] 0.5092077
## [1] 0.5092077
## [1] 0.5092077
```

```
rect.hclust(trophotreeMCQ, k = 29, border = 2:28)
```



```
trophocut_mcq29 <- cutree(trophotreeMCQ, k = 29)
trophodunn_mcq29 <- dunn(distance= trophodist, clusters = trophocut_mcq29, me
thod= 'euclidean')
trophodunn_mcq29

## [1] 0.4098497
```

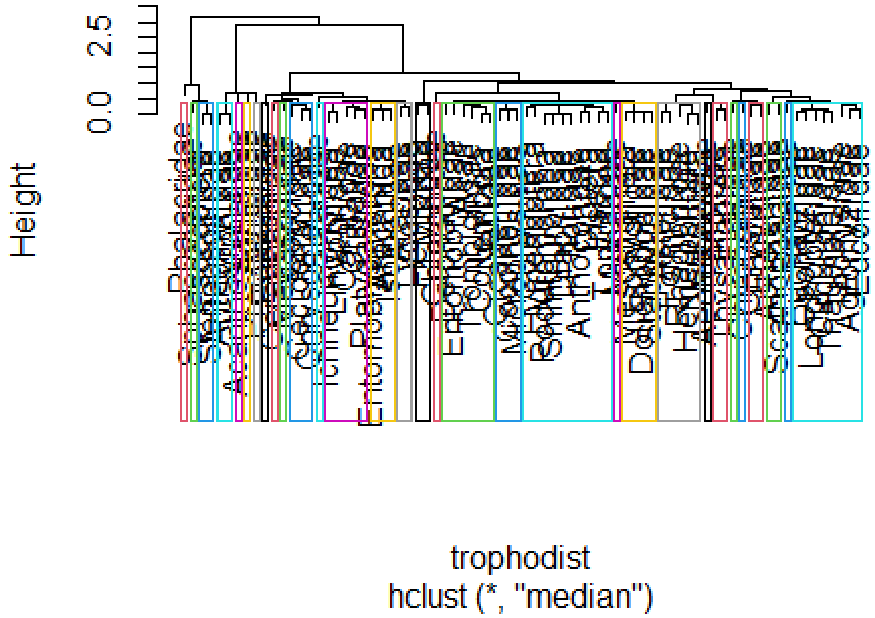
The optimal cluster number seems to be 29.

Next, the 'median' method.

```r
trophotreeMED <- hclust(trophodist, "median")
plot(trophotreeMED, main="")

x <- c(5:40)
for (i in x) {
  trophocut_med <- cutree(trophotreeMED, k = i )
  trophodunn <- dunn(distance= trophodist, clusters = trophocut_med, method=
'euclidean')
  print(trophodunn)
}
```

```
## [1] 0.08207764
## [1] 0.1100611
## [1] 0.1100611
## [1] 0.129964
## [1] 0.129964
## [1] 0.129964
## [1] 0.129964
## [1] 0.129964
## [1] 0.1720642
## [1] 0.1720642
## [1] 0.191713
## [1] 0.191713
## [1] 0.191713
## [1] 0.191713
## [1] 0.191713
## [1] 0.191713
## [1] 0.191713
## [1] 0.191713
## [1] 0.191713
## [1] 0.191713
## [1] 0.191713
## [1] 0.191713
## [1] 0.3242512
## [1] 0.3242512
## [1] 0.3242512
## [1] 0.3242512
## [1] 0.4102352
## [1] 0.4102352
## [1] 0.4102352
## [1] 0.4102352
## [1] 0.4098497
## [1] 0.4098497
## [1] 0.4393966
## [1] 0.4393966
## [1] 0.4393966
## [1] 0.4393966
```

```r
rect.hclust(trophotreeMED, k = 31, border = 2:28)
```

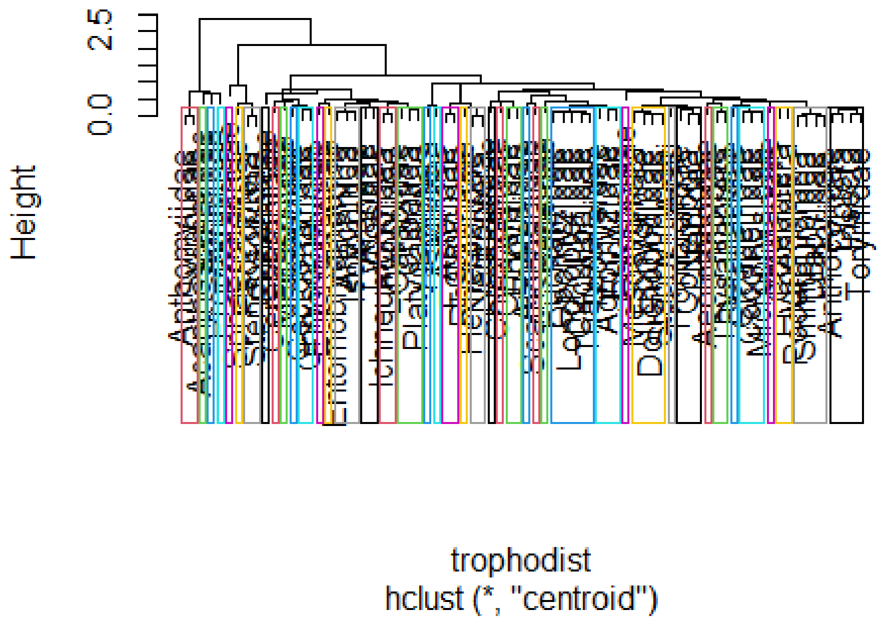


```
trophocut_med31 <- cutree(trophotreeMED, k = 31)
trophodunn_med31 <- dunn(distance= trophodist, clusters = trophocut_med31, me
thod= 'euclidean')
trophodunn_med31

## [1] 0.4102352
```

The optimal cluster number seems to be 31.

Finally, the 'centroid' method.

```
trophotreeCEN <- hclust(trophodist, "centroid")
plot(trophotreeCEN, main="")

x <- c(5:50)
for (i in x) {
  trophocut_cen <- cutree(trophotreeCEN, k = i )
  trophodunn <- dunn(distance= trophodist, clusters = trophocut_cen, method=
'euclidean')
  print(trophodunn)
}

## [1] 0.08270121
## [1] 0.08270121
## [1] 0.08270121
## [1] 0.08270121
## [1] 0.08270121
## [1] 0.1324432
## [1] 0.1324432
## [1] 0.1324432
## [1] 0.1324432
## [1] 0.1324432
## [1] 0.1324432
## [1] 0.1324432
## [1] 0.1768612
## [1] 0.1976142
## [1] 0.2588326
## [1] 0.2588326
## [1] 0.2588326
## [1] 0.2588326
## [1] 0.2719344
## [1] 0.3488709
## [1] 0.3488709
## [1] 0.3488709
## [1] 0.3971739
## [1] 0.3971739
## [1] 0.3971739
## [1] 0.3832626
## [1] 0.3832626
## [1] 0.3832626
## [1] 0.3832626
## [1] 0.3832626
## [1] 0.4127715
## [1] 0.4393966
## [1] 0.4393966
## [1] 0.4393966
## [1] 0.4393966
## [1] 0.4393966
## [1] 0.4393966
## [1] 0.5245227
## [1] 0.6912715
## [1] 0.6859947
## [1] 0.6357246
## [1] 0.6357246
## [1] 0.6357246
## [1] 0.6207389
## [1] 0.5092077
## [1] 0.5092077

rect.hclust(trophotreeCEN, k = 43, border = 2:28)
```



```
trophocut_cen43 <- cutree(trophotreeAVG, k = 43)
trophodunn_cen43 <- dunn(distance= trophodist, clusters = trophocut_cen43, me
thod= 'euclidean')
trophodunn_cen43

## [1] 0.6357246
```

The optimal cluster number appears to be 43.

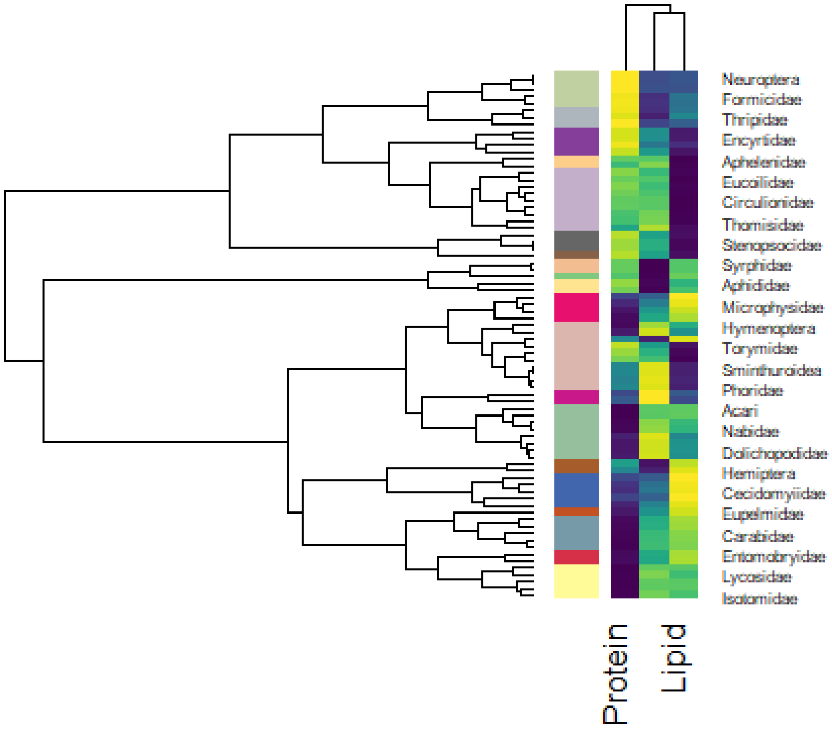
Tropho-species clustering

The 'complete' method produced the lowest but still useful number of clusters (20) and will thus be taken forward. Now we can extract the cluster data.

```
tropho_cl <- mutate(tropho, cluster = trophocut_com20)
count(tropho_cl,cluster)

##    cluster  n
## 1        1  1
## 2        2  8
## 3        3  3
## 4        4  9
## 5        5 10
## 6        6  2
## 7        7  2
## 8        8  2
## 9        9  5
## 10       10  5
## 11       11  5
## 12       12  5
## 13       13  4
## 14       14  2
## 15       15  4
## 16       16  2
## 17       17  1
## 18       18  2
## 19       19  1
## 20       20  3

TrophoClusters <- table(tropho_cl$cluster,tropho_taxon)
```

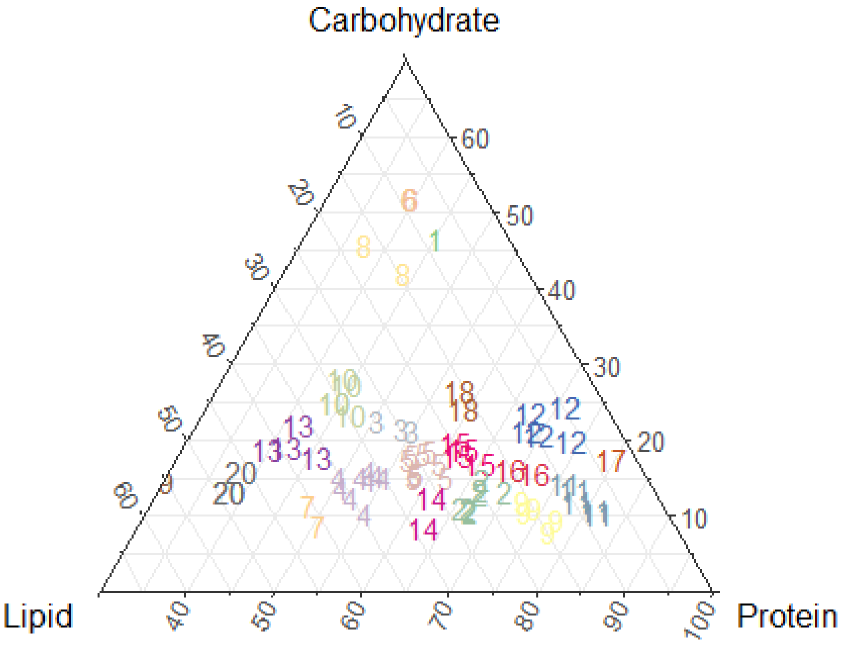
Next, we can create a paired heatmap and dendrogram to show the difference in macronutrient content between clusters.

```
hclustfunc <- function(x) hclust(x, method="complete")
distfunc <- function(x) dist(x,method="euclidean")
d <- distfunc(tropho_sc)
fit <- hclustfunc(d)
clusters <- cutree(fit, h=0.74)
nofclust.height <-  length(unique(as.vector(clusters)));

cl.row <- hclustfunc(distfunc(tropho_sc))
cl.col <- hclustfunc(distfunc(t(tropho_sc)))

hmcols <- rev(viridis(2750))
selcol <- colorRampPalette(brewer.pal(12,"Set3"))
selcol2 <- colorRampPalette(brewer.pal(8,"Accent"))
clustcol.height = selcol2(nofclust.height);

heatmap.2(as.matrix(tropho_sc),
          trace='none',
          dendrogram='both',
          key=F,
          Colv=T,
          scale='row',
          hclust=hclustfunc, distfun=distfunc, col=hmcols,
          symbreak=T,
          margins=c(7,10), keysize=0.1,
          lwid=c(5,0.5,3), lhei=c(0.05,0.5),
          lmat=rbind(c(5,0,4),c(3,1,2)),
          labRow=rownames(tropho_sc),
          RowSideColors=clustcol.height[clusters], cexRow = 0.8, cexCol = 1.5
)
```



We can also show this using a ternary plot.

```r
Clustpal <- brewer.pal(8, "Accent")
Clustpal <- colorRampPalette(Clustpal)(20)

ggtern(tropho_cl, aes(x=Lipid,y=Carbohydrate, z=Protein))+
  geom_text(aes(label = as.factor(cluster), colour = as.factor(cluster)),
  vjust=-0.40) +
  scale_colour_manual(values=Clustpal) +
  #geom_point(size=4, shape=24, aes(fill=(as.factor(cluster)))) +
  #scale_fill_manual(values=Clustpal, name = "Cluster") +
  theme_bw() +
  #theme_legend_position('tr') +
  guides(col=FALSE) +
  #geom_encircle(alpha=0.5,size=1) +
  xlab("Lipid") + ylab("Carbohydrate") + zlab("Protein") +
  scale_T_continuous(limits=c(.0,.7))   +
  scale_L_continuous(limits=c(.0,.7))  +
  scale_R_continuous(limits=c(.3,1))
```

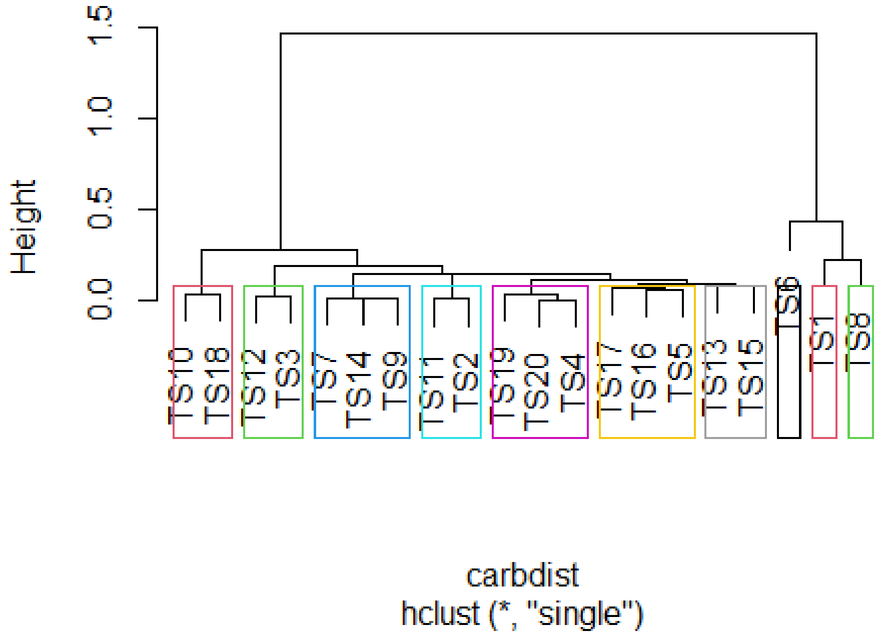


Tropho-species individual macronutrient clustering

To informatively name tropho-species, we will cluster them based on their mean macronutrient contents for each macronutrient individually. We must, however, again choose an optimal clustering method. Using the same method above, the 'single' method was found to be optimal.

Each macronutrient must then be clustered, and clusters extracted, individually.

```
TSn <- read.csv("TrophoMacroCluster.csv")
rownames(TSn) <- TSn[,1]
TSn_cluster <- TSn$Cluster

# Carbohydrate clustering

TSncarb <- TSn[2]
TSncarb_sc <- as.data.frame(scale(TSncarb))
carbdist<- dist(TSncarb_sc, method = "euclidean")

carbtreeSIN <- hclust(carbdist, method = "single")
plot(carbtreeSIN, main="")

x <- c(5:15)
for (i in x) {
  carbcut_sin <- cutree(carbtreeSIN, k = i )
  carbdunn <- dunn(distance= carbdist, clusters = carbcut_sin, method= 'eucli
dean')
  print(carbdunn)
}

## [1] 0.2319649
## [1] 0.2463051
## [1] 0.2566469
## [1] 0.3390231
## [1] 0.3764554
## [1] 0.710202
## [1] 0.6715166
## [1] 1.203863
## [1] 1.283473
## [1] 1.196052
## [1] 1.404537

rect.hclust(carbtreeSIN, k = 10, border = 2:28)
```



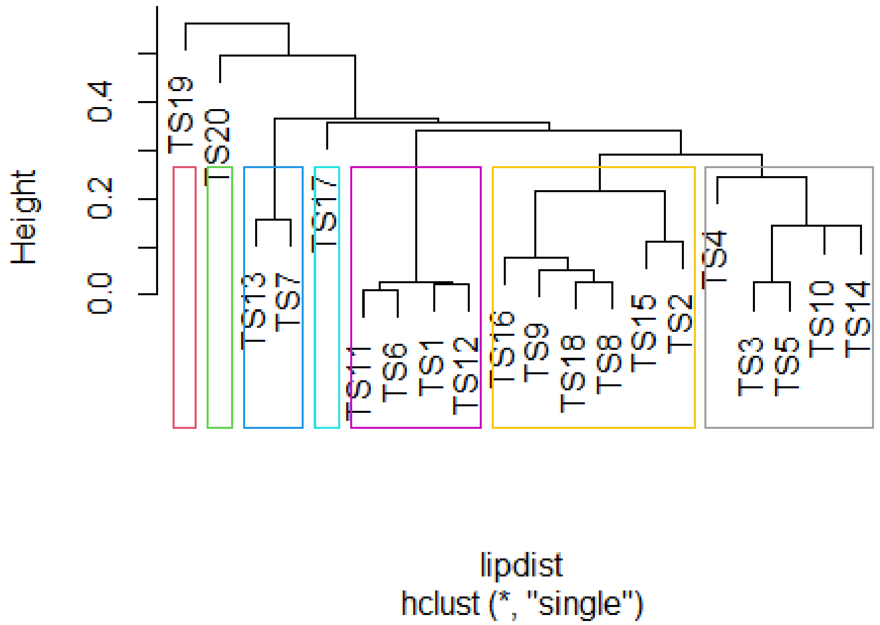
```
carbcut_sin10 <- cutree(carbtreeSIN, k = 10)
carbdunn_sin10 <- dunn(distance= carbdist, clusters = carbcut_sin10, method=
'euclidean')
carbdunn_sin10

## [1] 0.710202

carb_cl <- mutate(TSncarb, cluster = carbcut_sin10)
count(carb_cl,cluster)
```

```
##    cluster n
## 1        1 1
## 2        2 2
## 3        3 2
## 4        4 2
## 5        5 2
## 6        6 3
## 7        7 3
## 8        8 3
## 9        9 1
## 10      10 1
```

```
CarbClusters <- table(carb_cl$cluster,TSn_cluster)

# Lipid clustering

TSnlip <- TSn[3]
TSnlip_sc <- as.data.frame(scale(TSnlip))
lipdist<- dist(TSnlip_sc, method = "euclidean")

liptreeSIN <- hclust(lipdist, method = "single")
plot(liptreeSIN, main="")

x <- c(5:12)
for (i in x) {
  lipcut_sin <- cutree(liptreeSIN, k = i )
  lipdunn <- dunn(distance= lipdist, clusters = lipcut_sin, method= 'euclidea
n')
  print(lipdunn)
}
```

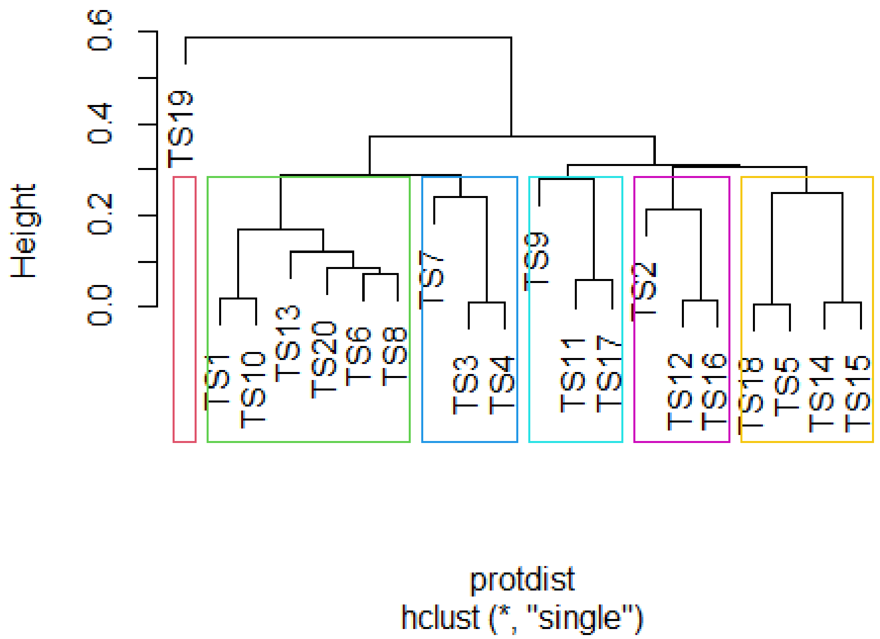
```
## [1] 0.2060919
## [1] 0.2557827
## [1] 0.5176627
## [1] 0.5061237
## [1] 0.6947065
## [1] 0.5062146
## [1] 0.9194819
## [1] 0.9096489
```

```
rect.hclust(liptreeSIN, k = 7, border = 2:28)
```



```
lipcut_sin7 <- cutree(liptreeSIN, k = 7)
lipdunn_sin7 <- dunn(distance= lipdist, clusters = lipcut_sin7, method= 'eucl
idean')
lipdunn_sin7

## [1] 0.5176627

lip_cl <- mutate(TSnlip, cluster = lipcut_sin7)
count(lip_cl,cluster)

##   cluster n
## 1       1 4
## 2       2 5
## 3       3 2
## 4       4 6
## 5       5 1
## 6       6 1
## 7       7 1

LipClusters <- table(lip_cl$cluster,TSn_cluster)


# Protein clustering

TSnprot <- TSn[4]
TSnprot_sc <- as.data.frame(scale(TSnprot))
protdist<- dist(TSnprot_sc, method = "euclidean")
```

```
prottreeSIN <- hclust(protdist, method = "single")
plot(prottreeSIN, main="")

x <- c(5:12)
for (i in x) {
  protcut_sin <- cutree(prottreeSIN, k = i )
  protdunn <- dunn(distance= protdist, clusters = protcut_sin, method= 'eucli
dean')
  print(protdunn)
}

## [1] 0.303253
## [1] 0.6116786
## [1] 0.592195
## [1] 0.5261852
## [1] 0.5085116
## [1] 0.4563415
## [1] 0.6032857
## [1] 0.7664841

rect.hclust(prottreeSIN, k = 6, border = 2:28)
```



```
protcut_sin6 <- cutree(prottreeSIN, k = 6)
protdunn_sin6 <- dunn(distance= protdist, clusters = protcut_sin6, method= 'e
uclidean')
protdunn_sin6
```

```
## [1] 0.6116786
```

```
prot_cl <- mutate(TSnprot, cluster = protcut_sin6)
count(prot_cl,cluster)
```

```
##   cluster n
## 1       1 6
## 2       2 3
## 3       3 3
## 4       4 4
## 5       5 1
## 6       6 3
```

```
ProtClusters <- table(prot_cl$cluster,TSn_cluster)
```

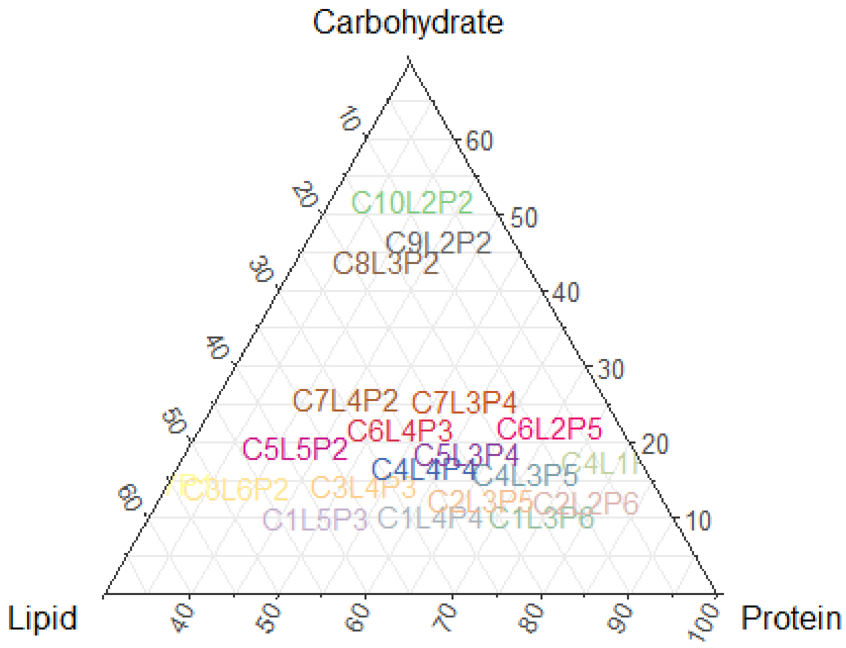
Tropho-species comparison

To exemplify the macronutrient content differences between tropho-species, ternary plots can be used.

```
TS <- read.csv("TSAvgMacros.csv")
rownames(TS) <- TS[,1]
TSmacros <- TS[2:4]
summary(TS)
```

```
##       TS             Carbohydrate        Lipid           Protein     
##  Length:20          Min.   : 7.314   Min.   : 4.222   Min.   :30.92  
##  Class :character   1st Qu.:10.852   1st Qu.:13.918   1st Qu.:45.58  
##  Mode  :character   Median :14.189   Median :21.353   Median :57.14  
##                     Mean   :18.356   Mean   :24.359   Mean   :57.28  
##                     3rd Qu.:19.996   3rd Qu.:31.727   3rd Qu.:69.12  
##                     Max.   :49.121   Max.   :57.260   Max.   :81.19
```

```
TSpal <- brewer.pal(8, "Accent")
TSpal <- colorRampPalette(TSpal)(20)

ggtern(TS, aes(x=Lipid,y=Carbohydrate, z=Protein))+
  geom_text(aes(label = as.factor(TS), colour = as.factor(TS)), vjust=-0.40)
+
  scale_colour_manual(values=TSpal, name = "Tropho-species") +
  #geom_point(size=4, shape=24, aes(fill=Tropho.species)) +
  #scale_fill_manual(values=TSpal, name = "Tropho-species") +
  theme_bw() +
  theme_legend_position('tr') +
  guides(col=FALSE) +
  #geom_encircle(alpha=0.5,size=1) +
  xlab("Lipid") + ylab("Carbohydrate") + zlab("Protein") +
  scale_T_continuous(limits=c(.0,.7))   +
  scale_L_continuous(limits=c(.0,.7))  +
  scale_R_continuous(limits=c(.3,1))
```



Tropho-species aggregation

For the two downstream analyses for which tropho-species are purposed, they must first be aggregated, both for diet and invertebrate community data.

```
InTSd_to_Agg <- read.csv("TS_Diet_agg.csv")
Aggd <- aggregate(.~TS, data=InTSd_to_Agg, sum)
write.table(Aggd, "TS_Diet_agged.csv")

InTSi_to_Agg <- read.csv("Invert_TS_agg.csv")
Aggi <- aggregate(.~ENNRcode, data=InTSi_to_Agg, sum)
write.table(Aggi, "Invert_TS_agged.csv")
```

Tropho-species co-occurence analysis

Co-occurrence of tropho-species will be analysed in the same manner as in Chapter 4 for taxa, first creating a matrix and then a null model.

```
cooccurts <- read.csv("CooccurenceTSbin.csv")
rownames(cooccurts) <- cooccurts[,1]
coocts <- cooccurts[,-1]

cooctsmat <- create.N.matrix(coocts)

ts.cooccur <- cooccur(coocts, type = "spp_site", spp_names = TRUE, true_rand_
classifier = 0.1, prob = "hyper",  site_mask = NULL, only_effects = FALSE, ef
f_standard = TRUE, eff_matrix = FALSE, thresh=TRUE)
```

```
cooceffts <- effect.sizes(ts.cooccur)
coocprots <- prob.table(ts.cooccur)
```

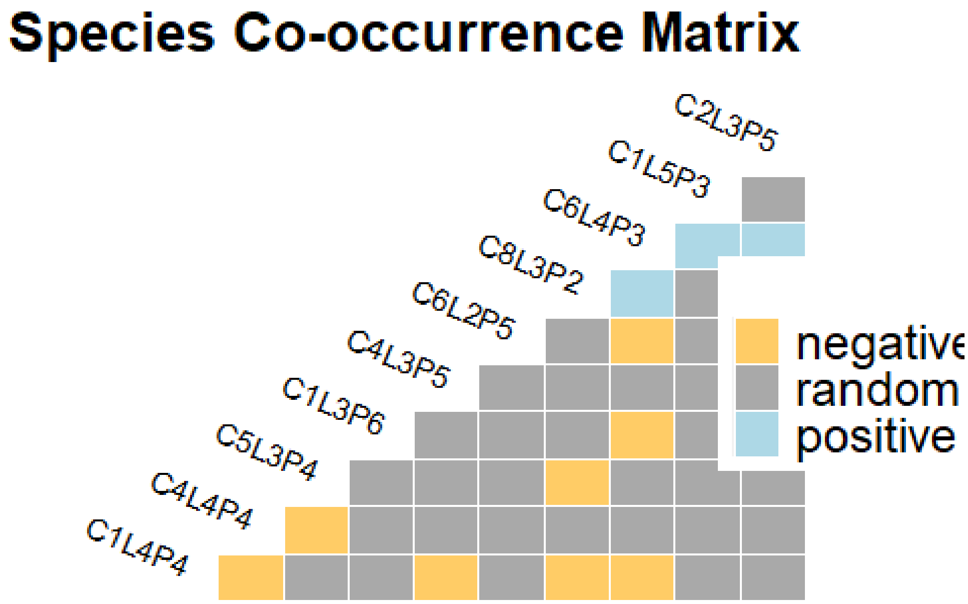
```
## Warning in prob.table(ts.cooccur): The co-occurrence model was run using '
thresh
## = TRUE.' The probability table may not include all species pairs
```

```
cooccurts.results <- print(ts.cooccur)
```

```
## Call:
## cooccur(mat = coocts, type = "spp_site", thresh = TRUE, spp_names = TRUE,
##     true_rand_classifier = 0.1, prob = "hyper", site_mask = NULL,
##     only_effects = FALSE, eff_standard = TRUE, eff_matrix = FALSE)
##
## Of 190 species pair combinations, 122 pairs (64.21 %) were removed from th
e analysis because expected co-occurrence was < 1 and 68 pairs were analyzed
##
## Cooccurrence Table:
##    sp1 sp2 sp1_inc sp2_inc obs_cooccur prob_cooccur exp_cooccur    p_lt
p_gt
## 9    2  16      11      88           1        0.016         4.0 0.04878 0.
99361
## 14   3  11      36      40           1        0.024         5.9 0.00857 0.
99910
## 15   3  12      36     113           9        0.068        16.7 0.00410 0.
99877
## 19   3  16      36      88           5        0.053        13.0 0.00154 0.
99966
## 20   3  19      36      46           2        0.028         6.8 0.01656 0.
99692
## 27   4  16      13      88           9        0.019         4.7 0.99744 0.
01332
## 42   6  16      60      88          32        0.089        21.6 0.99955 0.
00128
## 52  12  13     113      28           6        0.053        13.0 0.00380 0.
99906
## 61  13  19      28      46           1        0.022         5.3 0.01659 0.
99806
## 64  15  16      22      88           2        0.033         7.9 0.00333 0.
99954
## 68  16  19      88      46          24        0.068        16.6 0.99606 0.
01003
##    sp1_name sp2_name
## 9    C1L3P6   C6L4P3
## 14   C1L4P4   C4L3P5
## 15   C1L4P4   C4L4P4
## 19   C1L4P4   C6L4P3
## 20   C1L4P4   C8L3P2
## 27   C1L5P3   C6L4P3
## 42   C2L3P5   C6L4P3
## 52   C4L4P4   C5L3P4
## 61   C5L3P4   C8L3P2
## 64   C6L2P5   C6L4P3
## 68   C6L4P3   C8L3P2
```

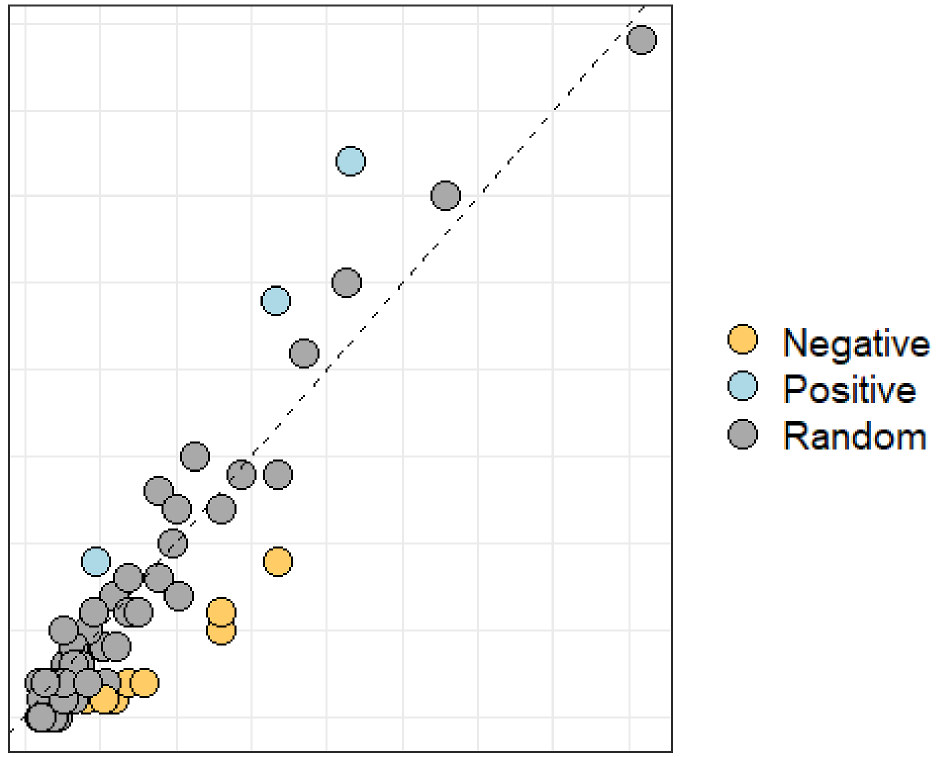
We can then plot these results as a matrix and as expected vs. observed co-occurrences.

```
plot(ts.cooccur)
```



```
df = ts.cooccur$results
df$type = "Random"
df$type[df$p_lt<0.05] = "Negative"
df$type[df$p_gt<0.05] = "Positive"

ove.cots <- ggplot(df,aes(x=exp_cooccur,y=obs_cooccur)) +
  geom_point(aes(fill=type), pch=21, lwd=5) + geom_abline(linetype="dashed")
+
  #geom_label_repel(data=subset(df,sp1_name=="Geospiza magnirostris"),
  # aes(label=paste(sp1_name,sp2_name,sep="\n")),
  # size=2,nudge_x=-1,nudge_y=-1) +
  scale_fill_manual(values=c("#FFCC66","light blue","dark gray")) +
  theme_bw() + theme(axis.text=element_text(size=12), axis.title=element_text
(size=14, face="bold"), legend.title=element_blank(), legend.text=element_tex
t(size=14)) +
  labs(x = "Expected co-occurrence", y = "Observed co-occurrence")
```

```
ove.cots
```



Tropho-species ENNR

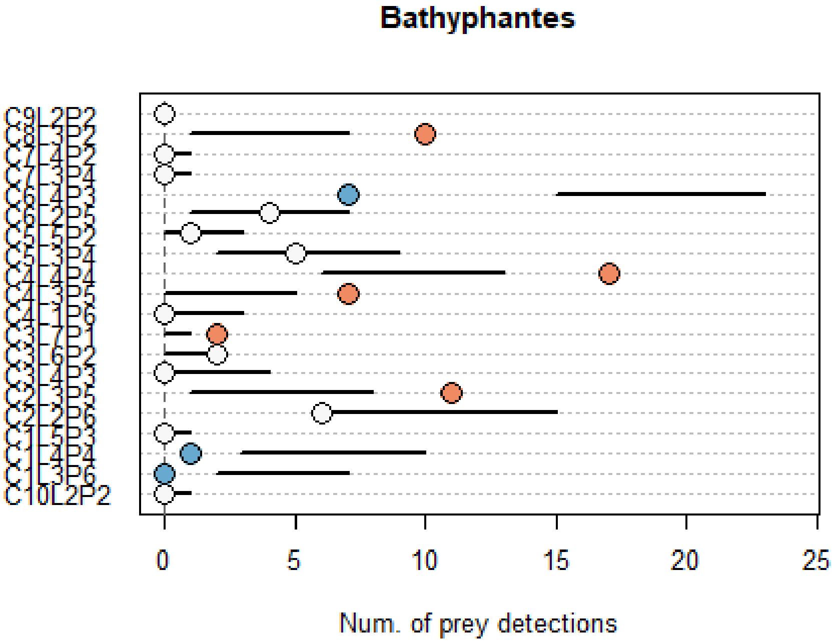
As above for taxa, prey choice will be assssed for spider genera, life stages and sexes using 'econullnetr'.

Tropho-species ENNR for genera

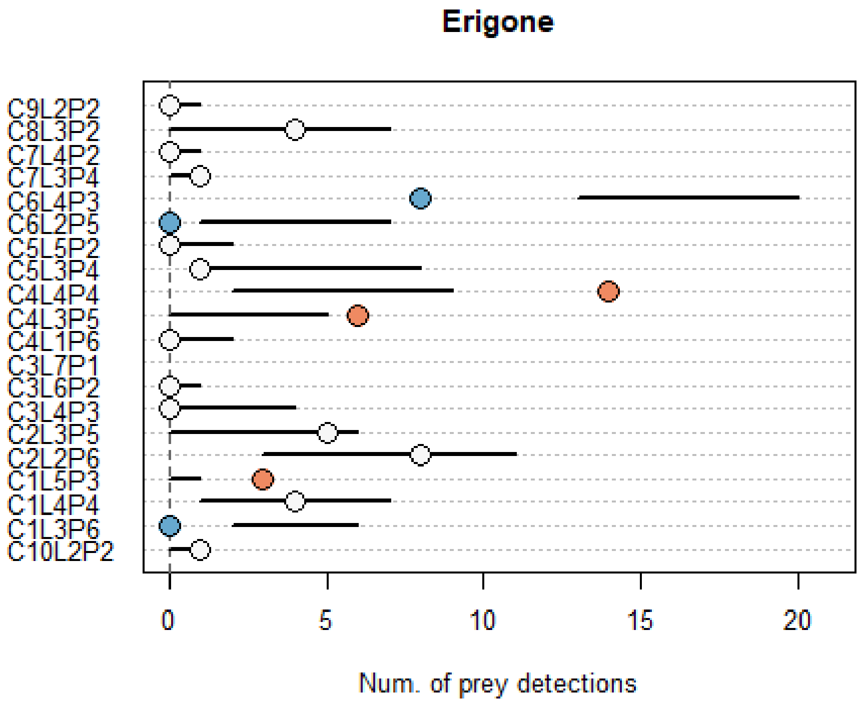
First, we build the model, plot the overall results and extract the data.

```
tsennr <- read.csv("TS_ENNR_Diet_Genusbin.csv")
tsinvertsennr <- read.csv("TS_ENNR_Inverts.csv")
tsENNR.fl <- read.csv("TS_ENNR_Diet.fl_Genus.csv")

genus.null <- generate_null_net(tsennr[,2:22], tsinvertsennr[,2:21],
                                sims = 999, data.type = "names",
                                summary.type = "sum",
                                r.samples = tsinvertsennr[,1],
                                c.samples = tsennr[,1],
                                r.weights = tsENNR.fl)

## Warning in generate_null_net(tsennr[, 2:22], tsinvertsennr[, 2:21], sims =
999, : One or more instances detected where a consumer interacted with a
##        resource that has zero abundance in 'resources'
```

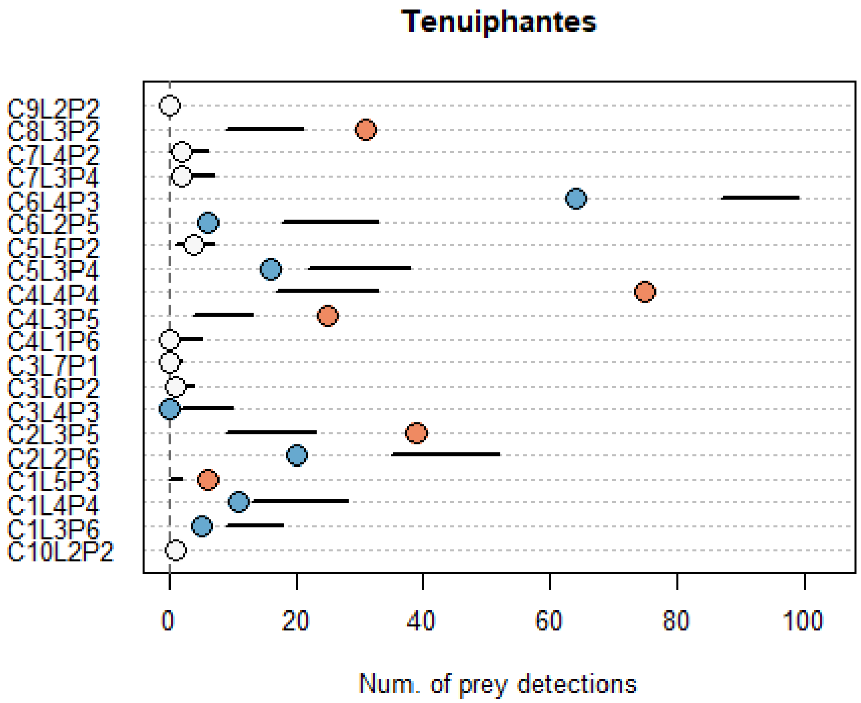
```
#par(mfrow = c(2,3))
par(mfrow = c(1,1))
plot_preferences(genus.null, "Bathyphantes", signif.level = 0.95, type = "counts",
                 xlab = "Num. of prey detections", p.cex = 1.5, l.cex = 0.8,
lwd = 2)

## Warning in test_interactions(nullnet, signif.level = signif.level): Be careful
## of Type I errors due to the large number of tests
```



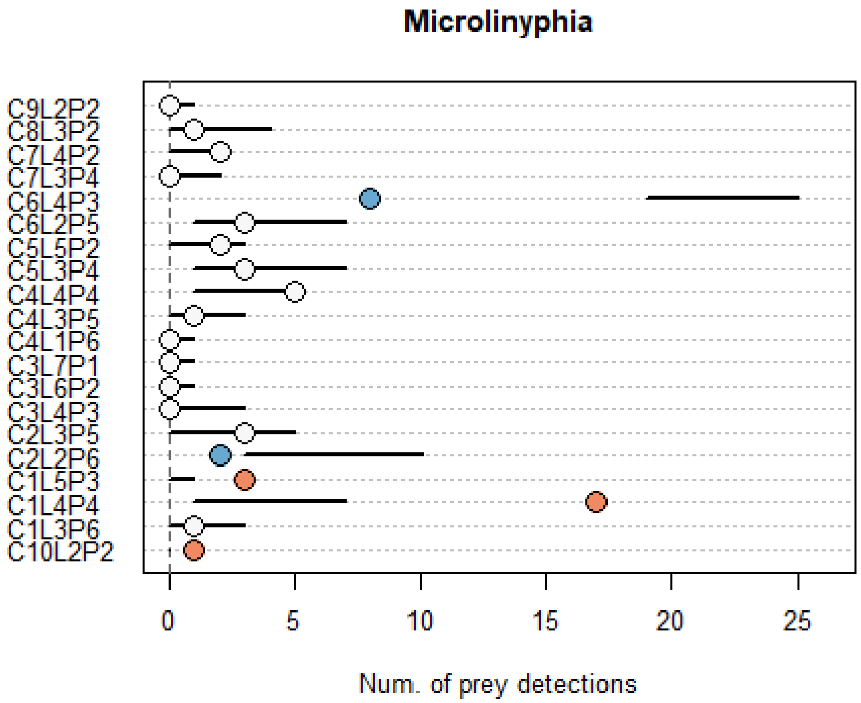
```
plot_preferences(genus.null, "Erigone", signif.level = 0.95, type = "counts",
                 xlab = "Num. of prey detections", p.cex = 1.5, l.cex = 0.8,
lwd = 2)

## Warning in test_interactions(nullnet, signif.level = signif.level): Be careful
## of Type I errors due to the large number of tests
```



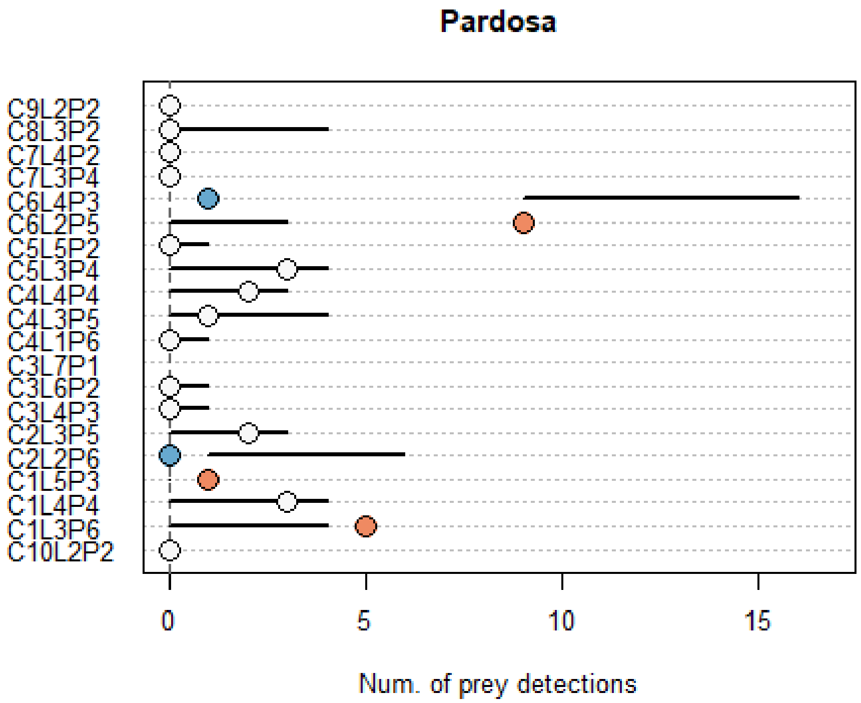
```
plot_preferences(genus.null, "Tenuiphantes", signif.level = 0.95, type = "cou
nts",
                 xlab = "Num. of prey detections", p.cex = 1.5, l.cex = 0.8,
lwd = 2)

## Warning in test_interactions(nullnet, signif.level = signif.level): Be car
eful
## of Type I errors due to the large number of tests
```



```
plot_preferences(genus.null, "Microlinyphia", signif.level = 0.95, type = "co
unts",
                 xlab = "Num. of prey detections", p.cex = 1.5, l.cex = 0.8,
lwd = 2)

## Warning in test_interactions(nullnet, signif.level = signif.level): Be car
eful
## of Type I errors due to the large number of tests
```



```
plot_preferences(genus.null, "Pardosa", signif.level = 0.95, type = "counts",
                 xlab = "Num. of prey detections", p.cex = 1.5, l.cex = 0.8,
lwd = 2)

## Warning in test_interactions(nullnet, signif.level = signif.level): Be car
eful
## of Type I errors due to the large number of tests
```

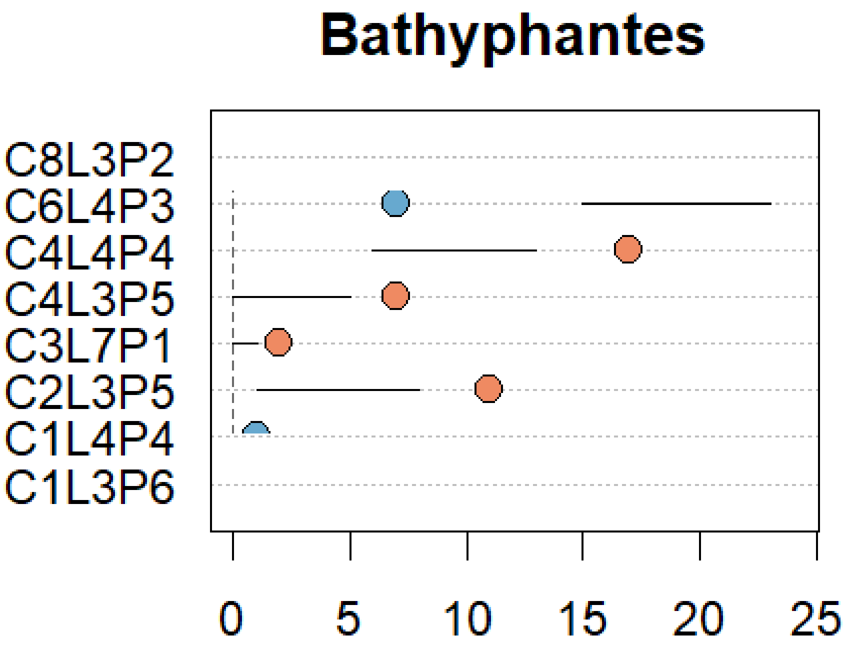


```
gen.links <- test_interactions(genus.null, signif.level = 0.95)

## Warning in test_interactions(genus.null, signif.level = 0.95): Be careful
of
## Type I errors due to the large number of tests
```

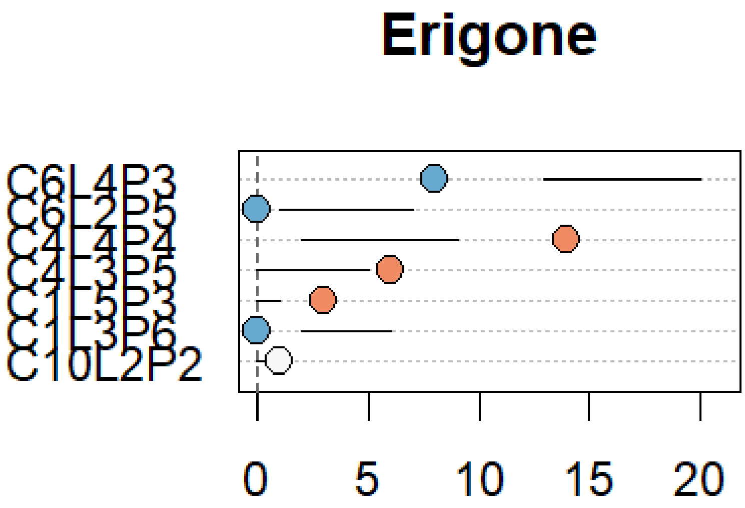
Then we can plot the significant results for each genus

```
# Bathyphates

gbti <- test_interactions(genus.null, signif.level = 0.95)

## Warning in test_interactions(genus.null, signif.level = 0.95): Be careful
of
## Type I errors due to the large number of tests

gbti <- gbti[gbti$Consumer == "Bathyphantes", ]
gbti[, 3] <- ifelse(rowSums(gbti[, 3:6]) == 0, NA, gbti[, 3])
gbti[, 4] <- ifelse(rowSums(gbti[, 3:6]) == 0, NA, gbti[, 4])
gbti[, 5] <- ifelse(rowSums(gbti[, 3:6]) == 0, NA, gbti[, 5])
gbti[, 6] <- ifelse(rowSums(gbti[, 3:6]) == 0, NA, gbti[, 6])

# EDIT 'ti' - to just the prey taxa that you want to show on the plot

gbti <- gbti[c(2,3,6,9,11,12,16,19),]

# Set up maximum x-axis value for xlim. Add an additional 5%
```

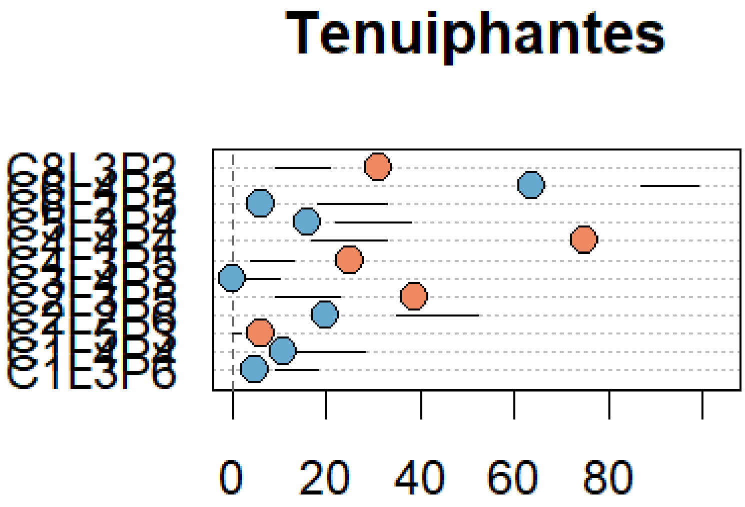
```
gbmin.x <- min(gbti[, 3:6], na.rm = TRUE)
gbmin.x <- max(0, gbmin.x, na.rm = TRUE)
gbmax.x <- max(gbti[, 3:6], na.rm = TRUE)
gbmax.x <- gbmax.x * 1.05
gbti$Setup <- seq(gbmin.x, gbmax.x, length.out = nrow(gbti))

# Plot built up in 2 stages: i) using min and max values to set the
#   y-axis range without having to use ylim (so this can be customised
#   by the user), ii) the main dbarplot and label, and iii) the error
graphics::dotchart(gbti$Setup, labels = paste(gbti$Resource, " ", sep = ""),
                   col = 1, pt.cex = 0, cex = 1.5, main = "Bathyphantes")
graphics::abline(v = 0, lty = 2, col = "dimgrey")


res.col <- c("#67A9CF", "#F7F7F7", "#EF8A62")

for (i in 1:nrow(gbti)){
  eval(parse(text = paste("lines(x = c(gbti$Lower.", 0.95 * 100,
                          ".CL[i], gbti$Upper.", 0.95 * 100,
                          ".CL[i]), y = c(i, i))", sep = "")))
  if(gbti$Test[i] == "Weaker") p.col <- res.col[1]
  if(gbti$Test[i] == "ns" | is.na(gbti$Test[i])) p.col <- res.col[2]
  if(gbti$Test[i] == "Stronger") p.col <- res.col[3]
  graphics::points(gbti$Observed[i], i, pch = 21, col = "black",
                   bg = p.col, cex = 2)
}
```



```r
# Erigone

geti <- test_interactions(genus.null, signif.level = 0.95)

## Warning in test_interactions(genus.null, signif.level = 0.95): Be careful
of
## Type I errors due to the large number of tests

geti <- geti[geti$Consumer == "Erigone", ]
geti[, 3] <- ifelse(rowSums(geti[, 3:6]) == 0, NA, geti[, 3])
geti[, 4] <- ifelse(rowSums(geti[, 3:6]) == 0, NA, geti[, 4])
geti[, 5] <- ifelse(rowSums(geti[, 3:6]) == 0, NA, geti[, 5])
geti[, 6] <- ifelse(rowSums(geti[, 3:6]) == 0, NA, geti[, 6])

# EDIT 'ti' - to just the prey taxa that you want to show on the plot

geti <- geti[c(1,2,4,11,12,15,16),]

# Set up maximum x-axis value for xlim. Add an additional 5%
gemin.x <- min(geti[, 3:6], na.rm = TRUE)
gemin.x <- max(0, gemin.x, na.rm = TRUE)
gemax.x <- max(geti[, 3:6], na.rm = TRUE)
gemax.x <- gemax.x * 1.05
geti$Setup <- seq(gemin.x, gemax.x, length.out = nrow(geti))

# Plot built up in 2 stages: i) using min and max values to set the
#   y-axis range without having to use ylim (so this can be customised
#   by the user), ii) the main dbarplot and label, and iii) the error
graphics::dotchart(geti$Setup, labels = paste(geti$Resource, " ", sep = ""),
                   col = 1, pt.cex = 0, cex = 1.5, main = "Erigone")
graphics::abline(v = 0, lty = 2, col = "dimgrey")


res.col <- c("#67A9CF", "#F7F7F7", "#EF8A62")

for (i in 1:nrow(geti)){
  eval(parse(text = paste("lines(x = c(geti$Lower.", 0.95 * 100,
                          ".CL[i], geti$Upper.", 0.95 * 100,
                          ".CL[i]), y = c(i, i))", sep = "")))
  if(geti$Test[i] == "Weaker") p.col <- res.col[1]
  if(geti$Test[i] == "ns" | is.na(geti$Test[i])) p.col <- res.col[2]
  if(geti$Test[i] == "Stronger") p.col <- res.col[3]
  graphics::points(geti$Observed[i], i, pch = 21, col = "black",
                   bg = p.col, cex = 2)
}
```

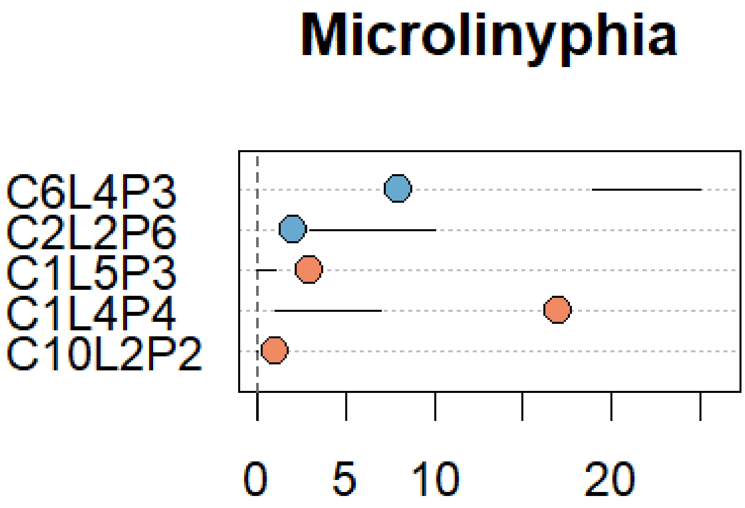


```
# Tenuiphantes

gtti <- test_interactions(genus.null, signif.level = 0.95)

## Warning in test_interactions(genus.null, signif.level = 0.95): Be careful
of
## Type I errors due to the large number of tests

gtti <- gtti[gtti$Consumer == "Tenuiphantes", ]
gtti[, 3] <- ifelse(rowSums(gtti[, 3:6]) == 0, NA, gtti[, 3])
gtti[, 4] <- ifelse(rowSums(gtti[, 3:6]) == 0, NA, gtti[, 4])
gtti[, 5] <- ifelse(rowSums(gtti[, 3:6]) == 0, NA, gtti[, 5])
gtti[, 6] <- ifelse(rowSums(gtti[, 3:6]) == 0, NA, gtti[, 6])

# EDIT 'ti' - to just the prey taxa that you want to show on the plot

gtti <- gtti[c(2,3,4,5,6,7,11,12,13,15,16,19),]

# Set up maximum x-axis value for xlim. Add an additional 5%
gtmin.x <- min(gtti[, 3:6], na.rm = TRUE)
gtmin.x <- max(0, gtmin.x, na.rm = TRUE)
gtmax.x <- max(gtti[, 3:6], na.rm = TRUE)
gtmax.x <- gtmax.x * 1.05
gtti$Setup <- seq(gtmin.x, gtmax.x, length.out = nrow(gtti))

# Plot built up in 2 stages: i) using min and max values to set the
#   y-axis range without having to use ylim (so this can be customised
#   by the user), ii) the main dbarplot and label, and iii) the error
graphics::dotchart(gtti$Setup, labels = paste(gtti$Resource, " ", sep = ""),
                   col = 1, pt.cex = 0, cex = 1.5, main = "Tenuiphantes")
graphics::abline(v = 0, lty = 2, col = "dimgrey")


res.col <- c("#67A9CF", "#F7F7F7", "#EF8A62")

for (i in 1:nrow(gtti)){
  eval(parse(text = paste("lines(x = c(gtti$Lower.", 0.95 * 100,
                          ".CL[i], gtti$Upper.", 0.95 * 100,
                          ".CL[i]), y = c(i, i))", sep = "")))
  if(gtti$Test[i] == "Weaker") p.col <- res.col[1]
  if(gtti$Test[i] == "ns" | is.na(gtti$Test[i])) p.col <- res.col[2]
  if(gtti$Test[i] == "Stronger") p.col <- res.col[3]
  graphics::points(gtti$Observed[i], i, pch = 21, col = "black",
                   bg = p.col, cex = 2)
}
```



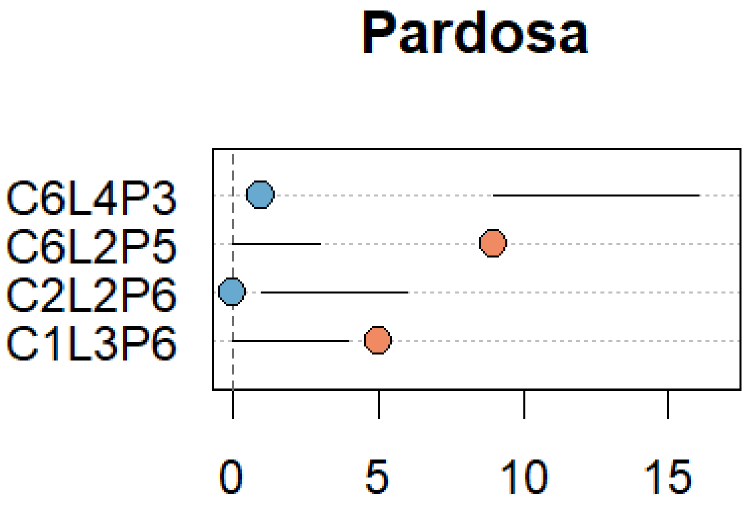
```
# Microlinyphia

gmti <- test_interactions(genus.null, signif.level = 0.95)

## Warning in test_interactions(genus.null, signif.level = 0.95): Be careful
of
## Type I errors due to the large number of tests
```

```r
gmti <- gmti[gmti$Consumer == "Microlinyphia", ]
gmti[, 3] <- ifelse(rowSums(gmti[, 3:6]) == 0, NA, gmti[, 3])
gmti[, 4] <- ifelse(rowSums(gmti[, 3:6]) == 0, NA, gmti[, 4])
gmti[, 5] <- ifelse(rowSums(gmti[, 3:6]) == 0, NA, gmti[, 5])
gmti[, 6] <- ifelse(rowSums(gmti[, 3:6]) == 0, NA, gmti[, 6])

# EDIT 'ti' - to just the prey taxa that you want to show on the plot

gmti <- gmti[c(1,3,4,5,16),]


# Set up maximum x-axis value for xlim. Add an additional 5%
gmmin.x <- min(gmti[, 3:6], na.rm = TRUE)
gmmin.x <- max(0, gmmin.x, na.rm = TRUE)
gmmax.x <- max(gmti[, 3:6], na.rm = TRUE)
gmmax.x <- gmmax.x * 1.05
gmti$Setup <- seq(gmmin.x, gmmax.x, length.out = nrow(gmti))

# Plot built up in 2 stages: i) using min and max values to set the
#   y-axis range without having to use ylim (so this can be customised
#   by the user), ii) the main dbarplot and label, and iii) the error
graphics::dotchart(gmti$Setup, labels = paste(gmti$Resource, " ", sep = ""),
                   col = 1, pt.cex = 0, cex = 1.5, main = "Microlinyphia")
graphics::abline(v = 0, lty = 2, col = "dimgrey")


res.col <- c("#67A9CF", "#F7F7F7", "#EF8A62")

for (i in 1:nrow(gmti)){
  eval(parse(text = paste("lines(x = c(gmti$Lower.", 0.95 * 100,
                          ".CL[i], gmti$Upper.", 0.95 * 100,
                          ".CL[i]), y = c(i, i))", sep = "")))
  if(gmti$Test[i] == "Weaker") p.col <- res.col[1]
  if(gmti$Test[i] == "ns" | is.na(gmti$Test[i])) p.col <- res.col[2]
  if(gmti$Test[i] == "Stronger") p.col <- res.col[3]
  graphics::points(gmti$Observed[i], i, pch = 21, col = "black",
                   bg = p.col, cex = 2)
}
```



```
# Pardosa

gpti <- test_interactions(genus.null, signif.level = 0.95)

## Warning in test_interactions(genus.null, signif.level = 0.95): Be careful
of
## Type I errors due to the large number of tests

gpti <- gpti[gpti$Consumer == "Pardosa", ]
gpti[, 3] <- ifelse(rowSums(gpti[, 3:6]) == 0, NA, gpti[, 3])
gpti[, 4] <- ifelse(rowSums(gpti[, 3:6]) == 0, NA, gpti[, 4])
gpti[, 5] <- ifelse(rowSums(gpti[, 3:6]) == 0, NA, gpti[, 5])
gpti[, 6] <- ifelse(rowSums(gpti[, 3:6]) == 0, NA, gpti[, 6])

# EDIT 'ti' - to just the prey taxa that you want to show on the plot

gpti <- gpti[c(2,5,15,16),]


# Set up maximum x-axis value for xlim. Add an additional 5%
gpmin.x <- min(gpti[, 3:6], na.rm = TRUE)
gpmin.x <- max(0, gpmin.x, na.rm = TRUE)
gpmax.x <- max(gpti[, 3:6], na.rm = TRUE)
gpmax.x <- gpmax.x * 1.05
gpti$Setup <- seq(gpmin.x, gpmax.x, length.out = nrow(gpti))
```

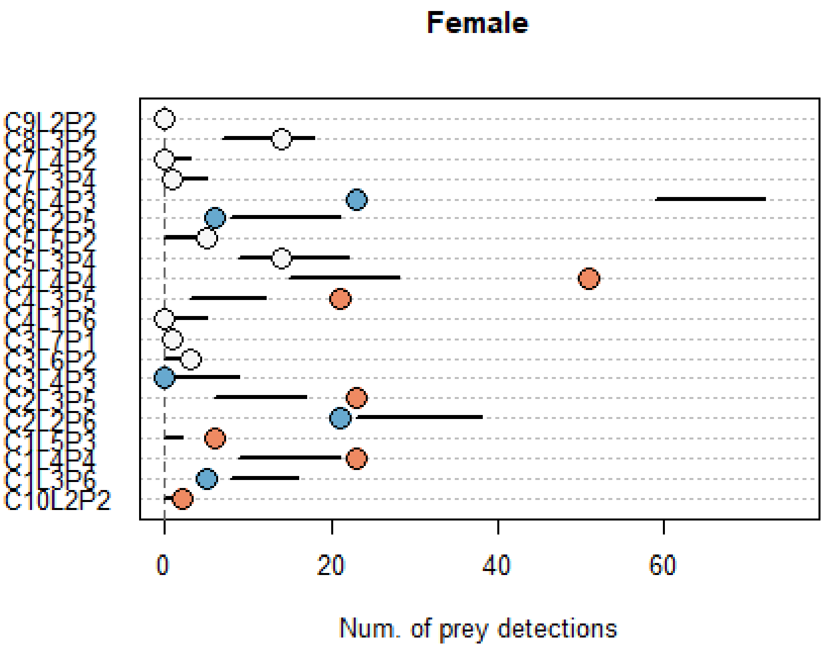
```
# Plot built up in 2 stages: i) using min and max values to set the
#   y-axis range without having to use ylim (so this can be customised
#   by the user), ii) the main dbarplot and label, and iii) the error
graphics::dotchart(gpti$Setup, labels = paste(gpti$Resource, " ", sep = ""),
                   col = 1, pt.cex = 0, cex = 1.5, main = "Pardosa")
graphics::abline(v = 0, lty = 2, col = "dimgrey")


res.col <- c("#67A9CF", "#F7F7F7", "#EF8A62")

for (i in 1:nrow(gpti)){
  eval(parse(text = paste("lines(x = c(gpti$Lower.", 0.95 * 100,
                          ".CL[i], gpti$Upper.", 0.95 * 100,
                          ".CL[i]), y = c(i, i))", sep = "")))
  if(gpti$Test[i] == "Weaker") p.col <- res.col[1]
  if(gpti$Test[i] == "ns" | is.na(gpti$Test[i])) p.col <- res.col[2]
  if(gpti$Test[i] == "Stronger") p.col <- res.col[3]
  graphics::points(gpti$Observed[i], i, pch = 21, col = "black",
                   bg = p.col, cex = 2)
}
```



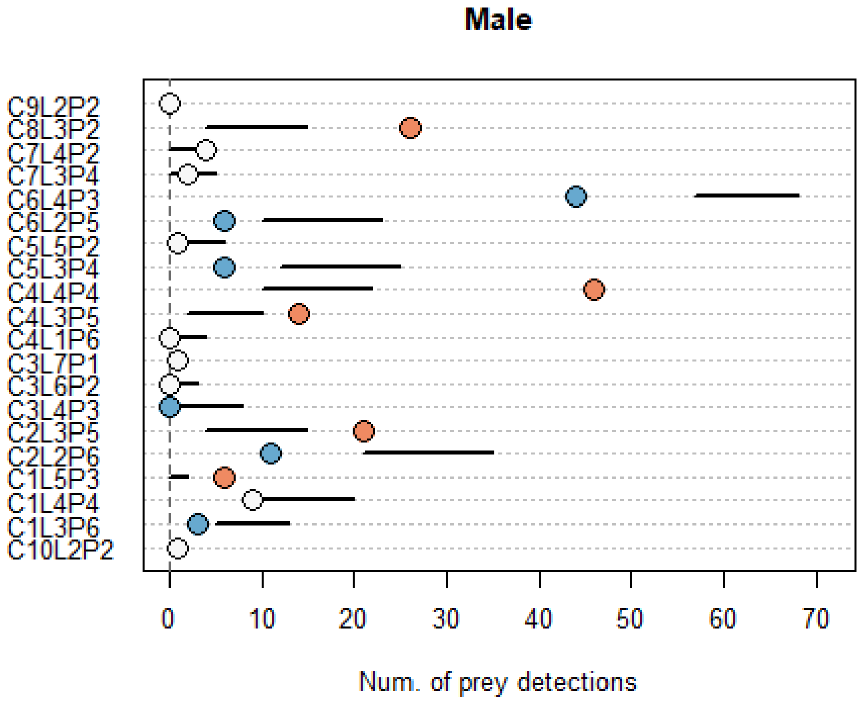
Tropho-species ENNR for sexes

And we now build the model, plot the overall preferences and extract the model data for sex.

```
sextsennr <- read.csv("TS_ENNR_Diet_Sexbin.csv")
tsinvertsennr <- read.csv("TS_ENNR_Inverts.csv")
sextsENNR.fl <- read.csv("TS_ENNR_Diet.fl_Sex.csv")

sex.null <- generate_null_net(sextsennr[,2:22], tsinvertsennr[,2:21],
                              sims = 999, data.type = "names",
                              summary.type = "sum",
                              r.samples = tsinvertsennr[,1],
                              c.samples = sextsennr[,1],
                              r.weights = sextsENNR.fl)

## Warning in generate_null_net(sextsennr[, 2:22], tsinvertsennr[, 2:21], sim
s = 999, : One or more instances detected where a consumer interacted with a
##        resource that has zero abundance in 'resources'

#par(mfrow = c(1,2))
par(mfrow = c(1,1))
plot_preferences(sex.null, "Female", signif.level = 0.95, type = "counts",
                 xlab = "Num. of prey detections", p.cex = 1.5, l.cex = 0.8,
lwd = 2)

## Warning in test_interactions(nullnet, signif.level = signif.level): Be car
eful
## of Type I errors due to the large number of tests
```



```
plot_preferences(sex.null, "Male", signif.level = 0.95, type = "counts",
                 xlab = "Num. of prey detections", p.cex = 1.5, l.cex = 0.8,
lwd = 2)

## Warning in test_interactions(nullnet, signif.level = signif.level): Be car
eful
## of Type I errors due to the large number of tests
```

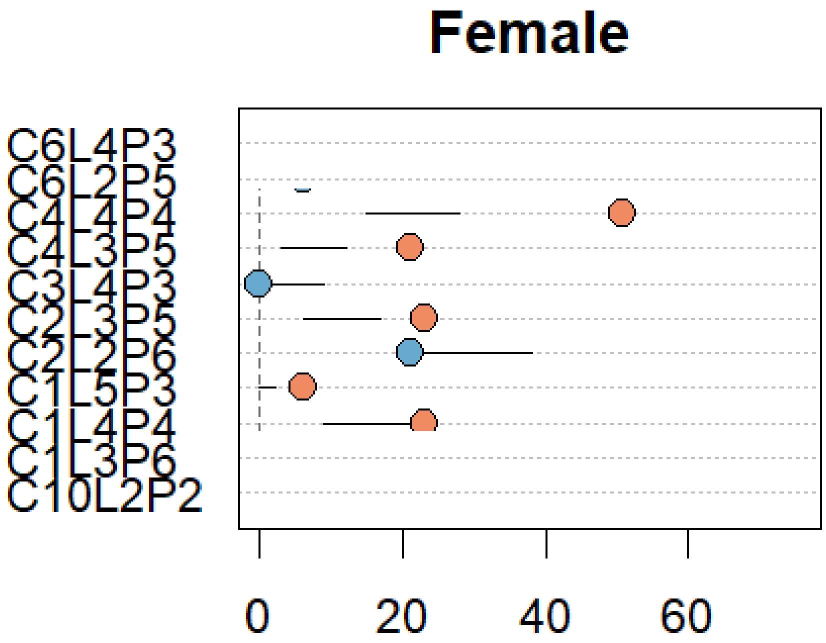


```
sex.links <- test_interactions(sex.null, signif.level = 0.95)

## Warning in test_interactions(sex.null, signif.level = 0.95): Be careful of
Type
## I errors due to the large number of tests
```

And then plot the significant results.

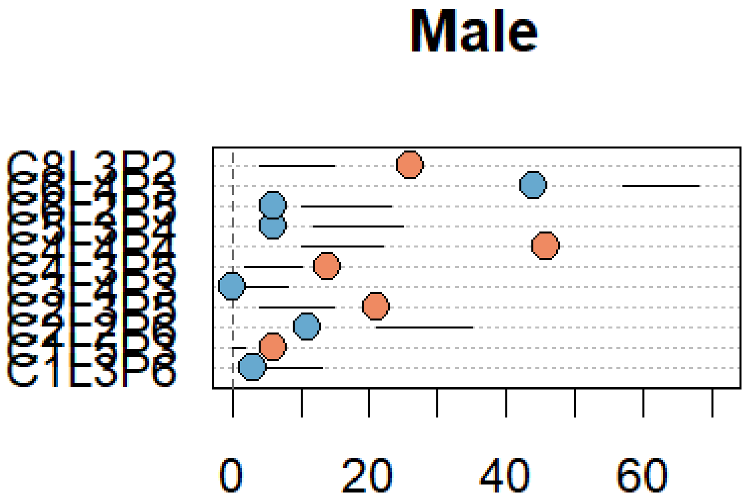
```
# Female

sfti <- test_interactions(sex.null, signif.level = 0.95)

## Warning in test_interactions(sex.null, signif.level = 0.95): Be careful of
Type
## I errors due to the large number of tests

sfti <- sfti[sfti$Consumer == "Female", ]
sfti[, 3] <- ifelse(rowSums(sfti[, 3:6]) == 0, NA, sfti[, 3])
sfti[, 4] <- ifelse(rowSums(sfti[, 3:6]) == 0, NA, sfti[, 4])
sfti[, 5] <- ifelse(rowSums(sfti[, 3:6]) == 0, NA, sfti[, 5])
```

```
sfti[, 6] <- ifelse(rowSums(sfti[, 3:6]) == 0, NA, sfti[, 6])

# EDIT 'ti' - to just the prey taxa that you want to show on the plot

sfti <- sfti[c(1,2,3,4,5,6,7,11,12,15,16),]

# Set up maximum x-axis value for xlim. Add an additional 5%
sfmin.x <- min(sfti[, 3:6], na.rm = TRUE)
sfmin.x <- max(0, sfmin.x, na.rm = TRUE)
sfmax.x <- max(sfti[, 3:6], na.rm = TRUE)
sfmax.x <- sfmax.x * 1.05
sfti$Setup <- seq(sfmin.x, sfmax.x, length.out = nrow(sfti))

# Plot built up in 2 stages: i) using min and max values to set the
#   y-axis range without having to use ylim (so this can be customised
#   by the user), ii) the main dbarplot and label, and iii) the error
graphics::dotchart(sfti$Setup, labels = paste(sfti$Resource, " ", sep = ""),
                   col = 1, pt.cex = 0, cex = 1.5, main = "Female")
graphics::abline(v = 0, lty = 2, col = "dimgrey")


res.col <- c("#67A9CF", "#F7F7F7", "#EF8A62")

for (i in 1:nrow(sfti)){
  eval(parse(text = paste("lines(x = c(sfti$Lower.", 0.95 * 100,
                          ".CL[i], sfti$Upper.", 0.95 * 100,
                          ".CL[i]), y = c(i, i))", sep = "")))
  if(sfti$Test[i] == "Weaker") p.col <- res.col[1]
  if(sfti$Test[i] == "ns" | is.na(sfti$Test[i])) p.col <- res.col[2]
  if(sfti$Test[i] == "Stronger") p.col <- res.col[3]
  graphics::points(sfti$Observed[i], i, pch = 21, col = "black",
                   bg = p.col, cex = 2)
}
```



```
# Male

smti <- test_interactions(sex.null, signif.level = 0.95)

## Warning in test_interactions(sex.null, signif.level = 0.95): Be careful of
Type
## I errors due to the large number of tests

smti <- smti[smti$Consumer == "Male", ]
smti[, 3] <- ifelse(rowSums(smti[, 3:6]) == 0, NA, smti[, 3])
smti[, 4] <- ifelse(rowSums(smti[, 3:6]) == 0, NA, smti[, 4])
smti[, 5] <- ifelse(rowSums(smti[, 3:6]) == 0, NA, smti[, 5])
smti[, 6] <- ifelse(rowSums(smti[, 3:6]) == 0, NA, smti[, 6])


# EDIT 'ti' - to just the prey taxa that you want to show on the plot

smti <- smti[c(2,4,5,6,7,11,12,13,15,16,19),]


# Set up maximum x-axis value for xlim. Add an additional 5%
smmin.x <- min(smti[, 3:6], na.rm = TRUE)
smmin.x <- max(0, smmin.x, na.rm = TRUE)
smmax.x <- max(smti[, 3:6], na.rm = TRUE)
smmax.x <- smmax.x * 1.05
smti$Setup <- seq(smmin.x, smmax.x, length.out = nrow(smti))
```

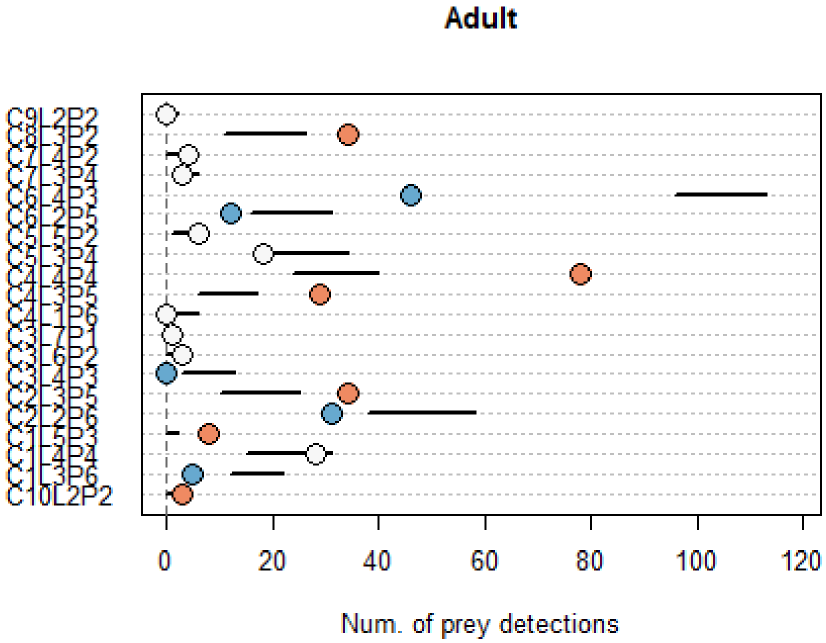
```
# Plot built up in 2 stages: i) using min and max values to set the
#   y-axis range without having to use ylim (so this can be customised
#   by the user), ii) the main dbarplot and label, and iii) the error
graphics::dotchart(smti$Setup, labels = paste(smti$Resource, " ", sep = ""),
                   col = 1, pt.cex = 0, cex = 1.5, main = "Male")
graphics::abline(v = 0, lty = 2, col = "dimgrey")


res.col <- c("#67A9CF", "#F7F7F7", "#EF8A62")

for (i in 1:nrow(smti)){
  eval(parse(text = paste("lines(x = c(smti$Lower.", 0.95 * 100,
                          ".CL[i], smti$Upper.", 0.95 * 100,
                          ".CL[i]), y = c(i, i))", sep = "")))
  if(smti$Test[i] == "Weaker") p.col <- res.col[1]
  if(smti$Test[i] == "ns" | is.na(smti$Test[i])) p.col <- res.col[2]
  if(smti$Test[i] == "Stronger") p.col <- res.col[3]
  graphics::points(smti$Observed[i], i, pch = 21, col = "black",
                   bg = p.col, cex = 2)
}
```



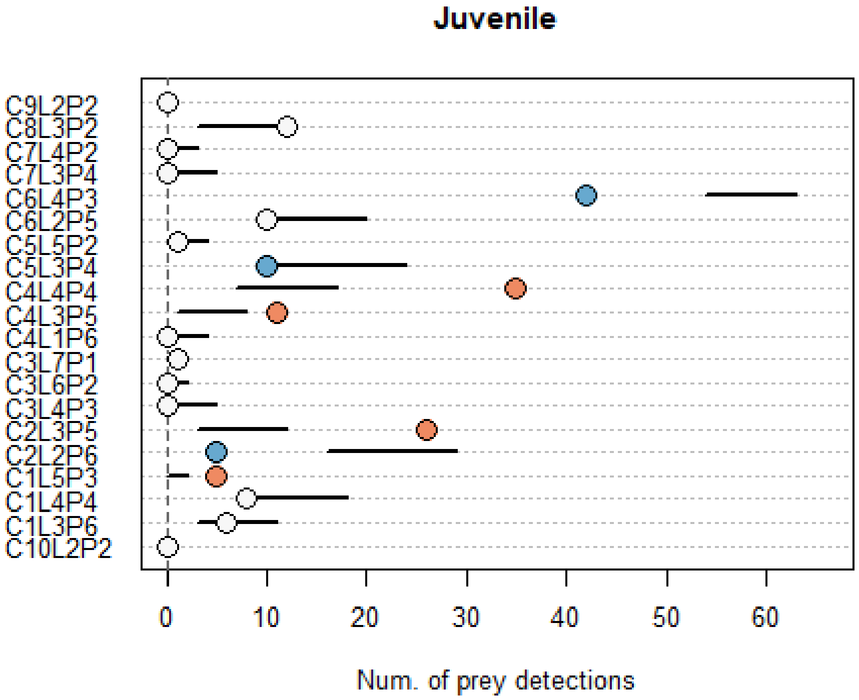
Tropho-species ENNR for life stages

And, finally, we build the model, produce overall preference plots and extract data for life stages.

```
lifetsennr <- read.csv("TS_ENNR_Diet_Lifebin.csv")
tsinvertsennr <- read.csv("TS_ENNR_Inverts.csv")
lifetsENNR.fl <- read.csv("TS_ENNR_Diet.fl_Life.csv")

life.null <- generate_null_net(lifetsennr[,2:22], tsinvertsennr[,2:21],
                               sims = 999, data.type = "names",
                               summary.type = "sum",
                               r.samples = tsinvertsennr[,1],
                               c.samples = lifetsennr[,1],
                               r.weights = lifetsENNR.fl)

## Warning in generate_null_net(lifetsennr[, 2:22], tsinvertsennr[, 2:21], :
One or more instances detected where a consumer interacted with a
##          resource that has zero abundance in 'resources'

#par(mfrow = c(1,2))
par(mfrow = c(1,1))
plot_preferences(life.null, "Adult", signif.level = 0.95, type = "counts",
                 xlab = "Num. of prey detections", p.cex = 1.5, l.cex = 0.8,
lwd = 2)
```



```
plot_preferences(life.null, "Juvenile", signif.level = 0.95, type = "counts",
                 xlab = "Num. of prey detections", p.cex = 1.5, l.cex = 0.8,
lwd = 2)
```

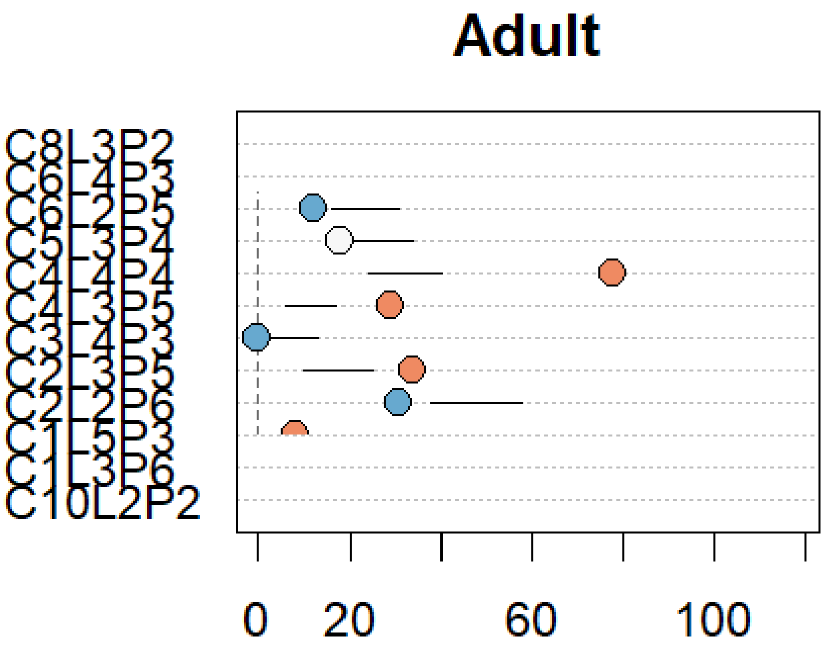


```
life.links <- test_interactions(life.null, signif.level = 0.95)
```

And then plot the significant results.

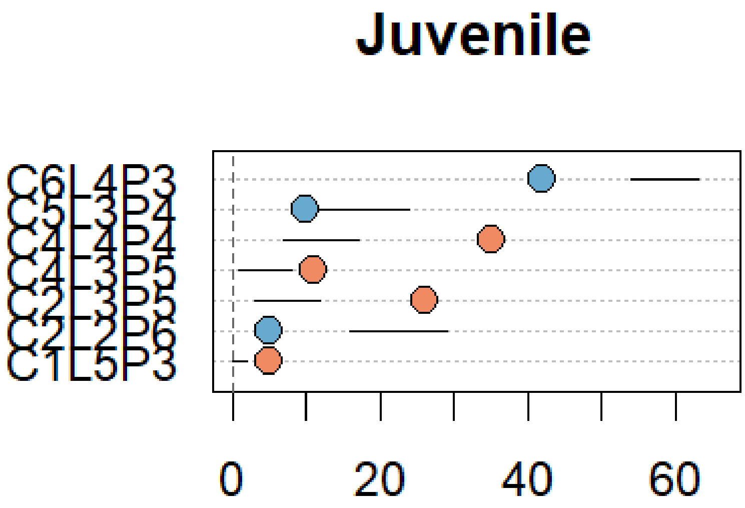
```
# Adult

lati <- test_interactions(life.null, signif.level = 0.95)
lati <- lati[lati$Consumer == "Adult", ]
lati[, 3] <- ifelse(rowSums(lati[, 3:6]) == 0, NA, lati[, 3])
lati[, 4] <- ifelse(rowSums(lati[, 3:6]) == 0, NA, lati[, 4])
lati[, 5] <- ifelse(rowSums(lati[, 3:6]) == 0, NA, lati[, 5])
lati[, 6] <- ifelse(rowSums(lati[, 3:6]) == 0, NA, lati[, 6])



# EDIT 'ti' - to just the prey taxa that you want to show on the plot

lati <- lati[c(1,2,4,5,6,7,11,12,13,15,16,19),]



# Set up maximum x-axis value for xlim. Add an additional 5%
lamin.x <- min(lati[, 3:6], na.rm = TRUE)
lamin.x <- max(0, lamin.x, na.rm = TRUE)
lamax.x <- max(lati[, 3:6], na.rm = TRUE)
lamax.x <- lamax.x * 1.05
lati$Setup <- seq(lamin.x, lamax.x, length.out = nrow(lati))
```

```
# Plot built up in 2 stages: i) using min and max values to set the
#   y-axis range without having to use ylim (so this can be customised
#   by the user), ii) the main dbarplot and label, and iii) the error
graphics::dotchart(lati$Setup, labels = paste(lati$Resource, " ", sep = ""),
                   col = 1, pt.cex = 0, cex = 1.5, main = "Adult")
graphics::abline(v = 0, lty = 2, col = "dimgrey")


res.col <- c("#67A9CF", "#F7F7F7", "#EF8A62")

for (i in 1:nrow(lati)){
  eval(parse(text = paste("lines(x = c(lati$Lower.", 0.95 * 100,
                          ".CL[i], lati$Upper.", 0.95 * 100,
                          ".CL[i]), y = c(i, i))", sep = "")))
  if(lati$Test[i] == "Weaker") p.col <- res.col[1]
  if(lati$Test[i] == "ns" | is.na(lati$Test[i])) p.col <- res.col[2]
  if(lati$Test[i] == "Stronger") p.col <- res.col[3]
  graphics::points(lati$Observed[i], i, pch = 21, col = "black",
                   bg = p.col, cex = 2)
}
```



```
# Juvenile

ljti <- test_interactions(life.null, signif.level = 0.95)
ljti <- ljti[ljti$Consumer == "Juvenile", ]
ljti[, 3] <- ifelse(rowSums(ljti[, 3:6]) == 0, NA, ljti[, 3])
```

```r
ljti[, 4] <- ifelse(rowSums(ljti[, 3:6]) == 0, NA, ljti[, 4])
ljti[, 5] <- ifelse(rowSums(ljti[, 3:6]) == 0, NA, ljti[, 5])
ljti[, 6] <- ifelse(rowSums(ljti[, 3:6]) == 0, NA, ljti[, 6])


# EDIT 'ti' - to just the prey taxa that you want to show on the plot

ljti <- ljti[c(4,5,6,11,12,13,16),]


# Set up maximum x-axis value for xlim. Add an additional 5%
ljmin.x <- min(ljti[, 3:6], na.rm = TRUE)
ljmin.x <- max(0, ljmin.x, na.rm = TRUE)
ljmax.x <- max(ljti[, 3:6], na.rm = TRUE)
ljmax.x <- ljmax.x * 1.05
ljti$Setup <- seq(ljmin.x, ljmax.x, length.out = nrow(ljti))

# Plot built up in 2 stages: i) using min and max values to set the
#   y-axis range without having to use ylim (so this can be customised
#   by the user), ii) the main dbarplot and label, and iii) the error
graphics::dotchart(ljti$Setup, labels = paste(ljti$Resource, " ", sep = ""),
                   col = 1, pt.cex = 0, cex = 1.5, main = "Juvenile")
graphics::abline(v = 0, lty = 2, col = "dimgrey")


res.col <- c("#67A9CF", "#F7F7F7", "#EF8A62")

for (i in 1:nrow(ljti)){
  eval(parse(text = paste("lines(x = c(ljti$Lower.", 0.95 * 100,
                          ".CL[i], ljti$Upper.", 0.95 * 100,
                          ".CL[i]), y = c(i, i))", sep = "")))
  if(ljti$Test[i] == "Weaker") p.col <- res.col[1]
  if(ljti$Test[i] == "ns" | is.na(ljti$Test[i])) p.col <- res.col[2]
  if(ljti$Test[i] == "Stronger") p.col <- res.col[3]
  graphics::points(ljti$Observed[i], i, pch = 21, col = "black",
                   bg = p.col, cex = 2)
}
```



Ex situ prey choice assays

```
exsitu <- read.csv("exsitu.csv")
summary(exsitu)

##     Spider              Sex            Maturity.post.starve     Diet          
##  Length:54          Length:54          Length:54            Length:54         
##  Class :character   Class :character   Class :character     Class :charact
er  
##  Mode  :character   Mode  :character   Mode  :character     Mode  :charact
er  
##                                                                           
##                                                                           
##                                                                           
##                                                                           
##   Initial.mass        Mortality        Eggs     Spiderlings
##  Min.   :0.0002400   Min.   :2.000   Min.   :0    Min.   :2   
##  1st Qu.:0.0007575   1st Qu.:2.000   1st Qu.:0    1st Qu.:2   
##  Median :0.0010000   Median :3.000   Median :0    Median :2   
##  Mean   :0.0010998   Mean   :3.048   Mean   :0    Mean   :2   
##  3rd Qu.:0.0014750   3rd Qu.:4.000   3rd Qu.:0    3rd Qu.:2   
##  Max.   :0.0020000   Max.   :5.000   Max.   :0    Max.   :2   
##                      NA's   :33      NA's   :43   NA's   :51  
##   Mass.change         First.meal         Time.to.eat       Aphid.time    
##  Min.   :-0.008390   Min.   :character  Min.   : 0.000   Min.   : 0.000  
##  1st Qu.:-0.000340   Class :character   1st Qu.: 0.000   1st Qu.: 0.000  
##  Median :-0.000160   Mode  :character   Median : 0.000   Median : 0.500  
##  Mean   :-0.000462                      Mean   : 2.344   Mean   : 9.844
##  3rd Qu.: 0.000030                      3rd Qu.: 0.000   3rd Qu.:24.000
##  Max.   : 0.000320                      Max.   :24.000   Max.   :48.000
##  NA's   :21                             NA's   :22       NA's   :22
##     Fly.time     Springtail.time
##  Min.   : 0.00   Min.   : 0.00
##  1st Qu.:12.00   1st Qu.: 1.00
##  Median :24.00   Median :12.00
##  Mean   :17.52   Mean   :13.66
##  3rd Qu.:24.00   3rd Qu.:24.00
##  Max.   :48.00   Max.   :48.00
##  NA's   :23      NA's   :22

exsitu$Diet <- as.factor(exsitu$Diet)
exsitu$First.meal <- as.factor(exsitu$First.meal)
exsitu$Sex <- as.factor(exsitu$Sex)
exsitu$Maturity <- as.factor(exsitu$Maturity.post.starve)
exsitu$Sex <- as.factor(exsitu$Sex)
```

The data can be subsetted for separate analyses of prey choice and mortality.

```
mortality <- subset(exsitu, is.na(Mass.change))
summary(mortality)

##     Spider              Sex     Maturity.post.starve         Diet
##  Length:21          Female: 6   Length:21          Aphid     :6
##  Class :character   Male  :10   Class :character   Fly       :8
##  Mode  :character   N/A   : 5   Mode  :character   Springtail:7
##
##
##
##
##   Initial.mass        Mortality          Eggs     Spiderlings  Mass.change
##  Min.   :0.000400   Min.   :2.000   Min.   :0    Min.   :2    Min.   : NA
##  1st Qu.:0.000710   1st Qu.:2.000   1st Qu.:0    1st Qu.:2    1st Qu.: NA
##  Median :0.000950   Median :3.000   Median :0    Median :2    Median : NA
##  Mean   :0.001016   Mean   :3.048   Mean   :0    Mean   :2    Mean   :NaN
##  3rd Qu.:0.001300   3rd Qu.:4.000   3rd Qu.:0    3rd Qu.:2    3rd Qu.: NA
##  Max.   :0.002000   Max.   :5.000   Max.   :0    Max.   :2    Max.   : NA
##                                     NA's   :18   NA's   :20   NA's   :21
##              First.meal  Time.to.eat    Aphid.time      Fly.time
##                   : 0   Min.   : NA   Min.   : NA   Min.   : NA
##  Aphid           : 0   1st Qu.: NA   1st Qu.: NA   1st Qu.: NA
##  Aphid&Springtail: 0   Median : NA   Median : NA   Median : NA
##  Fly             : 0   Mean   :NaN   Mean   :NaN   Mean   :NaN
##  Fly&Springtail  : 0   3rd Qu.: NA   3rd Qu.: NA   3rd Qu.: NA
##  Springtail      : 0   Max.   : NA   Max.   : NA   Max.   : NA
##  NA's            :21   NA's   :21    NA's   :21    NA's   :21
##  Springtail.time     Maturity
##  Min.   : NA     Adult   :14
##  1st Qu.: NA     Juvenile: 7
##  Median : NA  
##  Mean   :NaN  
##  3rd Qu.: NA  
##  Max.   : NA  
##  NA's   :21
```

```
choice <- subset(exsitu, is.na(Mortality))
summary(choice)
```

```
##     Spider              Sex     Maturity.post.starve         Diet
##  Length:33          Female:22   Length:33            Aphid     :14
##  Class :character   Male  : 7   Class :character     Fly       :13
##  Mode  :character   N/A   : 4   Mode  :character     Springtail: 6
##
##
##
##
##   Initial.mass       Mortality        Eggs     Spiderlings
##  Min.   :0.000240   Min.   : NA   Min.   :0    Min.   :2
##  1st Qu.:0.000800   1st Qu.: NA   1st Qu.:0    1st Qu.:2
##  Median :0.001200   Median : NA   Median :0    Median :2
##  Mean   :0.001153   Mean   :NaN   Mean   :0    Mean   :2
##  3rd Qu.:0.001500   3rd Qu.: NA   3rd Qu.:0    3rd Qu.:2
##  Max.   :0.001940   Max.   : NA   Max.   :0    Max.   :2
##                     NA's   :33    NA's   :25   NA's   :31
##   Mass.change                    First.meal  Time.to.eat       Aphid.time
##  Min.   :-0.0083900                    : 7   Min.   : 0.000   Min.   : 0.00
0
##  1st Qu.:-0.0003400   Aphid           :16   1st Qu.: 0.000   1st Qu.: 0.00
0
##  Median :-0.0001600   Aphid&Springtail: 1   Median : 0.000   Median : 0.50
0
##  Mean   :-0.0004615   Fly             : 3   Mean   : 2.344   Mean   : 9.84
4
##  3rd Qu.: 0.0000300   Fly&Springtail  : 1   3rd Qu.: 0.000   3rd Qu.:24.00
0
##  Max.   : 0.0003200   Springtail      : 5   Max.   :24.000   Max.   :48.00
0
##                                             NA's   :1        NA's   :1
##     Fly.time     Springtail.time     Maturity
##  Min.   : 0.00   Min.   : 0.00   Adult   :26
##  1st Qu.:12.00   1st Qu.: 1.00   Juvenile: 7
##  Median :24.00   Median :12.00
##  Mean   :17.52   Mean   :13.66
##  3rd Qu.:24.00   3rd Qu.:24.00
##  Max.   :48.00   Max.   :48.00
##  NA's   :2       NA's   :1
```

An analysis of ex situ choice can be carried out via MANOVA.

```
mandiet <- manova(cbind(Aphid.time, Springtail.time, Fly.time, Mass.change, T
ime.to.eat) ~ Diet, data=choice, na.action=na.omit)

summary(mandiet)

##           Df  Pillai approx F num Df den Df Pr(>F)
## Diet       2 0.44811   1.4438     10     50 0.1892
## Residuals 28

summary.aov(mandiet)

##  Response Aphid.time :
##             Df Sum Sq Mean Sq F value Pr(>F)
## Diet         2   72.3  36.142  0.1702 0.8444
## Residuals   28 5945.9 212.354
##
##  Response Springtail.time :
##             Df Sum Sq Mean Sq F value Pr(>F)
## Diet         2  201.4  100.68  0.4614 0.6351
## Residuals   28 6109.4  218.19
##
##  Response Fly.time :
##             Df Sum Sq Mean Sq F value Pr(>F)
## Diet         2  310.8  155.41  1.0488 0.3637
## Residuals   28 4148.9  148.18
##
##  Response Mass.change :
##             Df     Sum Sq    Mean Sq F value Pr(>F)
## Diet         2 9.5010e-06 4.7504e-06  2.2679 0.1222
## Residuals   28 5.8649e-05 2.0946e-06
##
##  Response Time.to.eat :
##             Df  Sum Sq Mean Sq F value Pr(>F)
## Diet         2   42.11  21.053  0.4834 0.6217
## Residuals   28 1219.44  43.552
##
## 2 observations deleted due to missingness
```

A GLM of mortality can analyse any effect of diet or other variables on mortality of spiders in the experiment.

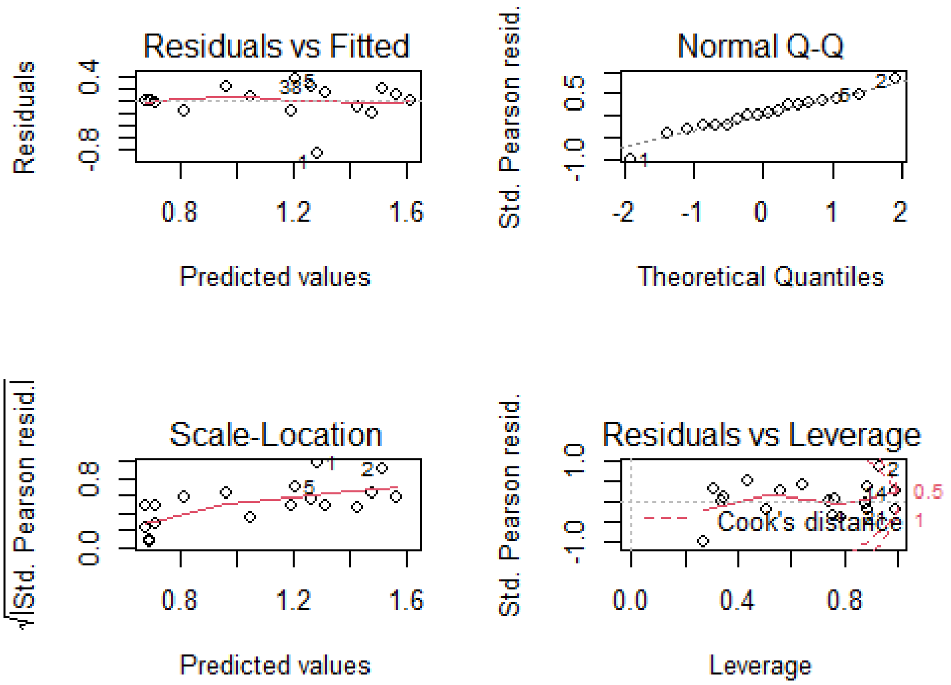
```
mort <- glm(Mortality ~ Initial.mass + Diet + Sex + Maturity +
              Initial.mass:Diet + Initial.mass:Sex + Initial.mass:Maturity +
              Diet:Sex + Diet:Maturity + Sex:Maturity
              , data=mortality, family = poisson, na.action=na.omit)

summary(mort)

##
## Call:
## glm(formula = Mortality ~ Initial.mass + Diet + Sex + Maturity +
##     Initial.mass:Diet + Initial.mass:Sex + Initial.mass:Maturity +
##     Diet:Sex + Diet:Maturity + Sex:Maturity, family = poisson,
##     data = mortality, na.action = na.omit)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -0.92295  -0.02653   0.00295   0.11127   0.35210
##
## Coefficients: (5 not defined because of singularities)
##                                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)                          7.673      6.318   1.214    0.225
## Initial.mass                     -3441.662   3398.005  -1.013    0.311
## DietFly                             -7.164      6.305  -1.136    0.256
## DietSpringtail                      -8.410      6.232  -1.350    0.177
## SexMale                             -2.736      4.123  -0.663    0.507
## SexN/A                             -14.029      9.740  -1.440    0.150
## MaturityJuvenile                     9.284      8.531   1.088    0.276
## Initial.mass:DietFly              3891.772   5069.945   0.768    0.443
## Initial.mass:DietSpringtail       4522.911   3357.149   1.347    0.178
## Initial.mass:SexMale              -311.787   1934.972  -0.161    0.872
## Initial.mass:SexN/A              17865.539  17331.915   1.031    0.303
## Initial.mass:MaturityJuvenile   -18464.138  17193.857  -1.074    0.283
## DietFly:SexMale                      2.799      4.156   0.673    0.501
## DietSpringtail:SexMale               3.985      3.013   1.323    0.186
## DietFly:SexN/A                       5.054      5.261   0.961    0.337
## DietSpringtail:SexN/A                   NA         NA      NA       NA
## DietFly:MaturityJuvenile                NA         NA      NA       NA
## DietSpringtail:MaturityJuvenile         NA         NA      NA       NA
## SexMale:MaturityJuvenile                NA         NA      NA       NA
## SexN/A:MaturityJuvenile                 NA         NA      NA       NA
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 8.5977  on 20  degrees of freedom
## Residual deviance: 1.2764  on  6  degrees of freedom
## AIC: 93.16
##
## Number of Fisher Scoring iterations: 4
```

```
par(mfrow=c(2,2))
plot(mort)
```

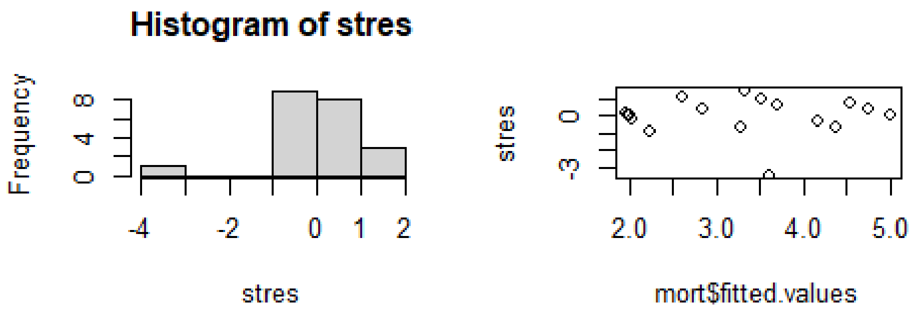
```
## Warning: not plotting observations with leverage one:
##   6, 8, 19
```

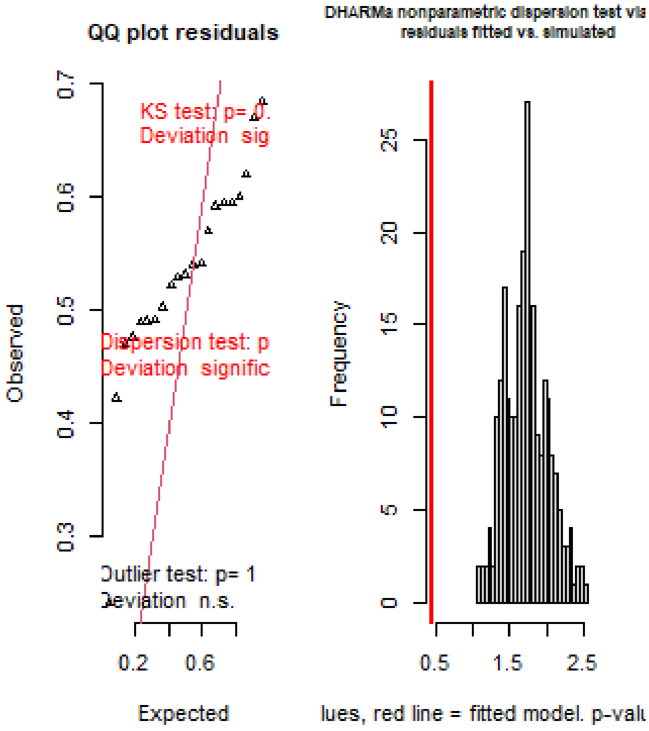
```
## Warning in sqrt(crit * p * (1 - hh)/hh): NaNs produced
```

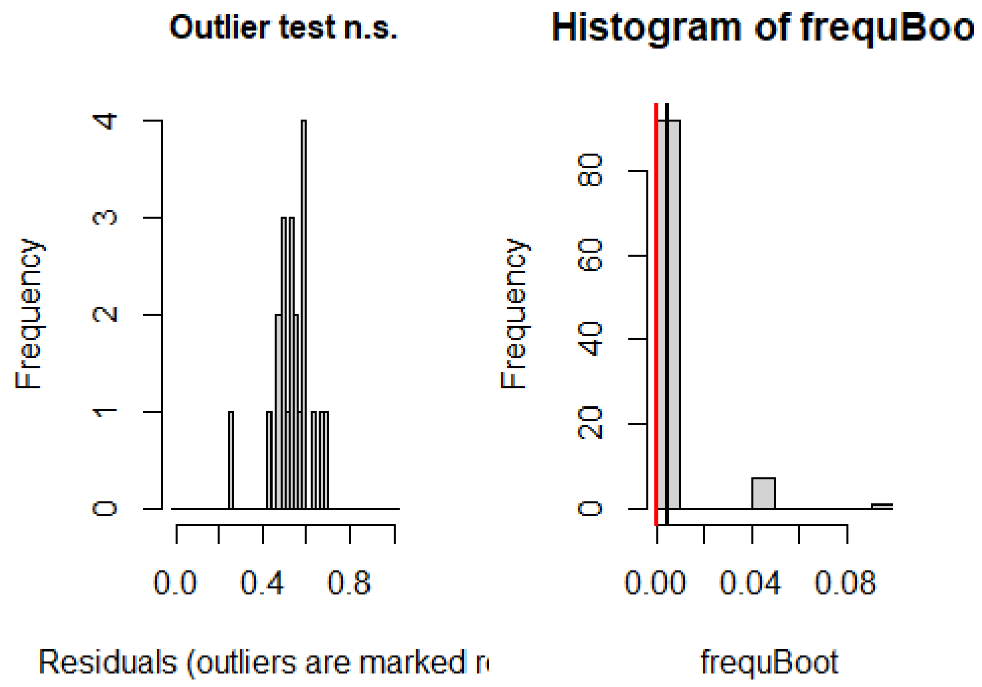
```
## Warning in sqrt(crit * p * (1 - hh)/hh): NaNs produced
```



```
stres<- (mort$residuals - mean(mort$residuals))/sd(mort$residuals)
hist(stres)
plot(stres ~ mort$fitted.values)
theta <- mort$deviance/mort$df.residual
theta

## [1] 0.212729

testResiduals(mort, plot = T)
```


Histogram of stres
Frequency
0
4
8
-4
-2
0
1
2
stres
stres
0
-3
2.0
3.0
4.0
5.0
mort$fitted.values


QQ plot residuals
KS test: p= 0.
Deviation sig
Dispersion test: p
Deviation signific
Outlier test: p= 1
Deviation n.s.
Observed
0.3
0.4
0.5
0.6
0.7
0.2
0.6
Expected
DHARMa nonparametric dispersion test via
residuals fitted vs. simulated
Frequency
0
5
10
15
20
25
0.5
1.5
2.5
lues, red line = fitted model. p-valu



```
## $uniformity
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  simulationOutput$scaledResiduals
## D = 0.3748, p-value = 0.003705
## alternative hypothesis: two-sided
##
##
## $dispersion
##
##  DHARMa nonparametric dispersion test via sd of residuals fitted vs.
##  simulated
##
## data:  simulationOutput
## ratioObsSim = 0.26162, p-value < 2.2e-16
## alternative hypothesis: two.sided
##
##
## $outliers
##
##  DHARMa bootstrapped outlier test
##
## data:  simulationOutput
## outliers at both margin(s) = 0, observations = 21, p-value = 1
## alternative hypothesis: two.sided
##  percent confidence interval:
```

```
##  0.00000000 0.04761905
## sample estimates:
## outlier frequency (expected: 0.00428571428571429 )
##                                                   0

## $uniformity
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  simulationOutput$scaledResiduals
## D = 0.3748, p-value = 0.003705
## alternative hypothesis: two-sided
##
##
## $dispersion
##
##  DHARMa nonparametric dispersion test via sd of residuals fitted vs.
##  simulated
##
## data:  simulationOutput
## ratioObsSim = 0.26162, p-value < 2.2e-16
## alternative hypothesis: two.sided
##
##
## $outliers
##
##  DHARMa bootstrapped outlier test
##
## data:  simulationOutput
## outliers at both margin(s) = 0, observations = 21, p-value = 1
## alternative hypothesis: two.sided
##  percent confidence interval:
##  0.00000000 0.04761905
## sample estimates:
## outlier frequency (expected: 0.00428571428571429 )
##                                                   0
```

```
mortality$Diet <- relevel(mortality$Diet, ref = "Fly")

summary.glm(mort)
```

```
##
## Call:
## glm(formula = Mortality ~ Initial.mass + Diet + Sex + Maturity +
##     Initial.mass:Diet + Initial.mass:Sex + Initial.mass:Maturity +
##     Diet:Sex + Diet:Maturity + Sex:Maturity, family = poisson,
##     data = mortality, na.action = na.omit)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -0.92295  -0.02653   0.00295   0.11127   0.35210  
## 
## Coefficients: (5 not defined because of singularities)
##                                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)                          7.673      6.318   1.214    0.225
## Initial.mass                     -3441.662   3398.005  -1.013    0.311
## DietFly                             -7.164      6.305  -1.136    0.256
## DietSpringtail                      -8.410      6.232  -1.350    0.177
## SexMale                             -2.736      4.123  -0.663    0.507
## SexN/A                             -14.029      9.740  -1.440    0.150
## MaturityJuvenile                     9.284      8.531   1.088    0.276
## Initial.mass:DietFly              3891.772   5069.945   0.768    0.443
## Initial.mass:DietSpringtail       4522.911   3357.149   1.347    0.178
## Initial.mass:SexMale              -311.787   1934.972  -0.161    0.872
## Initial.mass:SexN/A              17865.539  17331.915   1.031    0.303
## Initial.mass:MaturityJuvenile   -18464.138  17193.857  -1.074    0.283
## DietFly:SexMale                      2.799      4.156   0.673    0.501
## DietSpringtail:SexMale               3.985      3.013   1.323    0.186
## DietFly:SexN/A                       5.054      5.261   0.961    0.337
## DietSpringtail:SexN/A                   NA         NA      NA       NA
## DietFly:MaturityJuvenile                NA         NA      NA       NA
## DietSpringtail:MaturityJuvenile         NA         NA      NA       NA
## SexMale:MaturityJuvenile                NA         NA      NA       NA
## SexN/A:MaturityJuvenile                 NA         NA      NA       NA
## 
## (Dispersion parameter for poisson family taken to be 1)
## 
##     Null deviance: 8.5977  on 20  degrees of freedom
## Residual deviance: 1.2764  on  6  degrees of freedom
## AIC: 93.16
## 
## Number of Fisher Scoring iterations: 4
```

```
summary(mort)
```

```
## 
## Call:
## glm(formula = Mortality ~ Initial.mass + Diet + Sex + Maturity + 
##     Initial.mass:Diet + Initial.mass:Sex + Initial.mass:Maturity + 
##     Diet:Sex + Diet:Maturity + Sex:Maturity, family = poisson, 
##     data = mortality, na.action = na.omit)
## 
## Deviance Residuals: 
##      Min        1Q    Median        3Q       Max  
## -0.92295  -0.02653   0.00295   0.11127   0.35210  
## 
## Coefficients: (5 not defined because of singularities)
##                                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)                          7.673      6.318   1.214    0.225
## Initial.mass                     -3441.662   3398.005  -1.013    0.311
## DietFly                             -7.164      6.305  -1.136    0.256
## DietSpringtail                      -8.410      6.232  -1.350    0.177
## SexMale                             -2.736      4.123  -0.663    0.507
## SexN/A                             -14.029      9.740  -1.440    0.150
## MaturityJuvenile                     9.284      8.531   1.088    0.276
## Initial.mass:DietFly              3891.772   5069.945   0.768    0.443
## Initial.mass:DietSpringtail       4522.911   3357.149   1.347    0.178
## Initial.mass:SexMale              -311.787   1934.972  -0.161    0.872
## Initial.mass:SexN/A              17865.539  17331.915   1.031    0.303
## Initial.mass:MaturityJuvenile   -18464.138  17193.857  -1.074    0.283
## DietFly:SexMale                      2.799      4.156   0.673    0.501
## DietSpringtail:SexMale               3.985      3.013   1.323    0.186
## DietFly:SexN/A                       5.054      5.261   0.961    0.337
## DietSpringtail:SexN/A                   NA         NA      NA       NA
## DietFly:MaturityJuvenile                NA         NA      NA       NA
## DietSpringtail:MaturityJuvenile         NA         NA      NA       NA
## SexMale:MaturityJuvenile                NA         NA      NA       NA
## SexN/A:MaturityJuvenile                 NA         NA      NA       NA
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 8.5977  on 20  degrees of freedom
## Residual deviance: 1.2764  on  6  degrees of freedom
## AIC: 93.16
##
## Number of Fisher Scoring iterations: 4
```

```
anova(mort)
```

```
## Analysis of Deviance Table
##
## Model: poisson, link: log
##
## Response: Mortality
##
## Terms added sequentially (first to last)
##
##
##                        Df Deviance Resid. Df Resid. Dev
## NULL                                     20     8.5977
## Initial.mass            1  0.20426        19     8.3934
## Diet                    2  1.85725        17     6.5362
## Sex                     2  0.43902        15     6.0971
## Maturity                1  1.00816        14     5.0890
## Initial.mass:Diet       2  0.43240        12     4.6566
## Initial.mass:Sex        2  0.06704        10     4.5895
## Initial.mass:Maturity   1  1.17287         9     3.4167
## Diet:Sex                3  2.14028         6     1.2764
## Diet:Maturity           0  0.00000         6     1.2764
## Sex:Maturity            0  0.00000         6     1.2764
```