**Text S1**: R code to load the data and reproduce the hierarchical modeling on a set of networks.

#This code implements the analysis presented in

#"Understanding linkage rules in plant-pollinator networks

#by using hierarchical models that incorporate pollinator

#detectability and plant traits" by Ignasi Bartomeus

##Load the R libraries and the data required##

library(reshape)

library(bipartite)

library(unmarked)

#Data can be downloaded from the following webpage:

#[*http://figshare.com/articles/Plant\_Pollinator\_Network\_Data/154863*](http://figshare.com/articles/Plant_Pollinator_Network_Data/154863)

d1 <- read.table("http://files.figshare.com/479751/Data\_ntw\_all.txt", h = TRUE)

#see the structure of the data

str(d1)

#extract plant trait variables from the datsset

traits <- unique(d1[,c(1,2,6,7,8,9,10,17)])

##Create some objects that will be needed for the analysis

#create a list of all Sites

site\_list <- unique(d1$Site)

#create empty lists and data frames to store the main reults

est\_ntws <- list()

ntws <- list()

Params <- data.frame(Site = NA, pol= NA, occup= NA, detect= NA, covar= NA, visits= NA, predCA = NA, predOP = NA, d\_pol\_raw = NA, d\_pol\_est = NA)

bipart <- data.frame(Site = NA, ntw = NA, pol = NA, plant= NA , conn = NA, links = NA, N = NA, NODF = NA, R\_plants = NA, R\_pol = NA, H2 = NA)

#create two hypothetical invasive species based on the characteristics of the real invasive species

#using SEL2OP Opuntia traits

traitsTemp <- subset(traits, Site == site\_list[12])

new\_spOP <- traitsTemp[which(traitsTemp$Plant\_sp == "Opuntia\_stricta"),]

#using MED2 Carpobrotus traits

traitsTemp <- subset(traits, Site == site\_list[6])

new\_spCA <- traitsTemp [which(traitsTemp $Plant\_sp == "Carpobrotus\_aff.acinaciformis"),]

##Creat a loop through all sites, and within each site, a second loop through all pollinators

#loop through the 12 sites

for (s in 1:12){

#select site data

d2 <- subset(d1, Site == site\_list[s])

#construct the network

nw <- cast(d2, Plant\_sp ~ Insect\_sp, value = "Freq", fun = sum)

rownames(nw) <- nw[,1]

nw <- nw[-1]

#refactor the data

d2$Round <- as.factor(d2$Round)

d2$Plant\_sp <- factor(d2$Plant\_sp)

#make a vector with all pollinators

pol\_list <- levels(factor(d2$Insect\_sp))

#create an empty matrix to store the estimated network and the parameters:

estimated\_Ntw <- matrix(ncol = length(pol\_list), nrow = length(unique(d2$Plant\_sp)), dimnames = dimnames(nw))

Pol\_parameters <- data.frame(pol = pol\_list, occup = c(1:length(pol\_list)), detect = c(1:length(pol\_list)), covar = c(1:length(pol\_list)), visits = c(1:length(pol\_list)), predCA = c(1:length(pol\_list)), predOP = c(1:length(pol\_list)))

#loop thought pollinators

for (i in 1:length(pol\_list)){

#select each species

d1p1 <- subset(d2, Insect\_sp == pol\_list[i])

#Create the occupancy matrix (visits to each plant per round)

y <- cast(d1p1, Plant\_sp ~ Round, value = "Freq", fun = sum, add.missing= TRUE)

y <- as.matrix(y[1:length(y[,1]),])

#Exclude Singletones from the analysis

if(sum(y) < 2){

#add NA in the Output data frame, as those are not analyzed

Pol\_parameters[i,2] <- NA

Pol\_parameters[i,3] <- NA

Pol\_parameters[i,4] <- NA

Pol\_parameters[i,5] <- sum(y)

Pol\_parameters[i,6] <- NA

Pol\_parameters[i,7] <- NA

estimated\_Ntw[,i] <- nw[,i]

}

else {

#prepare the data for modeling

pl <- rownames(y)

traits2 <- subset(traits, Site == site\_list[s])

covs <- traits2[which(traits2$Plant\_sp %in% pl),]

#Format for unmarked and summarize data

umf <- unmarkedFramePCount(y=y, siteCovs= covs)

summary(umf)

# Fit the models

# Detection covariates follow first tilde, then come abundance covariates

#First all possible family and abundance models

lam <- pcount(~1 ~ 1, data=umf)

lam\_Family <- pcount(~1 ~ P\_Family, data=umf,se = FALSE)

lam\_meanflw <- pcount(~1 ~ mean\_flw, data=umf,se = FALSE)

lam\_Family\_meanflw <- pcount(~1 ~ P\_Family + mean\_flw, data=umf, se = FALSE)

#Then all possible traits models

#one by one

lam\_Morphology <- pcount(~1 ~ P\_Morphology, data=umf,se = FALSE)

lam\_Inflorescence <- pcount(~1 ~ P\_Inflorescence, data=umf,se = FALSE)

lam\_Shape <- pcount(~1 ~ P\_Shape, data=umf,se = FALSE)

lam\_Color <- pcount(~1 ~ P\_Color, data=umf,se = FALSE)

#all traits

lam\_Morphology\_Inflorescence\_Shape\_Color <- pcount(~1 ~ P\_Morphology + P\_Inflorescence + P\_Shape + P\_Color, data=umf,se = FALSE)

#Combinations of 3 Traits

lam\_Morphology\_Inflorescence\_Shape <- pcount(~1 ~ P\_Morphology + P\_Inflorescence + P\_Shape, data=umf,se = FALSE)

lam\_Morphology\_Inflorescence\_Color <- pcount(~1 ~ P\_Morphology + P\_Inflorescence + P\_Color, data=umf,se = FALSE)

lam\_Morphology\_Shape\_Color <- pcount(~1 ~ P\_Morphology + P\_Shape + P\_Color, data=umf,se = FALSE)

lam\_Inflorescence\_Shape\_Color <- pcount(~1 ~ P\_Inflorescence + P\_Shape + P\_Color, data=umf,se = FALSE)

#Combinations of 2 Traits

lam\_Morphology\_Inflorescence <- pcount(~1 ~ P\_Morphology + P\_Inflorescence, data=umf,se = FALSE)

lam\_Shape\_Color <- pcount(~1 ~ P\_Shape + P\_Color, data=umf,se = FALSE)

lam\_Morphology\_Color <- pcount(~1 ~ P\_Morphology + P\_Color, data=umf,se = FALSE)

lam\_Inflorescence\_Color <- pcount(~1 ~ P\_Inflorescence + P\_Color, data=umf,se = FALSE)

lam\_Morphology\_Shape <- pcount(~1 ~ P\_Morphology + P\_Shape, data=umf,se = FALSE)

lam\_Inflorescence\_Shape <- pcount(~1 ~ P\_Inflorescence + P\_Shape, data=umf,se = FALSE)

#Combination of each traits and abundance

lam\_Morphology\_meanflw <- pcount(~1 ~ P\_Morphology + mean\_flw, data=umf, se = FALSE)

lam\_Inflorescence\_meanflw <- pcount(~1 ~ P\_Inflorescence + mean\_flw, data=umf,se = FALSE)

lam\_Shape\_meanflw <- pcount(~1 ~ P\_Shape + mean\_flw, data=umf, se = FALSE)

lam\_Color\_meanflw <- pcount(~1 ~ P\_Color + mean\_flw, data=umf, se = FALSE)

#Combination of 2 Traits and abundance

lam\_Morphology\_Inflorescence\_meanflw <- pcount(~1 ~ P\_Morphology + P\_Inflorescence + mean\_flw, data=umf,se = FALSE)

lam\_Shape\_Color\_meanflw <- pcount(~1 ~ P\_Shape + P\_Color + mean\_flw, data=umf,se = FALSE)

lam\_Morphology\_Color\_meanflw <- pcount(~1 ~ P\_Morphology + P\_Color + mean\_flw, data=umf,se = FALSE)

lam\_Inflorescence\_Color\_meanflw <- pcount(~1 ~ P\_Inflorescence + P\_Color + mean\_flw, data=umf,se = FALSE)

lam\_Morphology\_Shape\_meanflw <- pcount(~1 ~ P\_Morphology + P\_Shape + mean\_flw, data=umf,se = FALSE)

lam\_Inflorescence\_Shape\_meanflw <- pcount(~1 ~ P\_Inflorescence + P\_Shape + mean\_flw, data=umf,se = FALSE)

#Combination of 3 Traits and abundance

lam\_Morphology\_Inflorescence\_Shape\_meanflw <- pcount(~1 ~ P\_Morphology + P\_Inflorescence + P\_Shape + mean\_flw, data=umf,se = FALSE)

lam\_Morphology\_Inflorescence\_Color\_meanflw <- pcount(~1 ~ P\_Morphology + P\_Inflorescence + P\_Color + mean\_flw, data=umf,se = FALSE)

lam\_Morphology\_Shape\_Color\_meanflw <- pcount(~1 ~ P\_Morphology + P\_Shape + P\_Color + mean\_flw, data=umf,se = FALSE)

lam\_Inflorescence\_Shape\_Color\_meanflw <- pcount(~1 ~ P\_Inflorescence + P\_Shape + P\_Color + mean\_flw, data=umf,se = FALSE)

#All traits and abundance

lam\_Morphology\_Inflorescence\_Shape\_Color\_meanflw <- pcount(~1 ~ P\_Morphology + P\_Inflorescence + P\_Shape + P\_Color + mean\_flw, data=umf,se = FALSE)

#create a list of all candidate models

ms<- fitList(lam, lam\_Family, lam\_meanflw, lam\_Family\_meanflw, lam\_Morphology, lam\_Inflorescence, lam\_Shape, lam\_Color, lam\_Morphology\_Inflorescence\_Shape\_Color, lam\_Morphology\_Inflorescence\_Shape, lam\_Morphology\_Inflorescence\_Color, lam\_Morphology\_Shape\_Color, lam\_Inflorescence\_Shape\_Color, lam\_Morphology\_Inflorescence, lam\_Shape\_Color, lam\_Morphology\_Color, lam\_Inflorescence\_Color, lam\_Morphology\_Shape, lam\_Inflorescence\_Shape, lam\_Morphology\_meanflw, lam\_Inflorescence\_meanflw, lam\_Shape\_meanflw, lam\_Color\_meanflw, lam\_Morphology\_Inflorescence\_meanflw, lam\_Shape\_Color\_meanflw, lam\_Morphology\_Color\_meanflw, lam\_Inflorescence\_Color\_meanflw, lam\_Morphology\_Shape\_meanflw, lam\_Inflorescence\_Shape\_meanflw, lam\_Morphology\_Inflorescence\_Shape\_meanflw, lam\_Morphology\_Inflorescence\_Color\_meanflw, lam\_Morphology\_Shape\_Color\_meanflw, lam\_Inflorescence\_Shape\_Color\_meanflw, lam\_Morphology\_Inflorescence\_Shape\_Color\_meanflw)

# Rank them by AIC

ms1 <- modSel(ms)

#Select best model

model <- ms1@Full$model[1]

#And extract estimates

m <- ms@fits[model][[1]]

beta1 <- coef(m)

# Note, estimates of detection coefficients are on the logit-scale

#store results on the Output data frame

Pol\_parameters[i,2] <- plogis(beta1)["lam(Int)"]

Pol\_parameters[i,3] <- plogis(beta1)["p(Int)"]

Pol\_parameters[i,4] <- ms1@Full$model[1]

Pol\_parameters[i,5] <- sum(y)

#calculate estimated visitation frequency and store it in the extimated network

estimated\_Ntw[,i] <- round(predict(m, type="state", newdata=covs)$Predicted)

#predict visitation to the new species

Pol\_parameters[i,6] <- round(predict(m, type="state", newdata=new\_spCA)$Predicted)

Pol\_parameters[i,7] <- round(predict(m, type="state", newdata=new\_spOP)$Predicted)

}

}

##store results at the community level

#get the complete raw network

nw2 <- as.matrix(nw)

dimnames(nw2) = dimnames(nw)

#store it in the list ntws

ntws[[s]] <- nw2

#fill site name

Pol\_parameters$Site <- rep(site\_list[s], length(Pol\_parameters[,1]))

sp <- specieslevel(nw2)

#Add singletones back

for(a in 1:length(estimated\_Ntw[1,])){

if (sum(estimated\_Ntw[,a]) < 1){

estimated\_Ntw[,a] <- nw2[,a]

}

}

#store network metrics using bipartite package fucntions specieslevel

spe <- specieslevel(estimated\_Ntw)

#number of pollinators raw community

d\_pol\_raw <- as.data.frame(sp$`higher trophic level`$d, optional = TRUE)

colnames(d\_pol\_raw) <- c("d\_pol\_raw")

Pol\_parameters$d\_pol\_raw <- d\_pol\_raw$d\_pol\_raw

#number of pollinators estimated community

d\_pol\_est <- as.data.frame(spe$`higher trophic level`$d, optional = TRUE)

colnames(d\_pol\_est) <- c("d\_pol\_est")

Pol\_parameters$d\_pol\_est <- d\_pol\_est$d\_pol\_est

#store the full estimated community

est\_ntws[[s]] <- estimated\_Ntw

#Add the parameters to the Output

Params <- rbind(Params,Pol\_parameters)

str(Pol\_parameters)

#Calculate parameters using networklevel function on bipartite package

n <- networklevel(nw2)

ne <- networklevel(estimated\_Ntw)

#store them

bipart\_params\_n <- data.frame(Site = rep(site\_list[s], 1), ntw = rep("original", 1), pol = n$`number of higher trophic species`, plant= n$`number of lower trophic species`, conn = n$connectance, links = n$`links per species`, N = n$nestedness, NODF = n$`weighted NODF`, R\_plants = n$`robustness lower exterminated`, R\_pol = n$`robustness higher exterminated`, H2 = n$H2)

bipart\_params\_ne <- data.frame(Site = rep(site\_list[s], 1), ntw = rep("estimated", 1), pol = ne$`number of higher trophic species`, plant= ne$`number of lower trophic species`, conn = ne$connectance, links = ne$`links per species`, N = ne$nestedness, NODF = ne$`weighted NODF`, R\_plants = ne$`robustness lower exterminated`, R\_pol = ne$`robustness higher exterminated`, H2 = ne$H2)

bipart <- rbind(bipart,bipart\_params\_n, bipart\_params\_ne)

#As is a slow loop, we can print the network number (s) to keep track

print(s)

}

#clean the Output

Params <- Params[-1,]

bipart <- bipart[-1,]

#See the Output:

est\_ntws #list of estimated Networks; access to the first one by est\_ntw[[1]]

ntws #idem for observed networks

Params #data frame of the parameters of all models

bipart #data frame of the network metrics for each network