Analyse de la structure sociale du grand dauphin en Méditerranée nord-occidentale

Olivier Gimenez

August 12, 2016

## Introduction

Dans ce document, je fais une analyse des données de photo-identification du grand dauphin récoltées dans le cadre du projet [GDEGeM](http://www.gdegem.org/). Je me concentre ici sur les Golfe du Lion et Provence, et j'utilise le fichier Matrice individus\_Prov + GDL.xlsx envoyé par email par Hélène (Labach) début novembre 2015.

## Préparation des données

On lit la feuille Droite-Gauche (7ème feuille) dans le fichier Matrice individus\_Prov + GDL.xlsx. Au préalable, sous Excel, on supprime les colonnes qui contiennent les identifiants des différentes structures (je sais, pas très "reproducible research"...). On manipule ce fichier brut pour en extraire les ids, les dates et la matrice de détections/non-détections.

# lecture des données  
library(gdata)  
#gecem = read.xls("Matrice individus\_Prov + GDL.xlsx", sheet = 7, header = F,colClasses = "character")  
#save(gecem,file='dat.RData')  
setwd('/Users/gimenez/Dropbox/OG/BOULOT/GENS/Hélène LABACH/Gdegem/gedegem\_la\_suite')  
load('dat.RData')  
gecem <- gecem[,1:290]  
#head(gecem)

On récupère les dates de sortie

dates\_temp <- gecem[1,]  
dates <- dates\_temp[,1]  
for (i in 2:dim(dates\_temp)[2]){  
 dates <- c(dates,dates\_temp[,i])  
}  
dates <- as.Date(dates,"%Y%m%d")  
#dates  
# on vérifie si certaines dates se retrouvent plusieurs fois  
duplicated(dates)

## [1] FALSE FALSE FALSE FALSE FALSE TRUE FALSE FALSE FALSE FALSE FALSE  
## [12] FALSE FALSE TRUE FALSE TRUE TRUE FALSE TRUE FALSE FALSE FALSE  
## [23] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  
## [34] TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  
## [45] FALSE TRUE FALSE FALSE FALSE TRUE FALSE FALSE FALSE FALSE FALSE  
## [56] FALSE FALSE TRUE FALSE FALSE TRUE FALSE FALSE FALSE FALSE FALSE  
## [67] TRUE FALSE FALSE FALSE TRUE FALSE FALSE FALSE FALSE FALSE FALSE  
## [78] FALSE FALSE FALSE FALSE FALSE FALSE TRUE FALSE FALSE TRUE FALSE  
## [89] FALSE FALSE FALSE FALSE FALSE TRUE FALSE TRUE FALSE TRUE FALSE  
## [100] FALSE TRUE FALSE FALSE FALSE FALSE TRUE FALSE FALSE FALSE FALSE  
## [111] FALSE FALSE FALSE FALSE FALSE TRUE FALSE TRUE FALSE FALSE FALSE  
## [122] FALSE TRUE FALSE FALSE FALSE FALSE TRUE FALSE TRUE TRUE FALSE  
## [133] FALSE FALSE FALSE TRUE FALSE TRUE FALSE FALSE FALSE TRUE FALSE  
## [144] FALSE FALSE TRUE FALSE FALSE FALSE TRUE FALSE FALSE FALSE TRUE  
## [155] FALSE TRUE FALSE TRUE TRUE FALSE FALSE TRUE FALSE FALSE FALSE  
## [166] TRUE FALSE FALSE FALSE FALSE FALSE TRUE FALSE FALSE FALSE TRUE  
## [177] FALSE FALSE FALSE TRUE FALSE FALSE TRUE TRUE FALSE TRUE FALSE  
## [188] FALSE TRUE FALSE FALSE FALSE TRUE TRUE FALSE TRUE TRUE FALSE  
## [199] TRUE TRUE FALSE TRUE FALSE FALSE TRUE FALSE FALSE FALSE FALSE  
## [210] FALSE TRUE FALSE FALSE FALSE FALSE TRUE TRUE FALSE FALSE FALSE  
## [221] FALSE FALSE TRUE FALSE TRUE FALSE TRUE FALSE FALSE FALSE TRUE  
## [232] FALSE TRUE FALSE FALSE TRUE FALSE TRUE FALSE TRUE FALSE FALSE  
## [243] TRUE TRUE TRUE FALSE FALSE FALSE FALSE TRUE FALSE TRUE FALSE  
## [254] TRUE FALSE TRUE TRUE FALSE FALSE TRUE FALSE TRUE FALSE TRUE  
## [265] TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  
## [276] FALSE TRUE FALSE FALSE TRUE FALSE FALSE FALSE TRUE FALSE FALSE  
## [287] FALSE FALSE FALSE FALSE

Dans le vecteur rendu par R, on a plusieurs 'TRUE' qui indique une date dupliquée. Je m'occuperai de ce problème plus bas.

On construit la matrice des détections/non-détections

gecem <- gecem[-1,]  
M <- gecem  
M <- replace(M, M=="",0) # no capture   
M <- replace(M, M=="g",1) # L   
M <- replace(M, M=="d",1) # R   
M <- replace(M, M=="dg", 1) # S   
tp <- dim(M)  
M <- as.matrix(M)  
M <- matrix(as.numeric(M),nrow=tp[1],ncol=tp[2],byrow=F)   
# il y avait des D, G et DG en majuscules,   
# que j'ai passés en minuscules   
# M  
# Les dimensions de M sont :  
tp

## [1] 928 290

On a donc 928 lignes, soit 928 individus identifiés individuellement, au cours de 290 occasions.

A ce stade, on voudrait appliquer un filtre en ne sélectionnant que les obs bien et moyennement marqués. On prendra un score inférieur ou égal 4, avec 2 et 2 au moins pour quality et distinctiveness (1 = meilleur, 3 = moins bon). Il nous faudra penser à garder en mémoire la proportion pour appliquer une correction afin d'obtenir la pop totale. Après discussion avec Hélène, le filtre s'avère un peu trop limitant. On refait les analyses en gardant l'observation si droite ou gauche est à 1 ou 2.

On commence par créer les 4 matrices qui regroupent les informations sur quality et distinctiveness

#filtreQD = read.xls("Matrice individus\_Prov + GDL.xlsx", sheet = 8, header = F,colClasses = "character")  
#filtreQG = read.xls("Matrice individus\_Prov + GDL.xlsx", sheet = 9, header = F,colClasses = "character")  
#filtreDD = read.xls("Matrice individus\_Prov + GDL.xlsx", sheet = 10, header = F,colClasses = "character")  
#filtreDG = read.xls("Matrice individus\_Prov + GDL.xlsx", sheet = 11, header = F,colClasses = "character")  
#save(filtreQD,filtreQG,filtreDD,filtreDG,file='filtre.RData')  
load('filtre.RData')  
  
# quality droite  
filtre <- filtreQD[-1,]  
filtre <- filtre[,-1]  
filtre <- replace(filtre, filtre=="",0) # no capture   
dimfiltre <- dim(filtre)  
filtre <- as.matrix(filtre)  
filtre <- matrix(as.numeric(filtre),nrow=dimfiltre[1],ncol=dimfiltre[2],byrow=F)  
filtre[is.na(filtre)]<-0 # il y a des caractères bizarres qui se sont introduits, dans des cellules vides, je les remplace par des 0 (après avoir fait qqs vérif)  
QD <- filtre  
  
# quality gauche  
filtre <- filtreQG[-1,]  
filtre <- filtre[,-1]  
filtre <- replace(filtre, filtre=="",0) # no capture   
dimfiltre <- dim(filtre)  
filtre <- as.matrix(filtre)  
filtre <- matrix(as.numeric(filtre),nrow=dimfiltre[1],ncol=dimfiltre[2],byrow=F)  
filtre[is.na(filtre)]<-0 # il y a des caractères bizarres qui se sont introduits, dans des cellules vides, je les remplace par des 0 (après avoir fait qqs vérif)  
QG <- filtre  
  
# distinctiveness droite  
filtre <- filtreDD[-1,]  
filtre <- filtre[,-1]  
filtre <- replace(filtre, filtre=="",0) # no capture   
dimfiltre <- dim(filtre)  
filtre <- as.matrix(filtre)  
filtre <- matrix(as.numeric(filtre),nrow=dimfiltre[1],ncol=dimfiltre[2],byrow=F)  
filtre[is.na(filtre)]<-0 # il y a des caractères bizarres qui se sont introduits, dans des cellules vides, je les remplace par des 0 (après avoir fait qqs vérif)  
DD <- filtre  
  
# distinctiveness gauche  
filtre <- filtreDG[-1,]  
filtre <- filtre[,-1]  
filtre <- replace(filtre, filtre=="",0) # no capture   
dimfiltre <- dim(filtre)  
filtre <- as.matrix(filtre)  
filtre <- matrix(as.numeric(filtre),nrow=dimfiltre[1],ncol=dimfiltre[2],byrow=F)  
filtre[is.na(filtre)]<-0 # il y a des caractères bizarres qui se sont introduits, dans des cellules vides, je les remplace par des 0 (après avoir fait qqs vérif)  
DG <- filtre

On applique le filtre défini ci-dessus, et on crée deux matrices qui sont la somme des gauche et droite.

QG[QG == 3] = 0  
QD[QD == 3] = 0  
DG[DG == 3] = 0  
DD[DD == 3] = 0  
qualite <- QG + QD  
distinc <- DG + DD

On réunit alors les deux matrices pour avoir la note globale. Puis on crée la nouvelle matrice d'observations :

selection <- qualite + distinc  
Mselect <- selection  
Mselect[Mselect>=1]=1  
dim(Mselect)

## [1] 928 290

Parmi ces 290 occasions, on retrouve plusieurs dates dupliquées :

dates[duplicated(dates)]

## [1] "2013-05-27" "2013-06-04" "2013-06-05" "2013-06-05" "2013-06-06"  
## [6] "2013-07-18" "2013-09-08" "2013-09-14" "2013-10-07" "2013-10-09"  
## [11] "2013-10-15" "2013-10-18" "2014-01-12" "2014-02-13" "2014-03-07"  
## [16] "2014-03-08" "2014-03-10" "2014-03-13" "2014-03-18" "2014-05-07"  
## [21] "2014-05-17" "2014-06-04" "2014-06-11" "2014-06-12" "2014-06-12"  
## [26] "2014-06-16" "2014-06-17" "2014-06-20" "2014-06-28" "2014-07-16"  
## [31] "2014-07-24" "2014-07-25" "2014-08-01" "2014-08-01" "2014-08-04"  
## [36] "2014-08-07" "2014-08-12" "2014-08-18" "2014-08-21" "2014-08-25"  
## [41] "2014-08-25" "2014-08-26" "2014-08-28" "2014-09-08" "2014-09-08"  
## [46] "2014-09-09" "2014-09-09" "2014-09-10" "2014-09-10" "2014-09-11"  
## [51] "2014-09-13" "2014-09-21" "2014-10-20" "2014-10-20" "2014-10-25"  
## [56] "2014-10-26" "2014-10-27" "2014-10-30" "2014-10-31" "2014-11-16"  
## [61] "2014-11-19" "2014-12-04" "2014-12-22" "2014-12-22" "2014-12-22"  
## [66] "2015-02-12" "2015-02-13" "2015-02-14" "2015-02-15" "2015-02-15"  
## [71] "2015-02-18" "2015-02-19" "2015-02-20" "2015-02-20" "2015-03-06"  
## [76] "2015-03-08" "2015-03-14"

Je regroupe ces observations sous une seule occasion dans M (en vérifiant bien qu'un même individu n'est pas vu à ces deux occasions), et je supprime le duplicat dans le vecteur des dates ; je fais ça autant de fois qu'il y a de duplicats :

Mselectdf <- data.frame(dates=dates,Mselect=t(Mselect))  
library(plyr)  
Mselect2 <- ddply(Mselectdf,"dates",numcolwise(max))  
Mselect <- t(Mselect2)  
Mselect <- Mselect[-1,]  
dim(Mselect)

## [1] 928 213

Mselect <- matrix(as.numeric(Mselect),nrow=dim(Mselect)[1],ncol=dim(Mselect)[2],byrow=F)

On fait subir le même traitement à M :

Mdf <- data.frame(dates=dates,M=t(M))  
library(plyr)  
M2 <- ddply(Mdf,"dates",numcolwise(max))  
M <- t(M2)  
M <- M[-1,]  
dim(M)

## [1] 928 213

indic <- (1:tp[2])[duplicated(dates)]  
dates <- dates[-indic]  
length(dates)

## [1] 213

tp <- dim(M)  
tp

## [1] 928 213

M <- matrix(as.numeric(M),nrow=tp[1],ncol=tp[2],byrow=F)

Calcule le nombre d'individus vus au moins 1 fois, 2 fois, 3 fois etc...

sum(apply(Mselect,1,sum)>0)

## [1] 635

sum(apply(Mselect,1,sum)>1)

## [1] 181

sum(apply(Mselect,1,sum)>2)

## [1] 67

sum(apply(Mselect,1,sum)>3)

## [1] 28

sum(apply(M,1,sum)>0)

## [1] 928

sum(apply(M,1,sum)>1)

## [1] 306

sum(apply(M,1,sum)>2)

## [1] 159

sum(apply(M,1,sum)>3)

## [1] 73

Ca fait une belle difference quand on ne regroupe pas les dates de sortie.

Il nous faut maintenant assigner ces individus à des groupes. Pour ce faire, on utilise la feuille des observations :

library(gdata)  
#obs\_gecem = read.xls("Matrice individus\_Prov + GDL.xlsx", sheet = 2, header = F,colClasses = "character")  
#save(obs\_gecem,file='obs.RData')  
load('obs.RData')

On se débarrasse des 3 dernières colonnes qui donnent des stats descriptives diverses :

obs\_gecem <- obs\_gecem[,2:290]  
obs\_gecem <- obs\_gecem[-1,] # on supprime les dates  
obs\_gecem <- replace(obs\_gecem, obs\_gecem =="",0) # no capture   
dim(obs\_gecem)

## [1] 930 289

On ne prend que les individus vus au moins 3 fois () :

obs\_gecem2 <- obs\_gecem[(apply(M,1,sum)>2),]  
dim(obs\_gecem2)

## [1] 161 289

# recupere le nom des groupes  
groups <- levels(factor(unlist(obs\_gecem2)))[-1]  
#groups

## Analyse de la structure sociale

Je cherche à créer un tableau avec en lignes ces groupes, et en colonne les individus puis dans les cellules des 1 quand l'individu est dans le groupe, et des 0 sinon :

nbindiv <- dim(obs\_gecem2)[1]  
nbgroups <- length(groups)  
network\_dolphin <- matrix(0,nrow=nbgroups,ncol=nbindiv)  
vect\_ind <- 1:nbindiv  
for (i in 1: nbgroups){  
 mask <- (obs\_gecem2==groups[i]) # ou sont les individus du groupe courant  
 mask2 <- apply(mask,1,sum)!=0 # quels sont les individus du groupe courant  
 interaction <- vect\_ind[mask2]  
 network\_dolphin[i,interaction] <- 1 # met un 1 pour indiquer l'interaction   
}  
  
colnames(network\_dolphin) <- rownames(obs\_gecem2)  
rownames(network\_dolphin) <- groups  
  
str(network\_dolphin)

## num [1:419, 1:161] 0 0 0 0 0 0 1 1 0 0 ...  
## - attr(\*, "dimnames")=List of 2  
## ..$ : chr [1:419] "130508\_BR\_SSO001" "130527\_GC\_STP001" "130527\_GC\_STP002" "130528\_GC\_STP003" ...  
## ..$ : chr [1:161] "2" "3" "7" "8" ...

Lit les données et charge le package qui va bien :

library(asnipe)  
#data\_pop\_struct <- as.matrix(data\_pop\_struct)  
dim(network\_dolphin)

## [1] 419 161

#dimnames(network\_dolphin)  
network<-get\_network(network\_dolphin,data\_format= "GBI", association\_index="HWI")

## Generating 161 x 161 matrix

Construit le réseau :

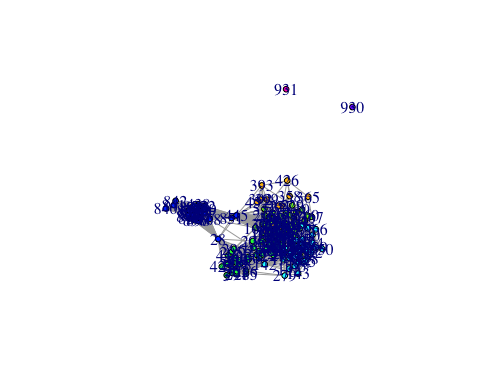
library(igraph)  
net<-graph.adjacency(network,mode='undirected',diag=FALSE,weighted=TRUE)

On clusterise. Attention, on a 7 groupes avec les 161 individus vs. les 4 groupes trouvés avec seulement 68 individus :

fc <- cluster\_fast\_greedy(net)  
comps = fc$membership   
colbar <- rainbow(max(comps)+1) # on rajoute les couleurs de l'arc-en-ciel - moment poésie...  
V(net)$color <- colbar[comps+1]  
  
#comps[comps==1]<-'red'  
#comps[comps==2]<-'steelblue1'  
#comps[comps==3]<-'green'  
#comps[comps==4]<-'yellow'  
#V(net)$color <- comps

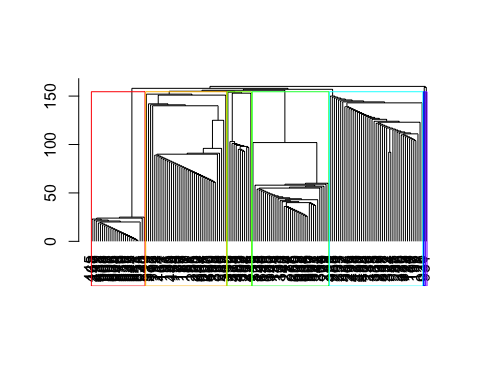
Tadannnn, le réseau social, enfin :

plot(net,vertex.size=6)



Moins glamour, le dendrogramme (vérifier que les couleurs correspondent à celle utilisées pour représenter graphiquement le réseau social) :

plot\_dendrogram(fc,mode = "hclust",colbar=colbar)



On teste si les individus s'associe préférentiellement de manière significative :

library(sna)   
library(asnipe)  
# Calculate network   
networks <- get\_network(network\_dolphin, data\_format="GBI", association\_index="HWI")

## Generating 161 x 161 matrix

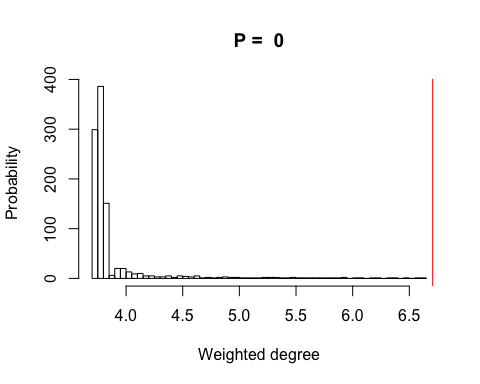
deg\_weighted <- degree(networks,gmode="graph", ignore.eval=FALSE)  
  
# Perform the permutations  
network\_perm <- network\_permutation(network\_dolphin, data\_format="GBI", association\_matrix=networks, permutations = 1000)

## Starting permutations, generating 1000 x 161 x 161 matrix

dim(network\_perm)

## [1] 1000 161 161

# Calculate the weighted degree for each permutation  
deg\_weighted\_perm <- degree(network\_perm,gmode="graph", g=c(1:1000), ignore.eval=FALSE)  
  
# Plot the distribution of permutations with the original data overlaid  
hist(colMeans(deg\_weighted\_perm),breaks=100,  
 main=paste("P = ",sum(mean(deg\_weighted) < colMeans(deg\_weighted\_perm))/ncol(deg\_weighted\_perm)),  
 xlab="Weighted degree", ylab="Probability")  
abline(v=mean(deg\_weighted), col="red")



Si on utilise SD(HWI) à la place de HWI :

networks <- get\_network(network\_dolphin, data\_format="GBI", association\_index="HWI")

## Generating 161 x 161 matrix

networks\_sd <- sd(networks)  
  
network\_perm <- network\_permutation(network\_dolphin, data\_format="GBI", association\_matrix=networks, permutations = 1000)

## Starting permutations, generating 1000 x 161 x 161 matrix

nb.permut <- dim(network\_perm)[1]  
network\_perm\_sd <- rep(NA, nb.permut)  
for(i in 1 : nb.permut){network\_perm\_sd[i] <- sd(network\_perm[i,,])}  
  
hist(network\_perm\_sd,breaks=100,main=paste("P = ",sum(networks\_sd < network\_perm\_sd) / nb.permut),xlab="SD(HWI)", ylab="Probability")  
abline(v=networks\_sd, col="red")

