

```
# COURS
rm(list=ls()) # on élimine les données stockées

# Lecture de données
# Le dossier de travail (working directory) est supposé défini ou on le fait avec setwd() ou le menu RStudio/Session/Set Working Directory
ecobrut=read.table('carabides_32sta_12eco.txt', h=T, sep="\t", row.names="Stations")
attach(ecobrut) # les variables du tableau sont accessibles directement (sans ecobrut$ )

# Transformation de certaines variables
# Humidité est OK
# pHeau et pHKCL => racine()
sqrt_pHeau=sqrt(pHeau) ; sqrt_pHKCL=sqrt(pHKCL) ;
# Ca, K, Mg, Na, P => log base 2 ou nppOrien
Log_Ca =log(Ca+1, base=2); Log_K =log(K+1, base=2); Log_Mg =log(Mg+1, base=2); Log_Na =log(Na +1, base=2); Log_P =log(P+1, base=2);

# Reconstruction d'un tableau (dataframe) à partir de variables
eco = data.frame(Humidite, sqrt_pHeau, sqrt_pHKCL,
Log_Ca, Log_K, Log_Mg, Log_Na, Log_P,
row.names=row.names(ecobrut)) ; # = code des stations de la nouvelle table ;

# Visualisation des relations entre les variables
pairs(eco)

# Matrice de corrélation
correlation = cor(eco)
edit(correlation)

# On réalise les codes r qui suivent
fichier = eco # pour réaliser les codes r qui suivent
label_objets = row.names(fichier) # pour stocker les labels des objets dans une variable

# Appel de l'ACP
library(vegan) # chargement de vegan
acp_resu <- rda(fichier, scale=TRUE) # rda() -> acp (un fichier) ou rda (plusieurs fichiers);
# scale=TRUE => matrice de corrélation;
summary(acp_resu) # résumé des résultats;

# Histogramme de la distribution de la variance expliquée par les axes
# Distribution des valeurs propres
eigenvalues = acp_resu$CA$eig / sum(acp_resu$CA$eig)*100

pca1 = paste('PCA 1 : ', round(eigenvalues['PC1'],2), '%', sep = "")
pca2 = paste('PCA 2 : ', round(eigenvalues['PC2'],2), '%', sep = "")

# Dessiner les valeurs propres et le % de variance de chaque axe
barplot(eigenvalues, main="Valeurs propres", col="bisque", las=2)
abline(h=mean(eigenvalues), col="red") # moyenne des valeurs propres
legend("topright",
"Moynne des valeurs propres",
lwd=1,
col=2,
bty="n")

# Représentation du plan factoriel

coordo <-scores(acp_resu) # On récupère les coordonnées
coordovar = data.frame(coordo$species)
coordoobj = data.frame(coordo$sites)

plot(coordoobj$PC1, # On choisit les axes
coordoobj$PC2,
type="n", # n= graphique vide
main="ACP", # Titre principal

xlab = pca1,
ylab = pca2,
xlim=c(-1.5, 2), # Range de l'axe X
ylim=c(-1.5, 1.5) # Range de l'axe
)

coordoobj$pos <- ifelse(coordoobj$PC1 < 0 , 2, 4) # position du label autour du point

points(coordoobj$PC1, # On dessine des symboles pour les sites
coordoobj$PC2,
col='blue', # couleur => peut être un vecteur de couleurs (groupes)
pch=15) # type de symboles => peut être un vecteur de symboles (groupes)

text(coordoobj$PC1, # On écrit les labels pour les sites
coordoobj$PC2,
col='blue', # couleur => peut être un vecteur de couleurs (groupes)
pos = coordoobj$pos, # position du symbole (dessous=1, gauche=2, dessous=3, droite=4)
labels = label_objets) # variable avec les noms des sites

abline(h=0, lty=3) # axe horizontal (lty = type de la ligne)
abline(v=0, lty=3) # axe vertical (lty = type de la ligne)

# Cercle des corrélations
coordo <-scores(acp_resu, choices=c(1,2,3,4)) # choices = choisir les axes
correlation = data.frame(cor(fichier, coordo$sites)) # appel au fichier de départ
a <- seq(0,2*pi,length=100) # On crée des angles en radians
plot( cos(a), sin(a), # Dessine le cercle
type = 'l', # type de plot = 1 (ligne)
lty = 3, # type de ligne = 3 (3=dotted)
xlab = pca1, ylab = pca2, # label des axes
main = "Corrélation circle") # titre du dessin
arrows(0,0, correlation$PC1, correlation$PC2, col="red") # Flèches
text(correlation$PC1, correlation$PC2,row.names(correlation), col='blue')
abline(h=0, lty=3) # axe horizontal (lty = type de la ligne)
abline(v=0, lty=3) # axe vertical (lty = type de la ligne)

# % de variance des variables expliquée par les axes (COS2)
cos2_var = goodness(acp_resu, display = "species", model = "CA") * 100
cos2_var

# % de variance des axes expliquée par les variables (CTR) - Methode 1
inertie_var = data.frame(inertcomp(acp_resu, display = "species"), row.names=label_variables)
cos2_var = goodness(acp_resu, display = "species", model = "CA") * 100
cos2_var$PC3 = cos2_var$PC3 - cos2_var$PC2
cos2_var$PC2 = cos2_var$PC2 - cos2_var$PC1
# Calcul des contributions brutes et non cumulatives
produit_pc1 = inertie_var$CA * cos2_var$PC1
ctr_pc1 = round(produit_pc1 /sum(produit_pc1)*100, digits = 2)
produit_pc2 = inertie_var$CA * cos2_var$PC2
ctr_pc2 = round(produit_pc2 /sum(produit_pc2)*100, digits = 2)
produit_pc3 = inertie_var$CA * cos2_var$PC3
ctr_pc3 = round(produit_pc3 /sum(produit_pc3)*100, digits = 2)
ctr_var = data.frame(ctr_pc1, ctr_pc2, ctr_pc3, row.names=label_variables)
ctr_var

# % de variance des axes expliquée par les variables (CTR) - Methode 2 (corrélation)
coordo <-scores(acp_resu)
correlation = data.frame(cor(fichier, coordo$sites)) # appel au fichier de départ
contribrut = correlation*correlation
ctr_var1 = contribrut$PC1 / sum(contribrut$PC1) *100
ctr_var2 = contribrut$PC2 / sum(contribrut$PC2) *100
ctr_var = data.frame(ctr_var1, ctr_var2, row.names=label_objets)
ctr_var

# Capacité d'explications pour les objets
COS2obj = goodness(acp_resu, display = "sites", model="CA") * 100
COS2obj

# Contributions relatives des objets aux axes
inertie_obj = data.frame(inertcomp(acp_resu, display = "sites"), row.names=label_objets)
cos2_obj = data.frame(goodness(acp_resu, display = "sites", model = "CA", choice =c(1,2,3)) * 100)
# il faut recalculer des valeurs brutes non cumulatives !
cos2_obj$PC3 = cos2_obj$PC3 - cos2_obj$PC2
cos2_obj$PC2 = cos2_obj$PC2 - cos2_obj$PC1
produit_pc1 = inertie_obj$CA * cos2_obj$PC1
# calcul des CTR
ctr_pc1 = round(produit_pc1 /sum(produit_pc1)*100, digits = 2)
produit_pc2 = inertie_obj$CA * cos2_obj$PC2
ctr_pc2 = round(produit_pc2 /sum(produit_pc2)*100, digits = 2)
produit_pc3 = inertie_obj$CA * cos2_obj$PC3
ctr_pc3 = round(produit_pc3 /sum(produit_pc3)*100, digits = 2)
ctr_obj = data.frame(ctr_pc1, ctr_pc2, ctr_pc3, row.names=label_objets)
ctr_obj

# Utilisation d'une routine toute faite
ACP_resultat <-function(pca)
{
# === EXTRAIT DE LEGENDRE 2008 ===

# Valeurs propres
ev = pca$CA$eig

# Quelles sont les valeurs propres plus grandes que la moyenne?
ev[ev > mean(ev)]

# Modele du baton brise (broken stick model)
n = length(ev)
bsm = data.frame(j=seq(1:n), p=0)
bsm$p[1] = 1/n
for (i in 2:n) {
bsm$p[i] = bsm$p[i-1] + (1/(n + 1 - i))
}
bsm$p = 100*bsm$p/n
bsm

# Dessiner les valeurs propres et le % de variance de chaque axe
par(mfrow=c(2,1))
barplot(ev, main="Valeurs propres", col="bisque", las=2)
abline(h=mean(ev), col="red") # moyenne des valeurs propres
legend("topright", "Moynne des valeurs propres", lwd=1, col=2, bty="n")
barplot(t(cbind(100*ev/sum(ev),bsm$p[n:1])), beside=T,
main="% variance", col=c("bisque",2), las=2)
legend("topright", c("% valeur propre", "Broken stick model"),
pch=15, col=c("bisque",2), bty="n")

# Dessin des variables environnementales pour les axes 1 et 2
# *****

# Scores des sites et "species" (ici les variables environnementales)
# Default: cadrage 2
scores(pca)
scores(pca, choices=1:2)
sit.sc2 = scores(pca, display="wa")
spe.sc2 = scores(pca, display="sp")
# Cadrage 1
sit.sc1 = scores(pca, display="wa", scaling=1)
spe.sc1 = scores(pca, display="sp", scaling=1)

# Dessin des variables seulement (fleches!)
windows()
# Mac: quartz()
plot(pca, display="sp", type="n",
main="ACP correlation - Variables environnementales ")
text(pca, display="sp", cex=1, pos=4, col="red")
arrows(0, 0, spe.sc2[,1], spe.sc2[,2], length=0.07, angle=20, col="red")

# 1c. Biplots (cadrages 1 et 2)
# *****
windows(12,6)
par(mfrow=c(1,2))
# Cadrage 1 "distance biplot"
plot(pca, scaling=1, main="Biplot PCA distance - cadrage 1 - DI conservées")
arrows(0, 0, spe.sc1[,1], spe.sc1[,2], length=0.07, angle=20, col="red")
# Cadrage 2 (default): "correlation biplot"
plot(pca, main="Biplot PCA correlation - cadrage 2 - corrélations conservées")
arrows(0, 0, spe.sc2[,1], spe.sc2[,2], length=0.07, angle=20, col="red")

# On peut utiliser 2 differents cadrages des vecteurs propres a l'aide de
# l'argument scaling :
# scaling = 1 : preserve les distances euclidiennes entre les objets
# scaling = 2 : preserve les correlations entre les descripteurs (ou variables)
# Les cadrages de vegan ne correspondent pas tout a fait a ceux de la theorie.
# Jari Oksanen leur fait subir quelques ajustements un peu mysterieux...
# *****
}
```

ACP_resultat(acp_resu);

Une bonne combinaison : cadrage 1 pour les objets + Cercle des corrélations