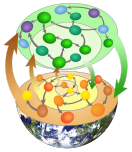


Séance 6: La visualisation des données

BIO 500 - Méthodes en écologie computationnelle

Dominique Gravel & Steve Vissault
Laboratoire d'écologie intégrative

Séance 6

- ✓ Ces diapositives sont disponibles en **version web** et en **PDF**.
- ✓ L'ensemble du matériel de cours est disponible sur la page du portail **moodle**.
- ✓ Vous trouverez du matériel supplémentaire dans le **cours** de **Kevin Cazelles** et **Nicolas Casajus** lors d'un atelier de communication visuelle du CSBQ.
- ✓ Certaines diapositives sont également extraites de la présentation de **David Taylor**

Faire une figure étape par étape avec R

Préparer les données adéquatement

- ✓ Habituellement un `data.frame` ou `une matrice`
- ✓ Une observation par ligne (format long)

Ouvrir une fenêtre graphique

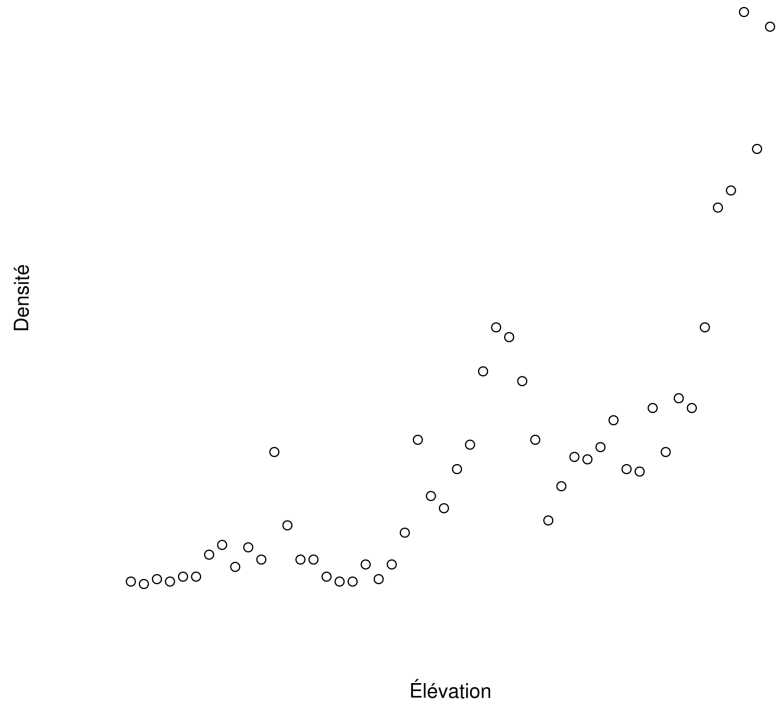
```
dev.new(width = 10, height = 7)
```

Fixer certains paramètres

```
# Fixer la largeur et la hauteur des marges
par(mar = c(5,6,2,1))

# Fixer le nombre de figures en colonnes et rangées
par(mfrow = c(1,1))
```

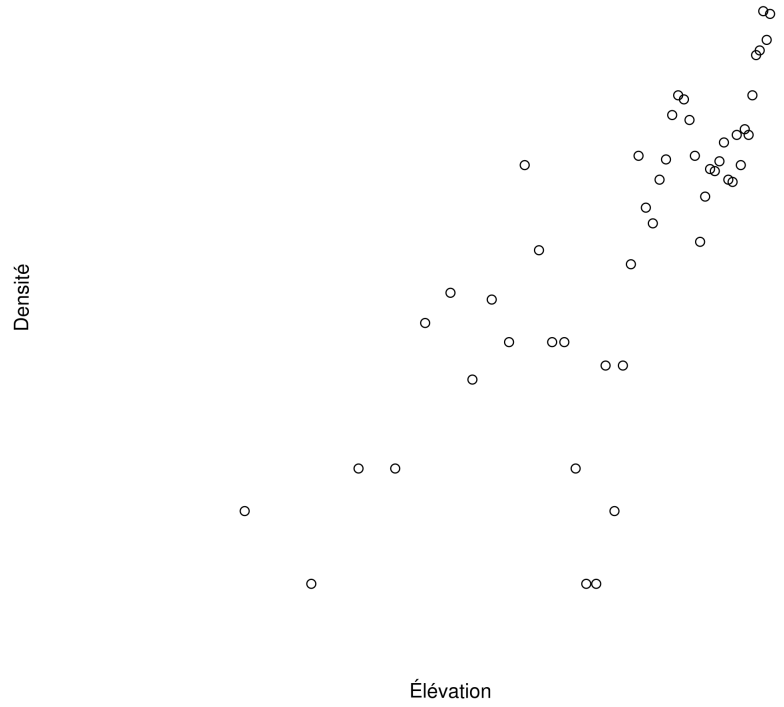
Démarrer une figure avec `plot()`

```
arbres <- read.csv2("donnees/arbres.csv")
densite <- table(arbres[,c(3,5)])
elevation <- as.numeric(row.names(densite))
plot(elevation, densite[,1], axes = FALSE,
      xlab = "Élévation", ylab = "Densité")
```



Échelles logarithmiques

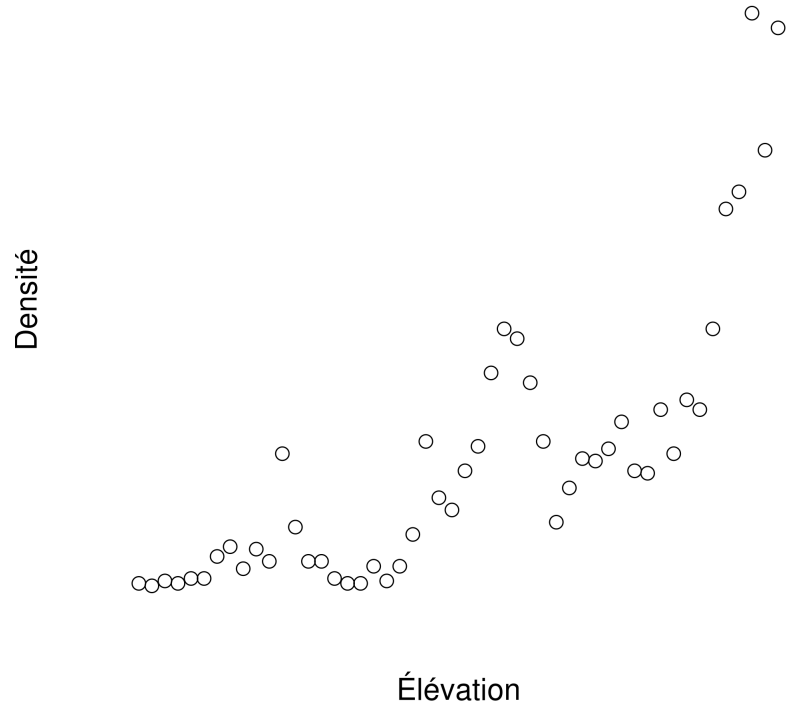
```
plot(elevation, densite[,1], axes = FALSE,
      xlab = "Élévation", ylab = "Densité",
      log = "xy")
```



Ajuster les tailles de caractères

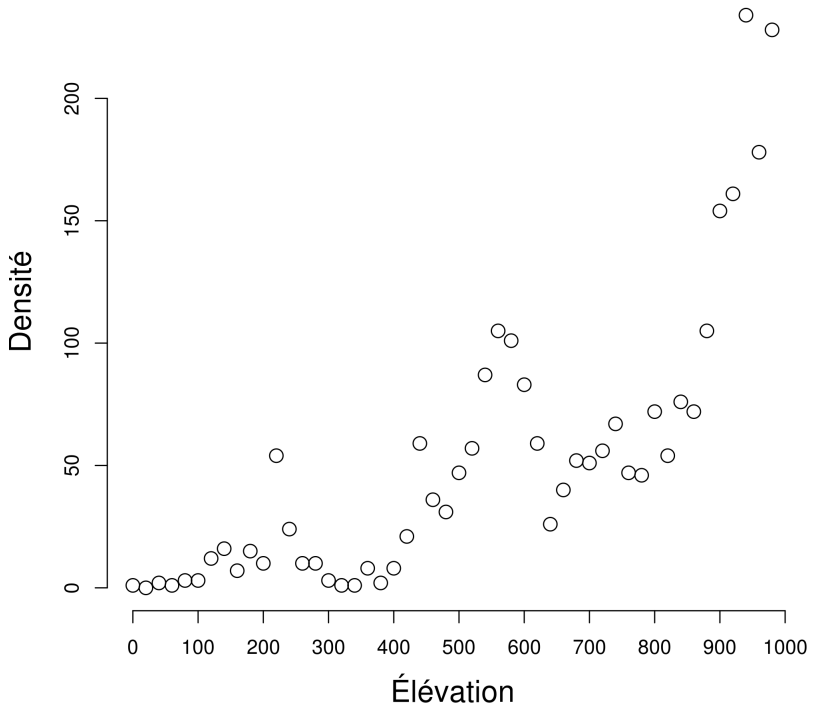
`cex` `cex.lab` `cex.axis`

```
plot(elevation, densite[,1], axes = FALSE,
      xlab = "Élévation", ylab = "Densité",
      cex.lab = 1.5, cex.axis = 1.25, cex = 1.5)
```



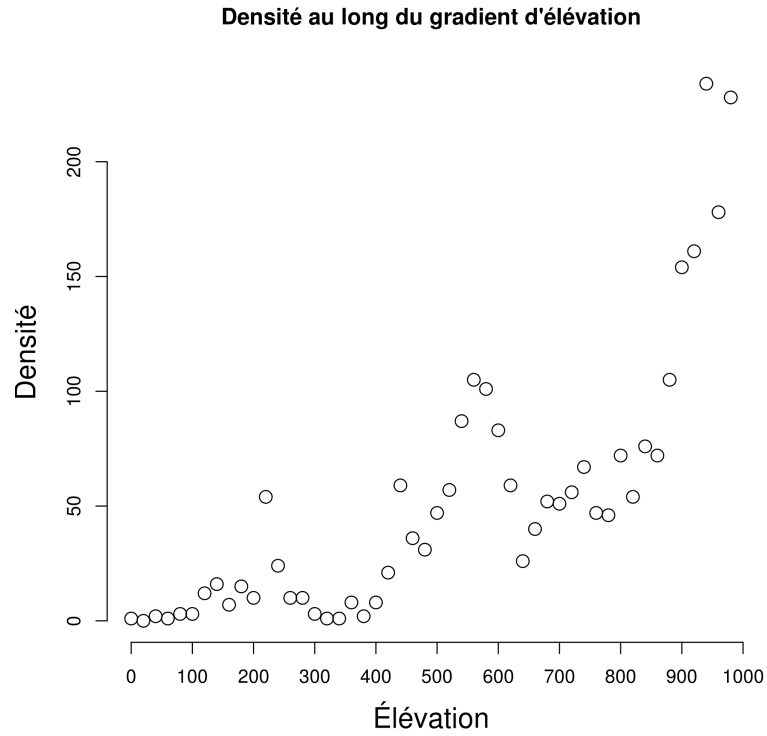
Modifier les axes

```
axis(1, seq(0,1000,100))
axis(2)
```



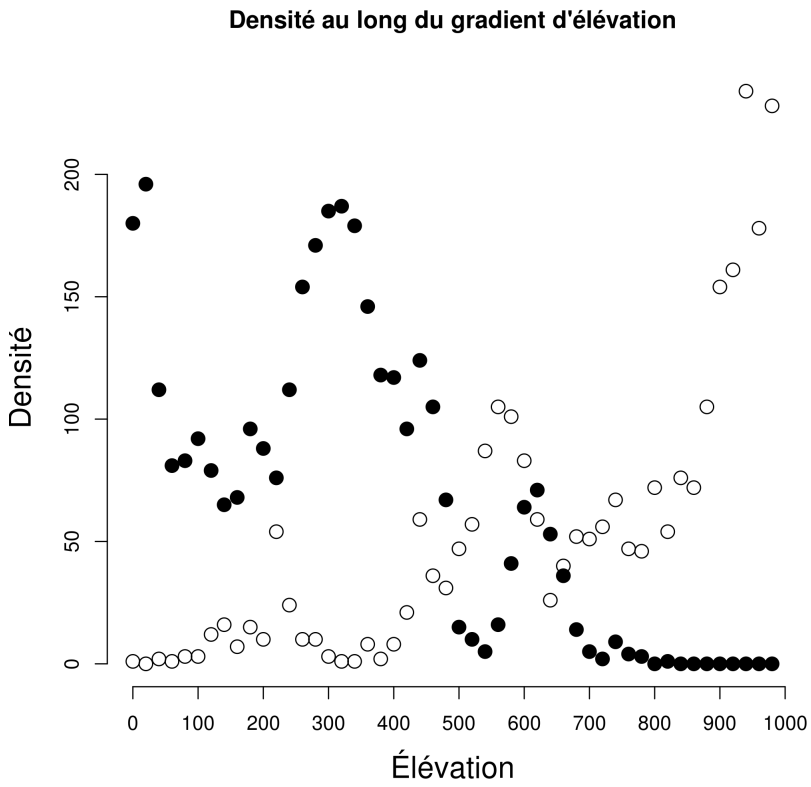
Ajouter un titre

```
title(main = "Densité au long du gradient d'élévation")
```



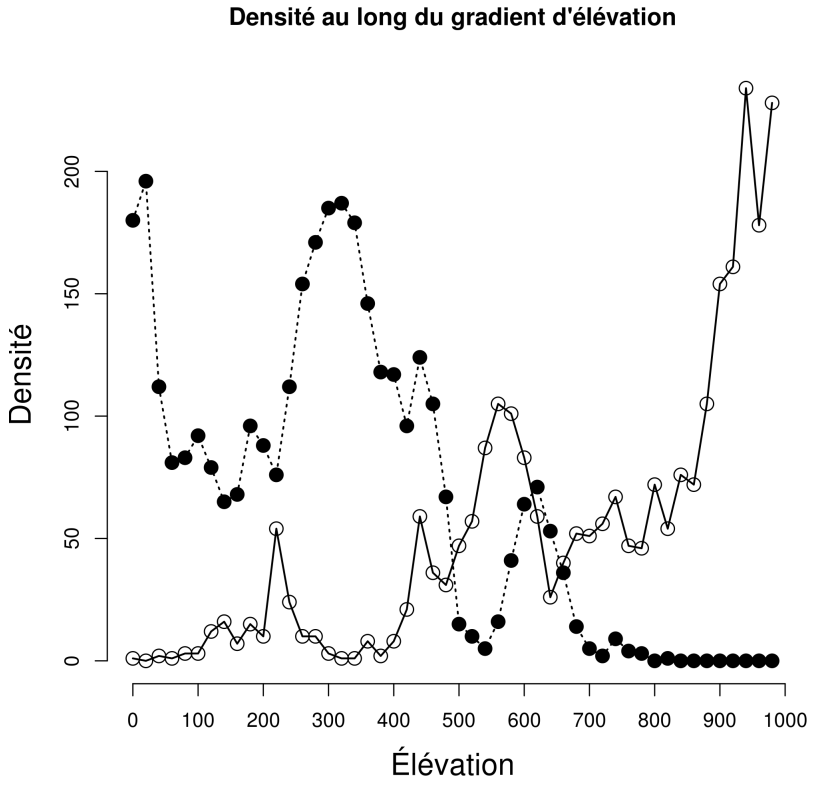
Superposer des points d'une autre série de données

```
points(elevation, densite[,3], pch = 19, cex = 1.5)
```



Superposer des lignes

```
lines(elevation, densite[,1],lty = 1, lwd = 1.5)
lines(elevation, densite[,3], lty  = 3, lwd = 1.5)
```



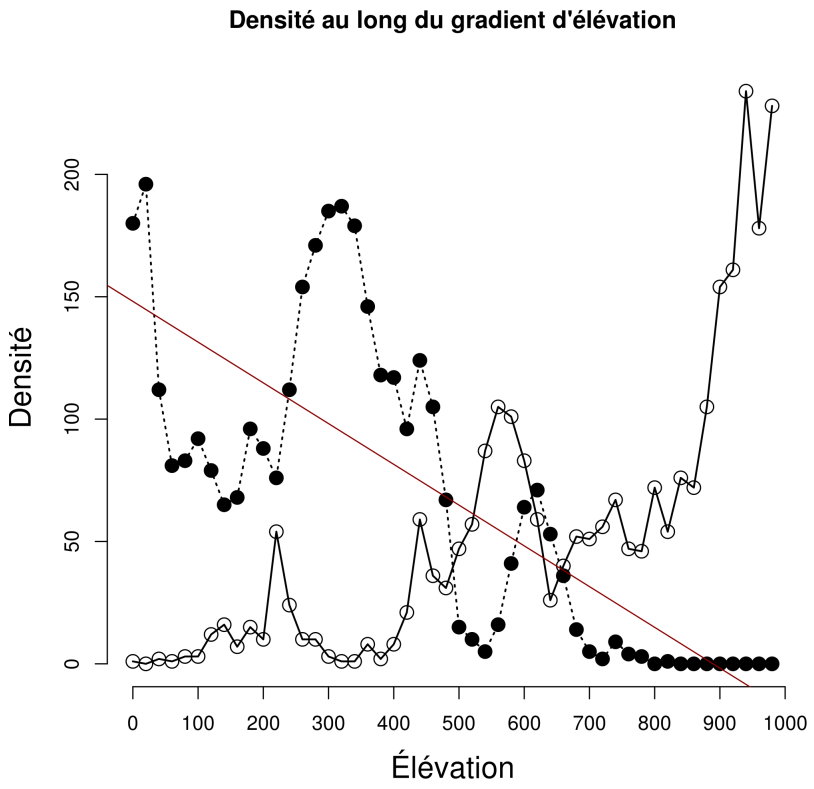
Ajouter une ligne de tendance

```
model = lm(densite[,3]~elevation)
summary(model)
abline(model, col = "darkred")
```

```
##
## Call:
## lm(formula = densite[, 3] ~ elevation)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -59.796 -26.743  -3.565  24.050  92.175
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 148.10588   11.23433  13.183   <2e-16 ***
## elevation    -0.16650    0.01976  -8.428    5e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 40.32 on 48 degrees of freedom
## Multiple R-squared:  0.5968, Adjusted R-squared:  0.5884
## F-statistic: 71.04 on 1 and 48 DF,  p-value: 4.999e-11
```

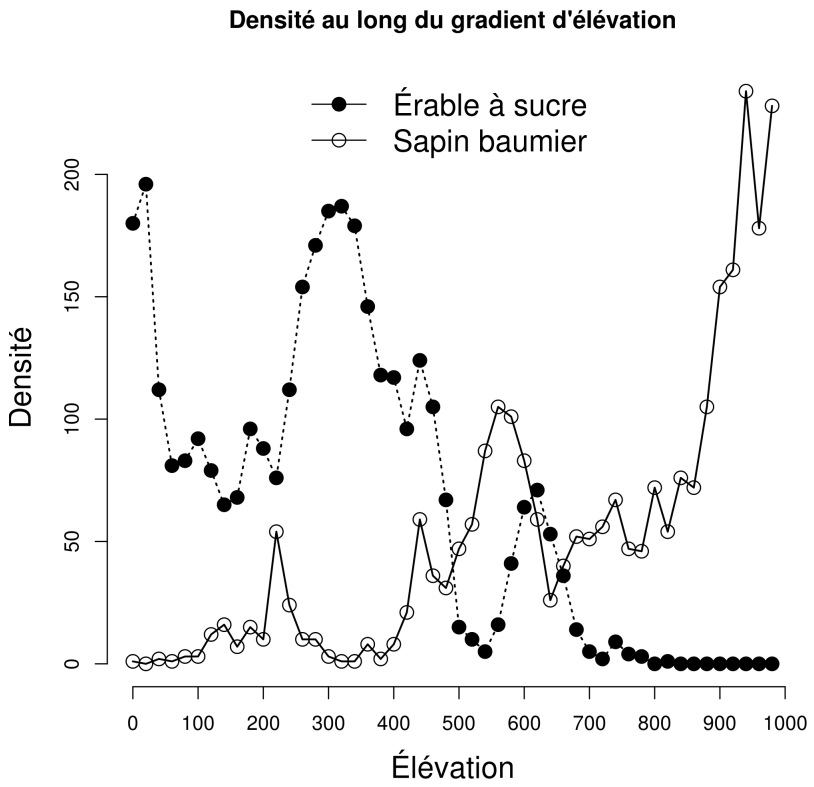
Ajouter une ligne de tendance

```
model = lm(densite[,3]~elevation)
abline(model, col = "darkred")
```



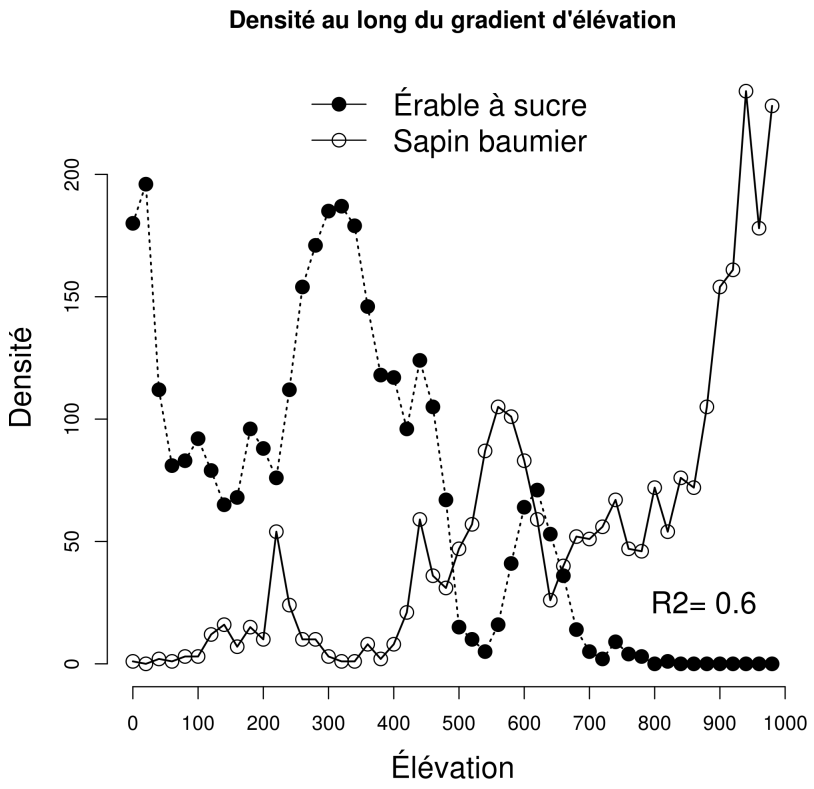
Ajouter une légende

```
legend("top", bty = "n", pch = c(19,1), lty = 1,
    legend = c("Érable à sucre", "Sapin baumier"),
    cex = 1.5)
```



Ajouter du texte

```
r2 <- round(summary(model)$r.squared, 2)
text(x = 850, y = 25, paste("R2=",r2),
    cex = 21.5)
```



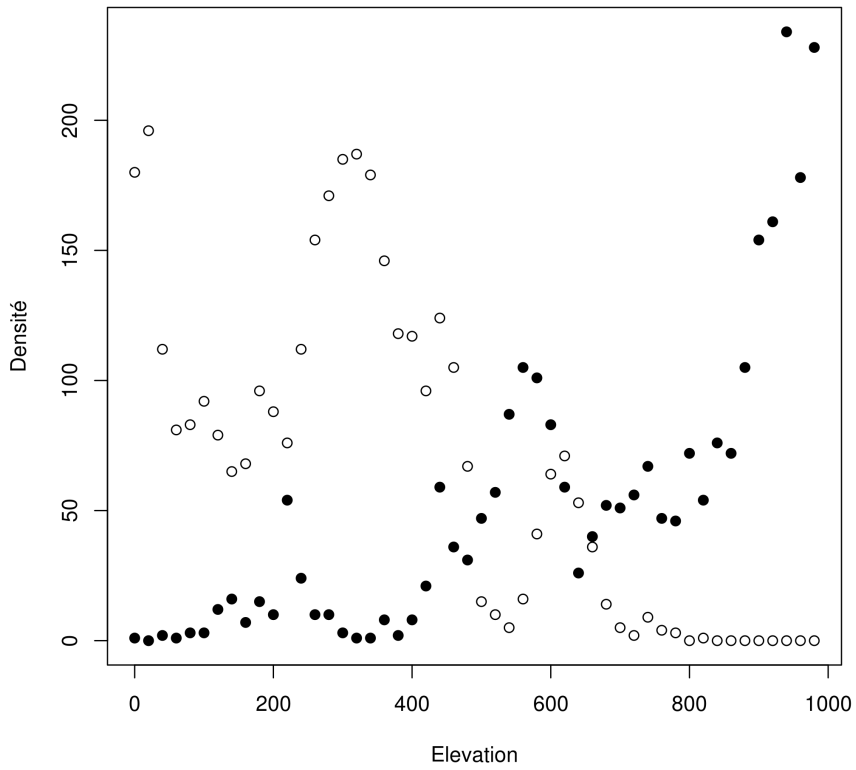
Pour plus d'information

- ✓ `?plot`
- ✓ `?par`
- ✓ `?axis`
- ✓ `?mtext`

Créer d'autres types de figure

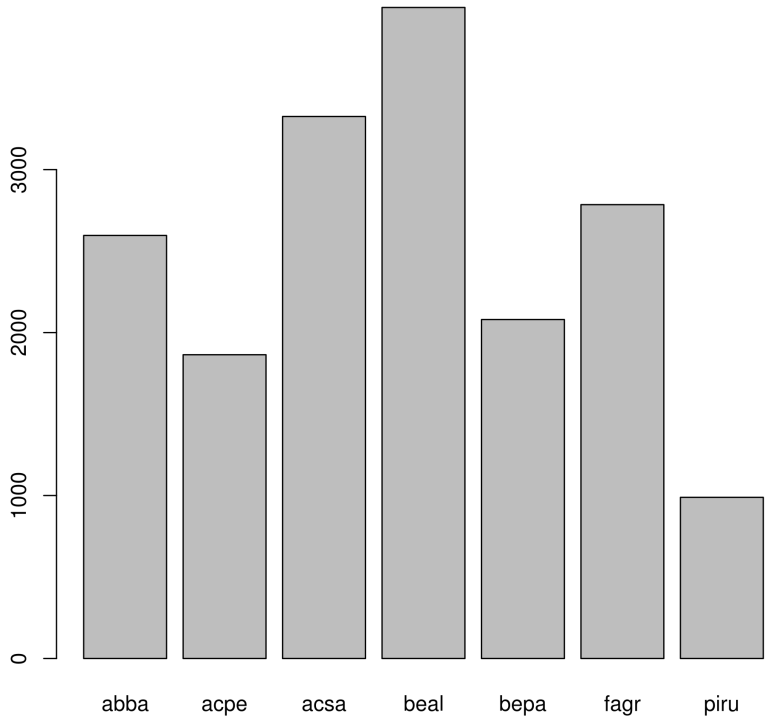
Diagramme de dispersion (Scatter plot)

```
arbres  <- read.csv2("donnees/arbres.csv")
densite <- table(arbres[,c(3,5)])
elevation <- as.numeric(row.names(densite))
plot(elevation, densite[,1], pch = 19,
  xlab = "Elevation", ylab = "Densité")
points(elevation, densite[,3])
```



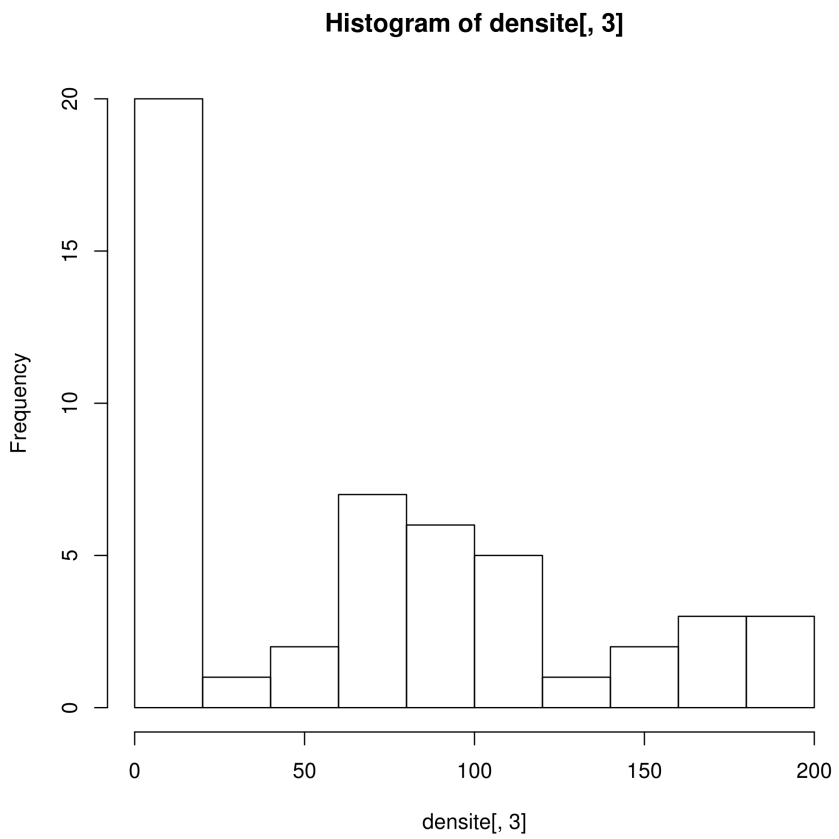
Diagrammes à bâtons (Bar plot)

```
arbres  <- read.csv2("donnees/arbres.csv")
n_tot <- table(arbres$esp)
barplot(n_tot)
```



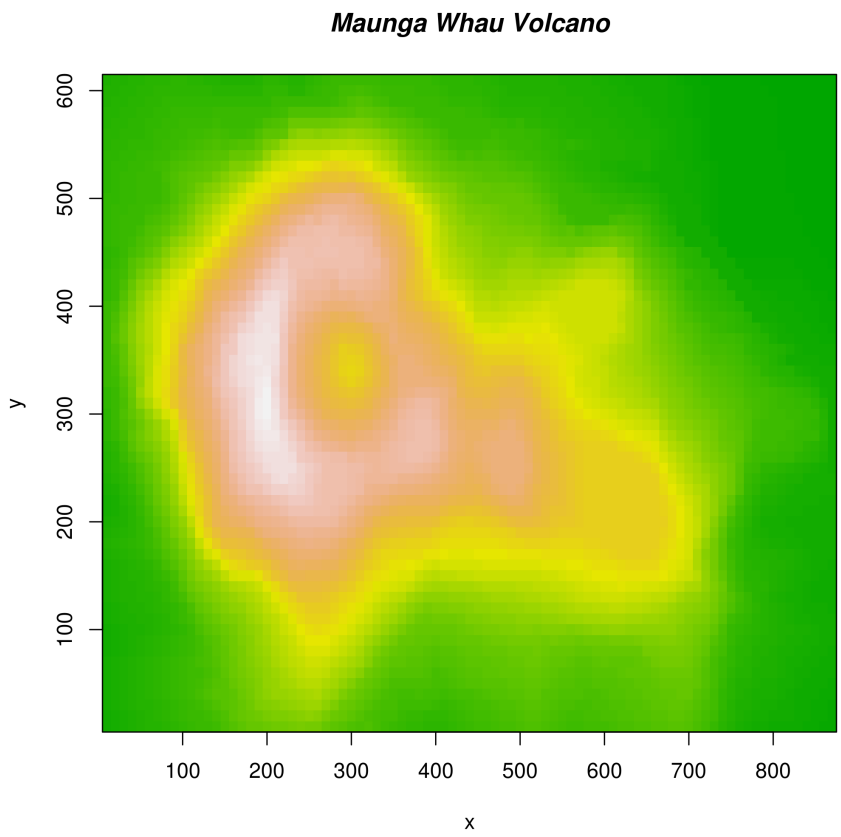
Histogrammes

```
hist(densite[,3])
```



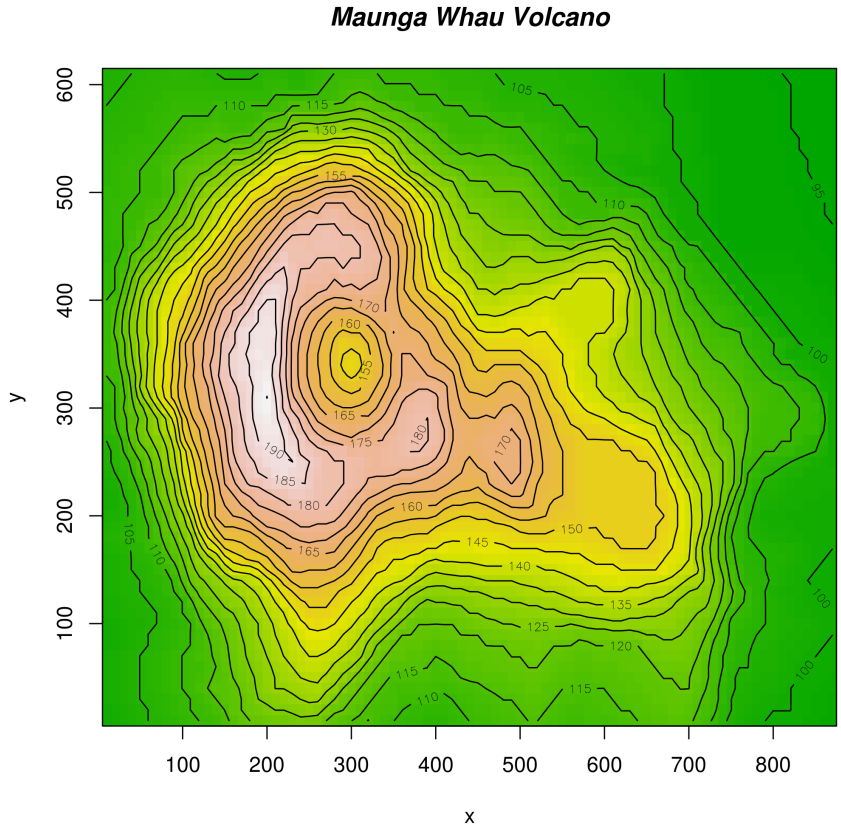
Représentation 3-D

```
x <- 10*(1:nrow(volcano))
y <- 10*(1:ncol(volcano))

image(x, y, volcano,
  col = terrain.colors(100), axes = FALSE)

axis(1, at = seq(100, 800, by = 100))
axis(2, at = seq(100, 600, by = 100))
box()

title(main = "Maunga Whau Volcano", font.main = 4)
```



Lignes de contour

```
x <- 10*(1:nrow(volcano))
y <- 10*(1:ncol(volcano))

image(x, y, volcano,
  col = terrain.colors(100), axes = FALSE)

axis(1, at = seq(100, 800, by = 100))
axis(2, at = seq(100, 600, by = 100))
box()

title(main = "Maunga Whau Volcano", font.main = 4)

contour(x, y, volcano,
  levels = seq(90, 200, by = 5),
  add = TRUE, col = "black")
```



Enregistrer une figure

```
dev.copy2pdf(file = "test.pdf")
dev.copy2png(file = "test.png")
dev.copy2eps(file = "test.eps")
```

Exercice: faire une représentation visuelle de la distribution de degrés

La visualisation de réseau avec igraph

Installation

```
install.packages("igraph")
```

```
## Installing package into '/home/travis/R/Library'
## (as 'lib' is unspecified)
```

```
library(igraph)
```

```
##
## Attaching package: 'igraph'
```

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

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

Transformer une matrice d'adjacence en objet **igraph**

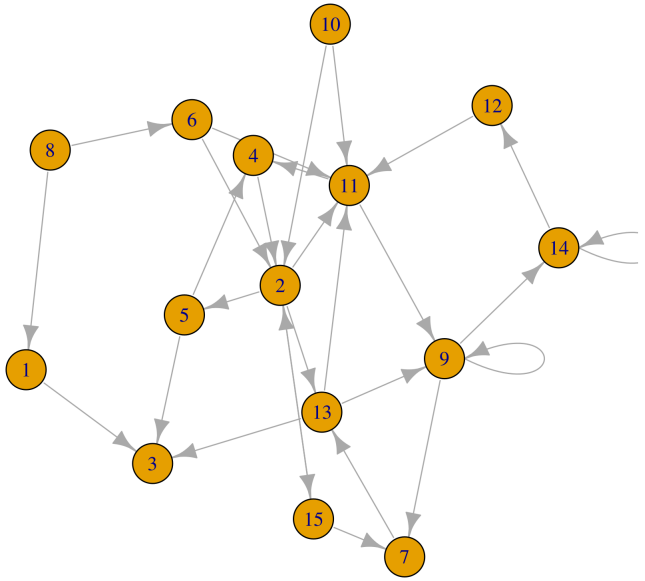
```
library(igraph)
C <- 0.1
S <- 15
L <- matrix(0, nr = S, nc = S)
L[runif(S*S) < C] = 1
sum(L)
```

```
## [1] 28
```

```
g <- graph.adjacency(L)
```

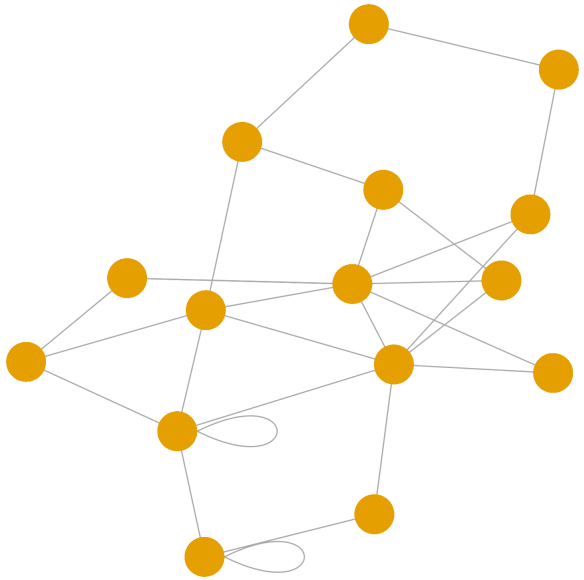
Utiliser la fonction `plot` pour faire une représentation d'un réseau

```
plot(g)
```



Version plus esthétique sans les paramètres par défaut

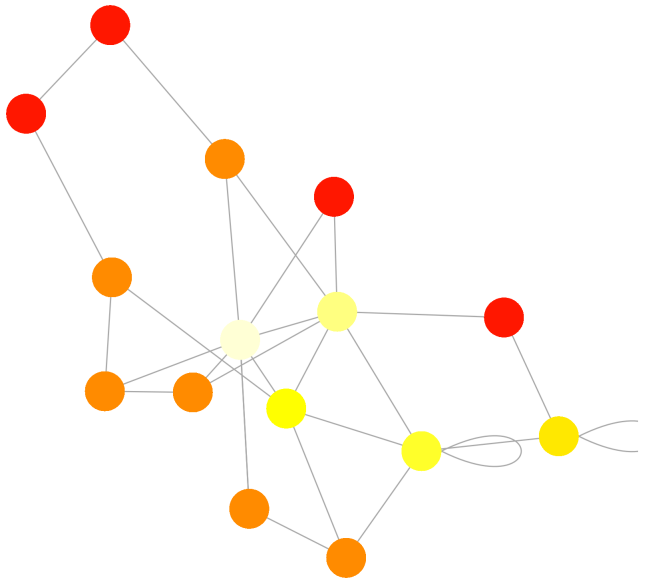
```
plot(g, vertex.label=NA, edge.arrow.mode = 0,
    vertex.frame.color = NA)
```



Exercice : Compiler la matrice d'adjacence et faire une première représentation du réseau avec `igraph`

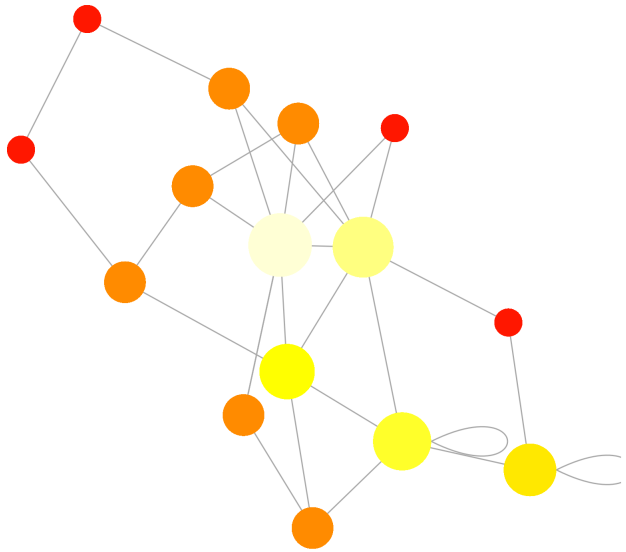
Changer la couleur des noeuds

```
# Calculer le degré
deg <- apply(L, 2, sum) + apply(L, 1, sum)

# Le rang pour chaque noeud
rk <- rank(deg)

# Faire un code de couleur
col.vec <- heat.colors(S)

# Attribuer aux noeuds la couleur
V(g)$color = col.vec[rk]

# Refaire la figure
plot(g, vertex.label=NA, edge.arrow.mode = 0,
    vertex.frame.color = NA)
```



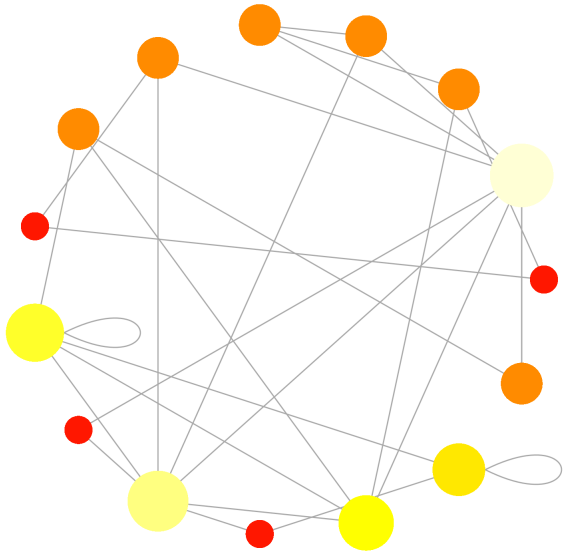
Changer la taille des noeuds

```
# Faire un code de ctaille
col.vec <- seq(10, 25, length.out = S)

# Attribuer aux noeuds la couleur
V(g)$size = col.vec[rk]

# Refaire la figure
plot(g, vertex.label=NA, edge.arrow.mode = 0,
    vertex.frame.color = NA)
```



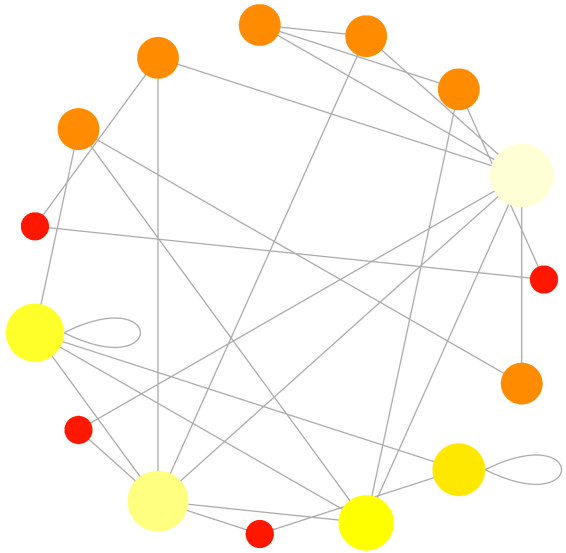
Changer la disposition des noeuds

```
plot(g, vertex.label=NA, edge.arrow.mode = 0,
    vertex.frame.color = NA, layout = layout.reingold.tilford(g))
```



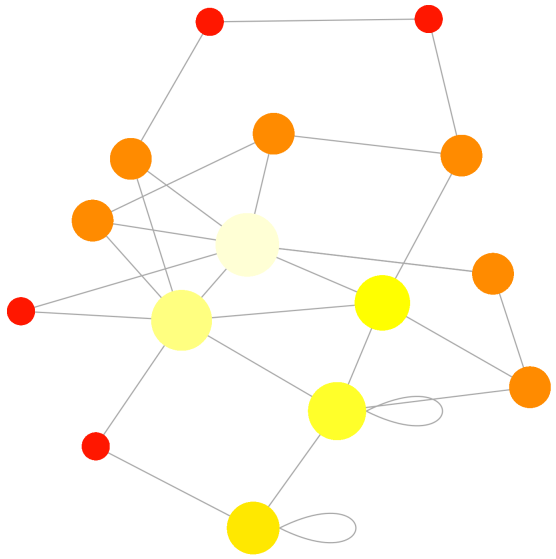
Changer la disposition des noeuds

```
plot(g, vertex.label=NA, edge.arrow.mode = 0,
    vertex.frame.color = NA,
    layout = layout.circle(g))
```



Changer la disposition des noeuds

```
plot(g, vertex.label=NA, edge.arrow.mode = 0,
    vertex.frame.color = NA,
    layout = layout.kamada.kawai(g))
```



Calcul de propriétés

```
wtc = walktrap.community(g)
modularity(wtc)
```

```
## [1] 0.2378827
```

Calcul de propriétés

```
distances(g)
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13]
##  [1,]    0    3    1    3    2    2    3    1    3     4     3     4     2
##  [2,]    3    0    2    1    1    1    2    2    2     1     1     2     1
##  [3,]    1    2    0    2    1    3    2    2    2     3     2     3     1
##  [4,]    3    1    2    0    1    2    3    3    2     2     1     2     2
##  [5,]    2    1    1    1    0    2    3    3    3     2     2     3     2
##  [6,]    2    1    3    2    2    0    3    1    2     2     1     2     2
##  [7,]    3    2    2    3    3    3    0    4    1     3     2     3     1
##  [8,]    1    2    2    3    3    1    4    0    3     3     2     3     3
##  [9,]    3    2    2    2    3    2    1    3    0     2     1     2     1
## [10,]    4    1    3    2    2    2    3    3    2     0     1     2     2
## [11,]    3    1    2    1    2    1    2    2    1     1     0     1     1
## [12,]    4    2    3    2    3    2    3    3    2     2     1     0     2
## [13,]    2    1    1    2    2    2    1    3    1     2     1     2     0
## [14,]    4    3    3    3    4    3    2    4    1     3     2     1     2
## [15,]    4    1    3    2    2    2    1    3    2     2     2     3     2
##       [,14] [,15]
##  [1,]     4     4
##  [2,]     3     1
##  [3,]     3     3
##  [4,]     3     2
##  [5,]     4     2
##  [6,]     3     2
```

Calcul de propriétés

```
eigen_centrality(g)$vector
```

```
##  [1] 0.08666383 1.00000000 0.26963740 0.51132514 0.39669367 0.44956962
##  [7] 0.43078052 0.11944126 0.66537719 0.42296513 0.89890871 0.25924396
## [13] 0.72718392 0.26497118 0.54143488
```

Exporter des tableaux

Exporter des tableaux

Exporter des tableaux depuis R vers son document de travail peut être difficile.

1. Enregistrer le `data.frame` dans un fichier avec la fonction `write.table()` ou `write.csv()`
2. Éditer et faire la mise en page dans MS Excel ou MS Word.

Le package `knitr` permet de faciliter cette procédure en exportant le `data.frame` directement dans son document de travail LaTeX.

Exporter des tableaux

Prenons le jeu de données `iris` directement disponible sous R.

```
data(iris)
class(iris)
```

```
## [1] "data.frame"
```

```
head(iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

Exporter des tableaux

Je souhaite maintenant exporter ce `data.frame` en LateX (un format que nous verrons lors de la prochaine séance):

```
library(knitr)
iris_tex <- kable(iris,format="latex")
writeLines(iris_tex, con = "./donnees/iris.tex", sep = "\n", useBytes = FALSE)
```

Travail de la semaine

Consignes

- ✓ Identifier clairement vos questions de recherche
- ✓ Illustrer le réseau de collaborations
- ✓ Compléter votre analyse au moyen de 3 figures et 1 tableau
- ✓ Mettre à jour le makefile

Évaluation

- ✓ Clareté des questions et adéquation des figures et du tableau
- ✓ Efficacité de la présentation
- ✓ Respect de normes graphiques
- ✓ Originalité

Essai

Objectifs

L'objectif d'un essai est de présenter une perspective sur un enjeu scientifique, appuyé par une argumentation logique et une lecture critique de la littérature. L'objectif spécifique de ce travail est de formuler et défendre une opinion sur les enjeux de reproductibilité en écologie.

Mise en situation

Vous êtes invités à préparer un article pour le journal Québec Science, où on vous a demandé de rédiger l'éditorial du mois sur cet enjeu. Vous devez défendre par quels moyens vous pourriez améliorer la reproductibilité de la science si vous étiez en charge d'un laboratoire dans une institution universitaire. Vous êtes invités à faire une lecture critique de la situation actuelle et à proposer des mesures qui permettront de répondre aux enjeux identifiés.

Attentes

Québec Science est un journal destiné à un grand public, alors je vous invite à personnaliser votre argumentation et à rendre original sa présentation. Vous pouvez utiliser des tableaux, des figures ou encore des encadrés pour étayer vos propos. Essayez de faire plus que de rapporter les arguments présentés en classe, n'hésitez pas à personnaliser votre essai.

Consignes

- ✓ Le texte doit faire au maximum 1500 mots et doit être accompagné d'un résumé court, provocateur de 100 mots. Le document peut être supporté par une figure et/ou un tableau.
- ✓ Le texte peut être structuré en sections afin de permettre au lecteur de suivre le développement de l'argumentaire.
- ✓ La section finale doit résumer les points principaux.
- ✓ Utiliser LaTeX pour la mise en forme du document.
- ✓ L'argumentaire doit être supporté de littérature scientifique appropriée. Vous pouvez certes utiliser les articles discutés en classe, mais essayez d'aller puis au-delà de ces références pour alimenter vos arguments.

Évaluation

- ✓ Respect des consignes
- ✓ Titre et résumé
- ✓ Formulation de la proposition
- ✓ Qualité de l'argumentation
    - Identification des problèmes
    - Proposition de solutions
- ✓ Originalité
- ✓ Mise en page
- ✓ Bibliographie
- ✓ Qualité de la langue