Visualisation de Réseaux

Thomas OLLARD, Ludovic PLANCHON, Abdellah MAGHOUS

06/12/2020

##Introducuion

##Qu'est-ce qu'un réseau?

De manière informelle,un réseau est un ensemble de nœuds reliés par un ensemble de lignes ou de flèches,pour ce faire on utilise différent packages comme GGally qui contient la fonction La ggnet2 qui est une fonction de visualisation permettant de tracer des objets réseau en tant ggplot2 qu'objets.

```
library(GGally)
```

On va utiliser 3 packages :

==> network packages classes les données relationnelles pour créer et modifier des objets réseau ,qui représente une plage de types de données relationnelles et prend en charge des attributs arbitraires de sommet

=> Sna packages est un ensemble d'outils qui permet d'analyse des réseaux sociaux, y compris les indices au niveau des nœuds et des graphes, les méthodes de distance structurelle et de covariance, la génération des graphes aléatoires et la visualisation de réseau 2D / 3D.

=> ggplot2 crée des visualisations de données élégantes à l'aide de la grammaire des graphiques

```
library(network)
library(sna)
library(ggplot2)
```

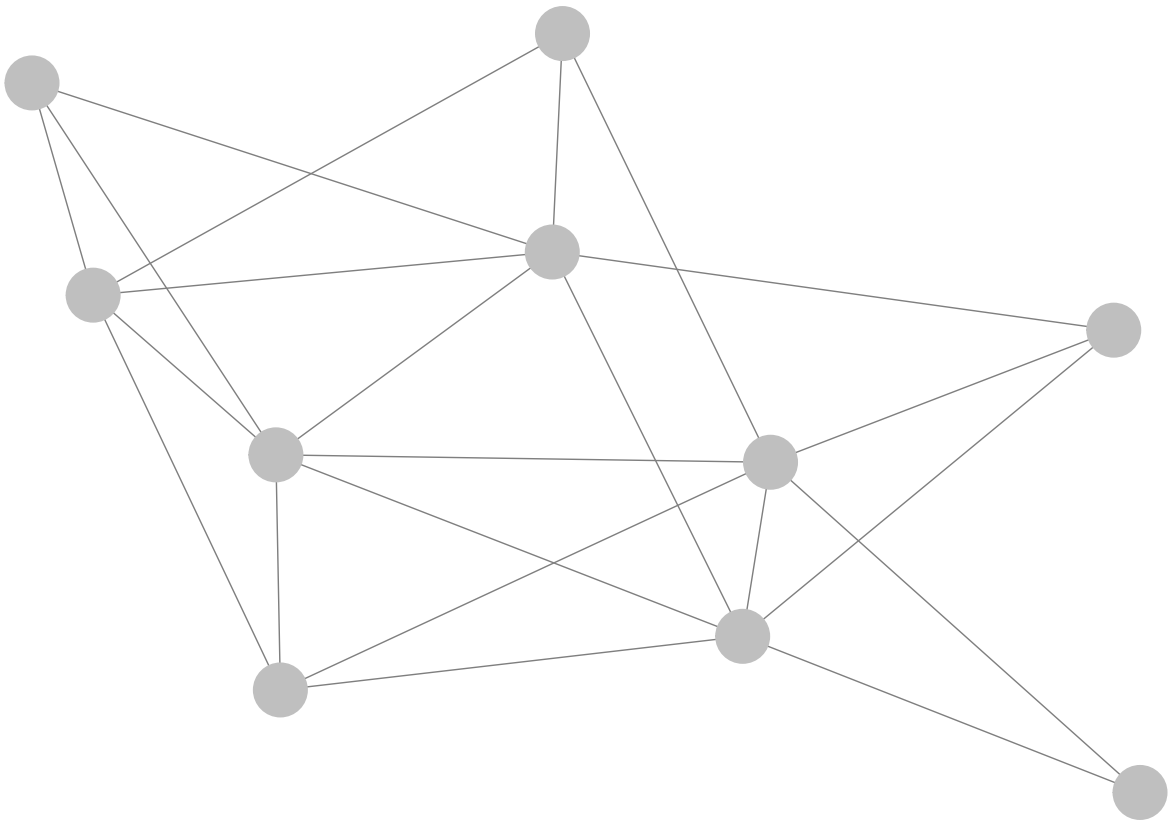
Il y a trois composants principaux que nous considérons lors de la modélisation d'un réseau:

=>Nœuds Les nœuds sont les points du réseau qui représentent les entités que nous voulons modéliser (réseau social)

=>Liens Les liens sont les lignes qui relient les nœuds.Ils représentent la connexion que nous voulons modéliser entre les entités (nœuds) de notre réseau. Nous pouvons dessiner des liens pour montrer qui est ami avec qui sur ce réseau.

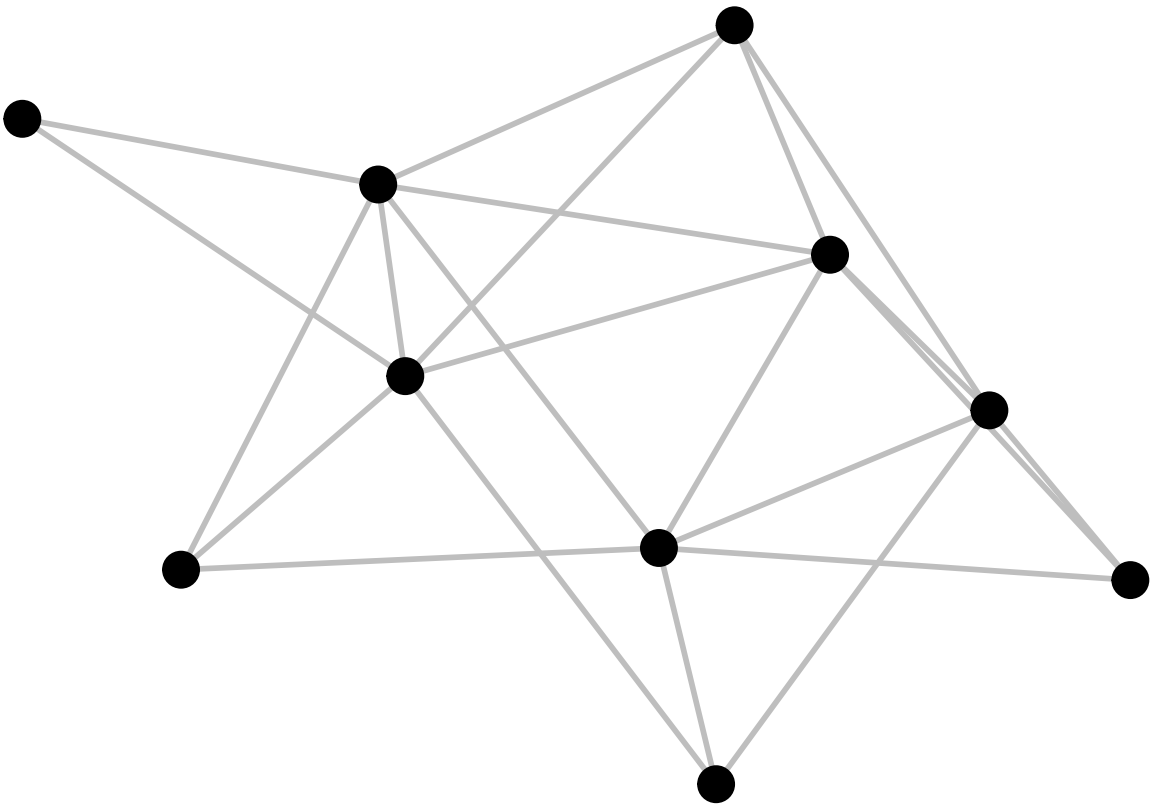
=>Les attributs nous donnent plus d'informations sur notre réseau. Dans notre réseau social, les attributs de nœud décrivent les individus dans le système, y compris des détails tels que l'âge, le nom, le sexe,etc et Nous pouvons afficher ces attributs sur un réseau avec différentes couleurs ou tailles.

```
# random graph
net = rgraph(10, mode = "graph", tprob = 0.5)
net = network(net, directed = FALSE)
# vertex names
network.vertex.names(net) = letters[1:10]
#visualisation
ggnet2(net)
```



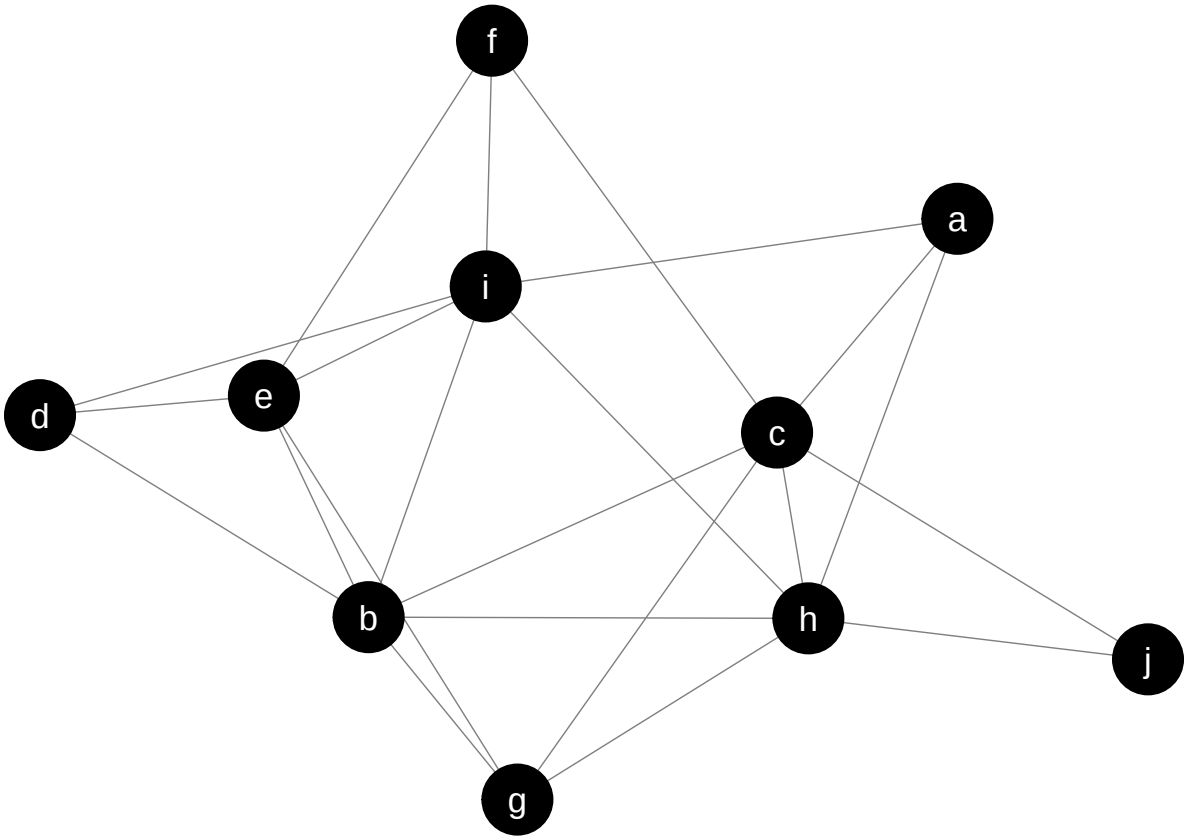
On peut modifier la taille et la couleur des nœuds ou la taille et la couleur des bords

```
ggnet2(net, node.size = 6, node.color = "black", edge.size = 1, edge.color = "grey")
```



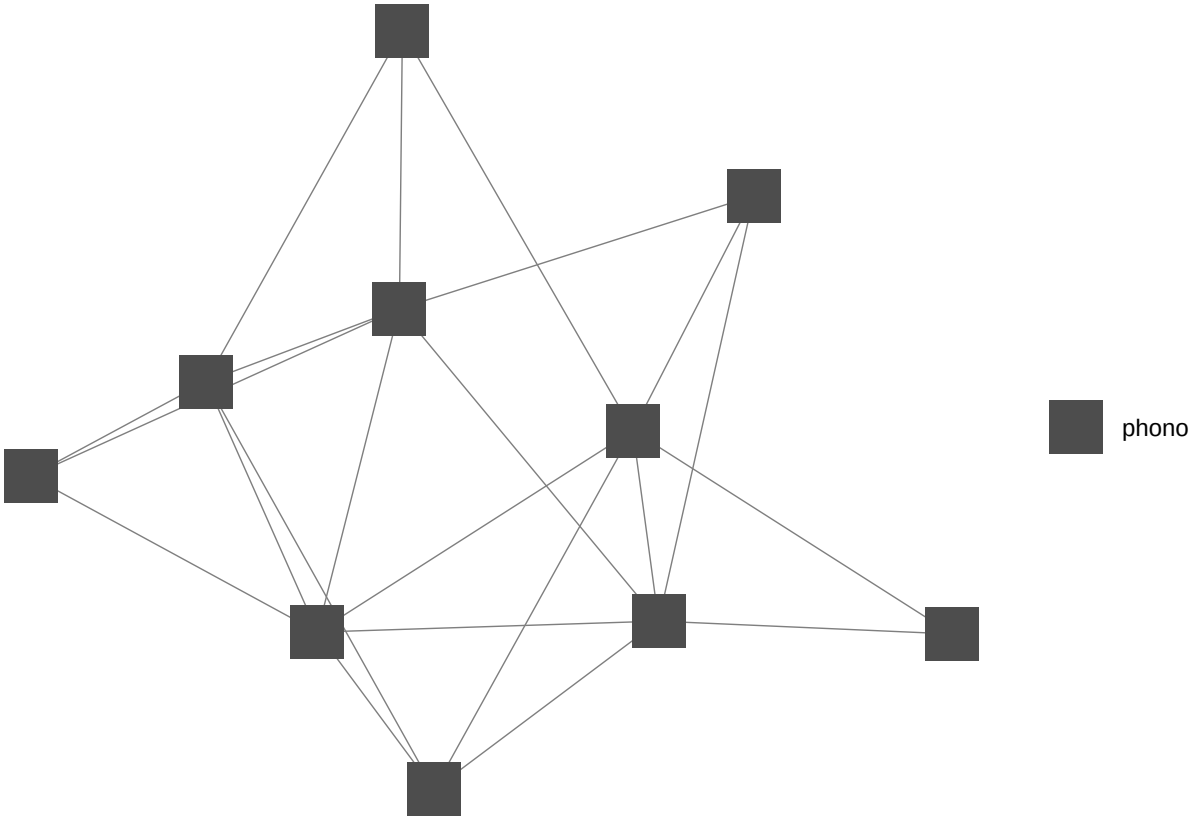
##Étiquettes de noeud Avec l'argument label et la fonction ggnet2 ,on peut étiqueter les nœuds d'un réseau en utilisant leurs noms de vertex, un autre attribut de sommet ou tout autre vecteur d'étiquettes

```
ggnet2(net, size = 12, label = TRUE, color = "black", label.color = "white")
```



Les formes et la transparence des nœuds peuvent être définies exactement comme la couleur et la taille des nœuds, soit via une valeur unique, un vecteur de valeurs (numériques) ou un attribut de sommet.

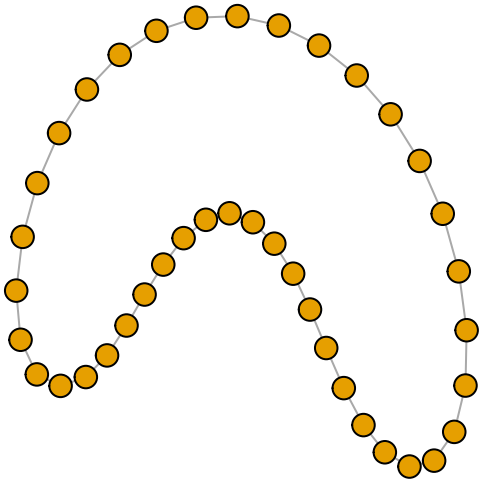
```
ggnet2(net, color = "phono", shape = 15)
```



##modélisation mathématique

on va utilier le package igraph pour créer et manipuler des graphiques et analyser des réseaux basées sur la longueur de chemin, ainsi que des composants et des motifs de graphiques

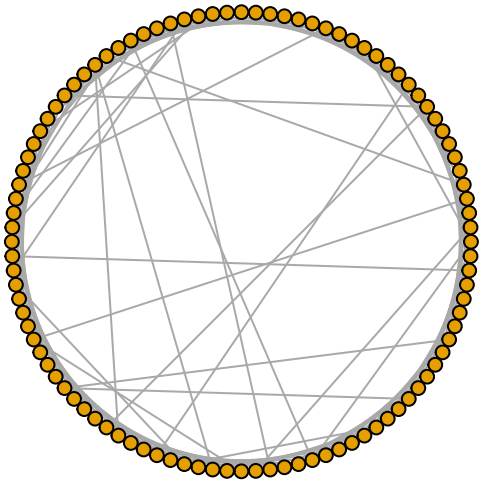
```
install.packages("igraph")
library(igraph)
rn <- make_ring(40)
plot(rn, vertex.size=10, vertex.label=NA)
```



les noeuds sont liés par hasard ,s'il y'en a une connexion entre les noeuds la probabilité est p sinon il est 1-p et et on suppose qu'on a N noeuds et chaque noeuds à N-1 possibilité d'avoir une liason avec d'autre noeud pour ce faire on va modéliser cette expérience par le lancement d'un dé biasée avec X une variable alétoire

réel qui suit un loi binomial P(X=K)=(N-1)C(k)pˆK*(1-p)ˆ(N-1-K)

```
sw <- sample_smallworld(dim=2, size=10, nei=1, p=0.1)
plot(sw, vertex.size=6, vertex.label=NA, layout=layout_in_circle)
```



voire l'Annexe pour les testes d'hypthése

##Etudes de cas

```
library(tidyverse)
library(tidygraph)
library(babynames)
library(ggraph)
```

création de la table des noeuds

head(babynames)

```
head(babynames)
```

```
## # A tibble: 6 x 5
##    year sex   name          n   prop
##   <dbl> <chr> <chr>     <int>  <dbl>
## 1  1880 F     Mary       7065 0.0724
## 2  1880 F     Anna       2604 0.0267
## 3  1880 F     Emma       2003 0.0205
## 4  1880 F     Elizabeth  1939 0.0199
## 5  1880 F     Minnie     1746 0.0179
## 6  1880 F     Margaret   1578 0.0162
```

on va : ==>Filtrer le tableau pour supprimer les noms en double ==>Sélectionner un échantillon aléatoire de n nombres entre un et le nombre de lignes dans notre tableau filtré (sans remplacement - ce qui signifie qu'une fois qu'un nombre a été tiré, il ne peut plus être dessiné à nouveau) ==>on retire les lignes indexées dans le tableau filtré par les nombres aléatoires que nous avons sélectionnés ==>Renvoyer ces lignes qui seront notre table de nœuds

```
get_random_names <- function (n) {
  unique_babynames <- distinct (babynames, name, .keep_all = TRUE)
  index <- sample (1: nrow (unique_babynames), n, replace = FALSE)
  names <- unique_babynames [index,]
  names
}
```

```
nœuds<-get_random_names(9)
```

Création de la table des liens Pour créer la table de liens, on va commençer par la colonne TOTO et DADA

```
TOTO<- sample (1: nrow (nœuds), nrow (nœuds) * 2, replace = TRUE)
DADA <- sample (1: nrow (nœuds), nrow (nœuds) * 2, replace = TRUE)
```

Fusionner les vecteurs pour former une seule table

```
liens<-data.frame(TOTO, DADA)
```

on veut pas dessiner un lien entre les deux mêmes nœuds deux fois car cela peut causer des problèmes plus tard si on décide d'ajouter des attributs de lien à notre graphique.

```
links <- data.frame(TOTO,DADA) %>%
  filter(!(TOTO == DADA))
```

```
links <- unique(links[c("TOTO", "DADA")])
```

on va créer une liste avant de tracer le réseau avec ggraph

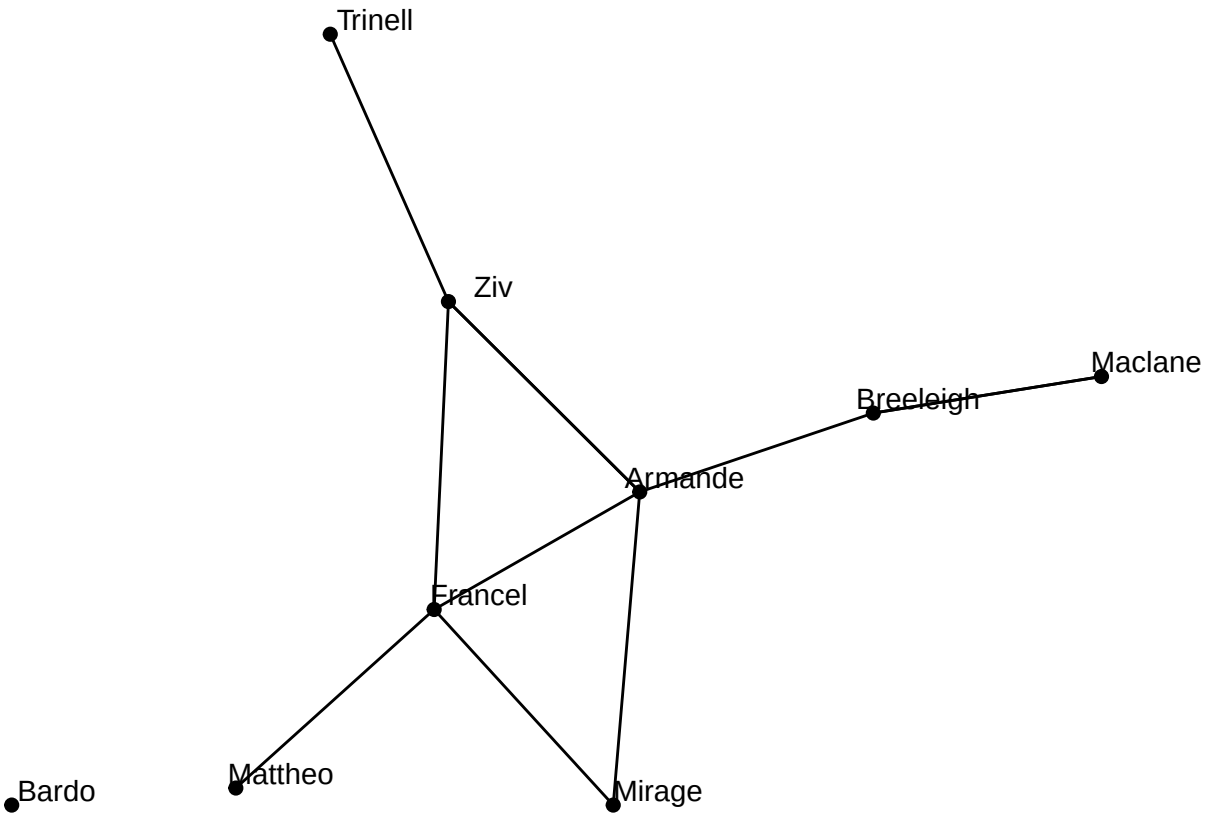
```
social_net_tbls <- tbl_graph(nodes = nœuds, edges = links, directed = FALSE)
```

on superpose le réseau dans l'ordre suivant: ==commencer par les données (l' objet tbl_graph que nous venons de créer) ==>Sélectionner la mise en page du graphique ==>spécifier les attributs ==>Ajouter les nœuds, les noms sous forme de texte ==>Modifier le thème (changer la couleur de fond)

```
social_net <- ggraph(social_net_tbls, layout = "stress") +
  geom_node_point(size = 2) +
  geom_node_text(aes(label = name), nudge_y = 0.05, nudge_x = 0.2)+
  geom_edge_link() +
  theme_void()
```

visualiser le networking

```
show(social_net)
```


Trinell
Ziv
Maclane
Breeleigh
Armande
Francel
Mattheo
Bardo
Mirage

Un réseau est un ensemble d'objets interconnectés les uns avec les autres. La visualisation d'un réseau permet donc de mettre en avant les relations et la force des liens entre ces différents objets, que nous appellerons noeuds, à partir d'une base de données, dont les informations sont moins évidentes à analyser au premier abord. Donc, nous avons trouvé une base de données sur Internet que nous allons utliser tout au long de cette présentation. La base de données est : insecta-ant-colony1-day01.edges Cette base de données provient du site : http://networkrepository.com/networks.php et contient les observations sur une journée du nombre de rencontres entre fourmis d'une même colonie.

##Visualisation du réseau

#Manipulation préalable des données

On installe d'abord les packages suivant :

```
install.packages("igraph")
install.packages("network")
install.packages("sna")
install.packages("ggraph")
install.packages("visNetwork")
install.packages("threejs")
install.packages("networkD3")
install.packages("ndtv")
```

Ensuite , on fait :

```
links <- read.csv("insecta-ant-colony1-day01.edges", header=F, as.is=T)
```

On se rend compte que notre tableau n'a qu'une colonne, nous allons donc la séparer en trois colonnes pour pouvoir l'utliser dans notre plot :

```
library(dplyr)
library(tidyr)
#Puis
df <- data.frame(links)
Liens <- df %>% separate(V1,c("Fourmi 1", "Fourmi 2", "Interactions"))
```

Nous avons donc maintenant un tableau a 3 colonnes pour nos liens, mais la dernière colonne représente des valeurs qualitatives alors qu'on veut des quantitatives :

```
library(dplyr)
MakeNum <- function(x) as.numeric(as.factor(x))
Liens <- mutate_at(Liens, 3, MakeNum)
str(Liens)
```

```
## 'data.frame':    4550 obs. of  3 variables:
##  $ Fourmi 1    : chr  "1" "1" "1" "1" ...
##  $ Fourmi 2    : chr  "2" "3" "4" "5" ...
##  $ Interactions: num  11 13 1 13 1 20 24 7 35 35 ...
```

Maintenant nous allons créer la base de données nos noeuds, qui correspondent aux fourmis.

```
Noeuds=data.frame(Fourmi=1:113)
```

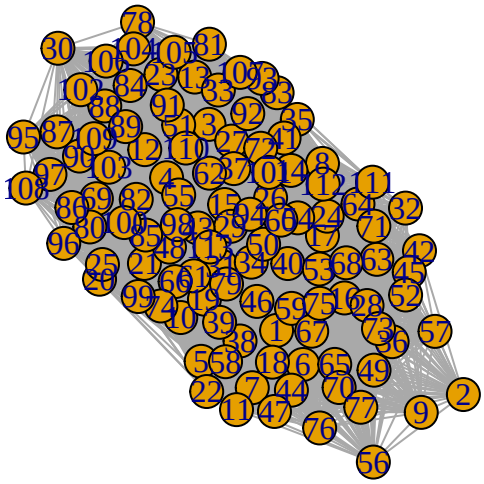
Maintenant que nous avons nos deux jeux de données, nous pouvons les rentrer dans une variable, que nous pourrons utiliser pour générer un plot:

```
library('igraph')
```

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

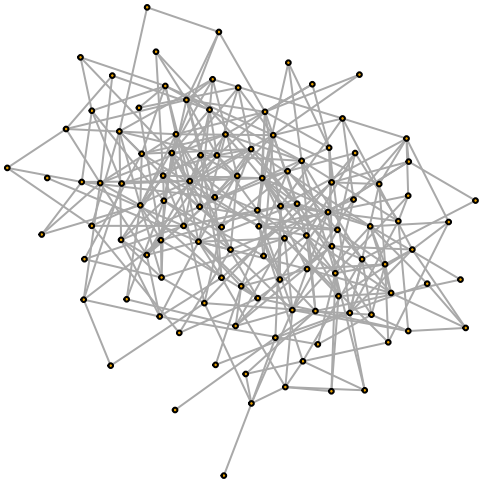
```
## The following object is masked from 'package:tidyr':
##
##     crossing
## The following objects are masked from 'package:dplyr':
##
##     as_data_frame, groups, union
## The following objects are masked from 'package:stats':
##
##     decompose, spectrum
## The following object is masked from 'package:base':
##
##     union
```

```
net <- graph_from_data_frame(d=Liens, vertices=Noeuds, directed=F)
plot(net)
```



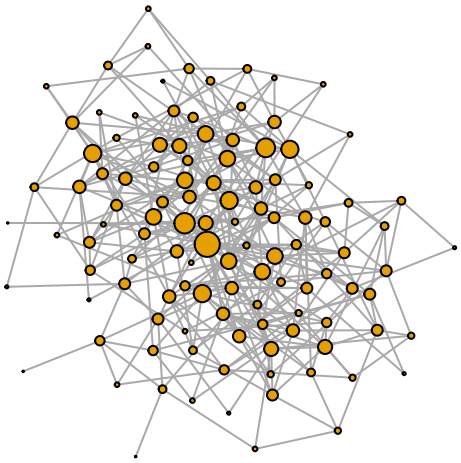
On a alors un rendu très brut et très basique de notre réseau, que nous allons affiner avec des commandes afin d'en faire ressortir les propriétés remarquables, nous allons donc supprimer tous les liens qui sont inférieurs a 55 interactions entre deux individus, nous allons donc pouvoir observer les fourmis qui se croisent le plus souvent (Onréduit la taille des noeuds et leurs noms aussi pour plus de clarté) :

```
cut.off <- 55
net.sp <- delete_edges(net, E(net)[Interactions<cut.off])
plot(net.sp, layout=layout_with_lgl, vertex.size=2, vertex.label=NA)
```



On va aussi changer la taille des noeuds suivant le nombre d'interactions avec différentes fourmis :

```
deg <- degree(net.sp, mode="all")
V(net.sp)$size <- deg*0.7
plot(net.sp, layout=layout_with_lgl,vertex.label=NA)
```



Nous ne changeons pas la taille des liens selon le nombre d'interactions, car ayant pris que les interactions supérieures à 55, la différence n'est pas très visible. Nous allons rajouter une légende grâce à ggraph pour avoir un graphique avec une taille de noeud qui correspond au nombre d'interactions avec des fourmis différentes :

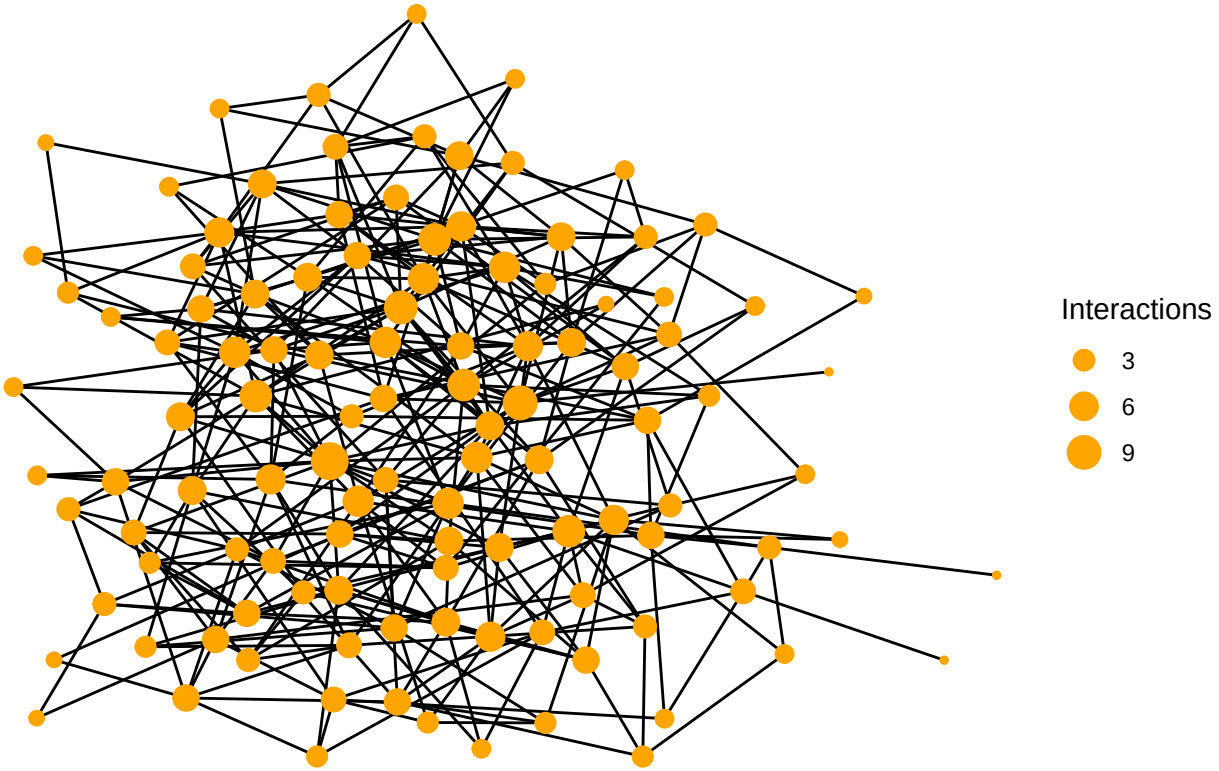
```
library(ggraph)
```

```
## Loading required package: ggplot2
```

```
library(igraph)
```

```
Interactions <-V(net.sp)$size
ggraph(net.sp, layout="kk") + geom_edge_link(color="black") + geom_node_point(color="orange", aes(size=
```

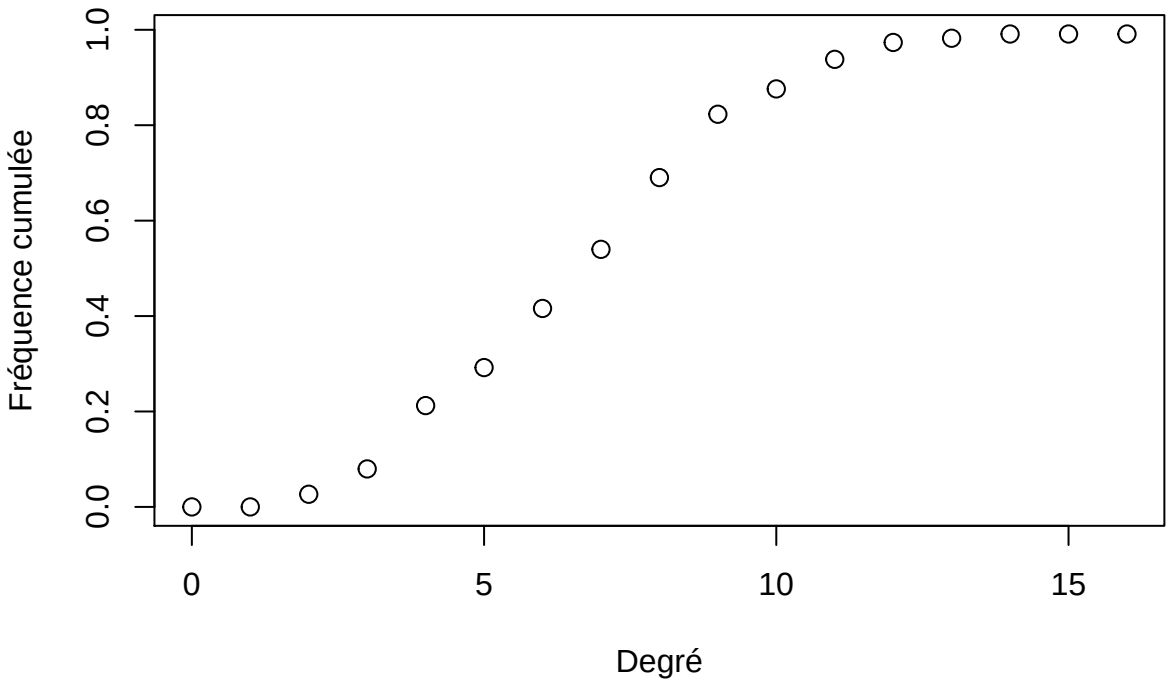
Interactions entre fourmis pendant une journée



On peut observer que les points les plus gros sont au milieu car ils se connectent au plus grand nombre de fourmis différentes.

On peut aussi visualiser notre réseau grâce a un graphe de cumul de fréquence des degrés :

```
deg.dist <- degree_distribution(net.sp, cumulative=T, mode="all")
deg.dist <- degree_distribution(net.sp, cumulative=T, mode="all")
plot( x=0:max(degree(net.sp)), y=1-deg.dist, pch=1, cex=1.2, col="black", xlab="Degré", ylab="Fréquence
```



On peut aussi mettre en avant le chemin le plus court entre deux individus, pour se faire :

```
Antpath <- shortest_paths(net.sp, from = V(net.sp)[Noeuds=="1"], to = V(net.sp)[Noeuds=="110"], output=
```

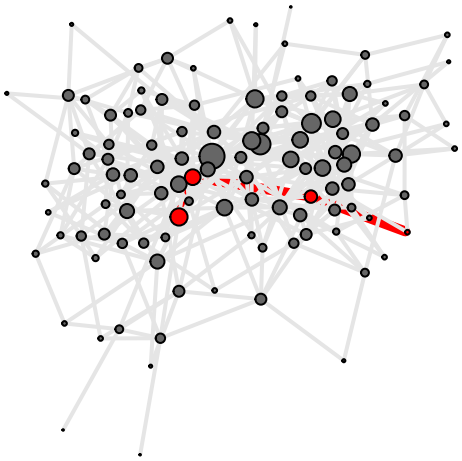
Les valeurs 1 et 110 renvoient aux individus selectionnées, donc pour voir le chemin le plus court entre d'autres individus, il suffit de changer ces deux variables. Les prochaines lignes de code vont servir à changer la couleur, la taille des liens et noeuds de notre chemin :

```
ecol <- rep("grey90", ecount(net.sp))
ecol[unlist(Antpath$epath)] <- "red"
ew <- rep(2, ecount(net.sp))
ew[unlist(Antpath$epath)] <- 5
vcol <- rep("gray40", vcount(net.sp))
vcol[unlist(Antpath$vpath)] <- "red"
```

On va ensuite génerer le plot :

```
plot(net.sp, vertex.color=vcol, edge.color=ecol,
     edge.width=ew, edge.arrow.mode=0, vertex.label=NA, main="Chemin le plus court entre l'individu X e
```

Chemin le plus court entre l'individu X et l'individu Y

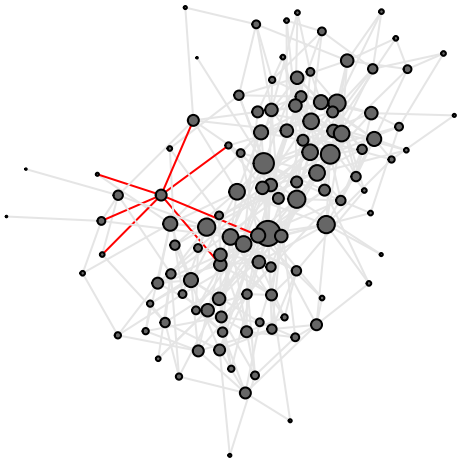


X=1 et Y=110

On peut aussi voir les chemins qui émanent d'un individu particulier, pour voir ses relations :

```
LiensLies <- incident(net.sp,  V(net.sp)[Noeuds=="100"], mode="all")
ecol <- rep("gray90", ecount(net.sp))
ecol[LiensLies] <- "red"
vcol <- rep("grey40", vcount(net.sp))
vcol[V(net)$Noeuds=="100"] <- "red"
plot(net.sp, vertex.color=vcol, edge.color=ecol, vertex.label=NA, main="Chemins émanents de l'individu
```

Chemins émanents de l'individu X



X=100

On peut ici aussi changer la variable X pour observer les interactions de chacun des individus.

```
detach("package:ggraph")
```

On a aussi moyen de créer un graph interactif sur lequel on peut cliquer sur un noeud pour ensuite afficher en surbrillance les noeuds et le liens qui lui sont attachés, mais malheureusement la fonction ne peut pas être mise sous forme de PDF, mais la fonction marche en html.

Nous venons donc de voir comment mettre un jeu de données sous forme de réseaux et aussi comment manipuler ces réseaux afin de mettre en avant différentes propriétés dans nos données, tels que l'individu qui a le plus été en relation avec d'autres individus ou mettre en avant les chemins les plus courts entre deux individus ce qui peut être très intéressant selon l'étude menée, comme par exemple en épidémiologie, et les réseaux reservent encore beaucoup de subtilitées encore inexplorées ici.

Conclusion : La visualiation de réseau à quoi ca sert?

Comme nous avons pu le voir, une visualisation en réseau sert à mettre en évidence les caractéristiques spécifiques d'une base de données. On peut choisir quoi mettre en évidence comme la force des liens, la taille des noeuds ou même tout simplement des zones de ce réseau avec de la couleur.

De fait, nous nous rendons compte que une visualisation en réseau d'une base de donnée exhaustive et peu lisible permet une représentation visuelle qui est trivialement plus efficace et simple à comprendre qu'une représentation chiffrée ou textuelle.

La principale difficulté étant de représenter correctement cette base de données à partir d'un réseau reste néanmoins un enjeu majeur. Avec de telles conditions, difficile de représenter parfaitement une base de données sans erroner ou omettre quuelques informations.

Les réseaux ont comme principale singularité de visualiser des relations sous forme de noeuds et des liens. Les liens sont les acteurs majeurs du réseau et doivent donc bien être définis. Ils peuvent indiquer une réciprocité ou non, une direction. Les relations peuvent aussi être ordonnées pour mettre en évidence des caractéristiques (liens forts ou faibles, nombre d'intermédiaires reliant 2 acteurs...).

En étant créatif sur comment représenter ces liens on peut donc mettre en avant plein de caractéristiques sur plusieurs réseaux d'un même jeu de données.

C'est ainsi que la visualisation de réseau est devenu un outil majeur dans des disciplines sociales. Mais aussi pertinente pour des disciplines telles que la visualisation chronologique, la visualisation géographique, la visualisation sociale et l'égo-réseau. Malgré tout la visualisation de ces réseaux a plus un but informatif qu'argumentatif.

Cependant ces différents réseaux aident à avoir une approche plus pédagogique d'un problème et aide à la représentation ce qui peut servir pour vulgariser des problèmes bien plus complexes aux yeux du grand public.

$si X_1, X_2, , , , , X_n$ échantillons de loi discrète on a $L(x_1, x_2, ...x_n; B) = \prod_{i=1}^{n} f(x_i)$

fonction de vraisemblance et $\hat{B} = \arg\max_{B\in(H)} L(x_1, x_2, ...x_n; B)$

la valeur du paramètre B sous laquelle ce qu'on a observé le plus probable

$$L(x_1, x_2, ...x_n; B)_{B\in(H)} = L(x_1, x_2, ...x_n; \hat{B})$$

soit (Xi)i=1,2, ,n variable aléatoire réel indépendants identiquement distribué

On teste H0 : les nœuds sont indépendants contre H1 :les nœuds son liées

⇒on test sur le paramètre d'une loi exponentielle $si X_1, X_2, , , , , X_n idd \sim \exp(B)$

avec la fonction de vraisemblance

$$L(x_1, x_2, ...x_n; B) = B^n e^{-B\sum_{i=1}^{n} x_i}$$

on a ∑Xi ~gamma(n,B) et maximal en $\hat{B} = \frac{1}{\overline{X}_n}$ $\overline{X_n} = \frac{\sum_{i=1}^{n} X_i}{n}$

si n est grand (∑Xi−nE(X)) /sqrt(nVar(X) converge en loi vers N(0,1) d'après théorème centrale limite

⇒on teste sur le paramètre d'une loi gaussienne $X_1, X_2, , , , , X_n idd \sim N(m_1, \sigma^2)$

Teste sur la moyenne à variance connue

Teste de HO:m1=m contre H1 :m1>mo

la fonction de vraisemblance $L(x1,x2, \ldots xn,m)=\prod \left(\frac{1}{\sigma\sqrt{2\pi}} EXP\left(-\frac{(Xi-m1)^2}{2\sigma^2}\right)\right)$

après une simple simplification $L(x1,x2, \ldots xn,m)=\left(\frac{1}{\sigma\sqrt{2\pi}}\right)^n \exp\left(-\frac{1}{2\sigma^2}\sum (Xi-m1)^2\right)$

donc la fonction de vraisemblance est croissante de $\overline{X_n} = \frac{\sum_{i=1}^{n} X_i}{n}$

la zone de rejet R={Xn>=t} donc sous H0 $\bar{Xn}$ ~N(m1,σ^2 /n) et sous

H1 $\bar{Xn}$ ~N(m0,σ^2 /n)

la puissance de ce teste est $P(\frac{\bar{Xn}-m}{\sqrt{\frac{\sigma^2}{n}}} > q(1-a))$=a

avec q(1-a) = la fonction réciproque du loi normale de 1-a si a<0 .05 on rejette H0

pour chaque niveau a fixé , la puissance du test tend vers 1 quand la taille de l'échantillon tend vers +infinie on dit que le test est convergent

Teste de HO:m1=m contre H1 :m1≠ m

d'une manière analogue on en déduit que $R=\{ [\bar{Xn}]<m1+q(\frac{a}{2})\frac{\sigma}{\sqrt{n}}$ ou $[\bar{Xn}]>m1-q(\frac{a}{2})\frac{\sigma}{\sqrt{n}}\}$

la zone de rejet pour ce test bilatérale

Tests sur la variance à moyenne connue

Teste de HO:σ =σo contre H1 :σ =σ 1 avec σ 1 < σo

on va passer par le rapport de vraisemblance V= $\frac{L(X1,X2,\ldots,Xn,\sigma 1)}{L(X1,X2,\ldots,Xn,\sigma 0}$

après une simplification $v=(\frac{\sigma 0}{\sigma 1})^n \exp((-\frac{1}{2\sigma^2}+\frac{1}{2\sigma^2})\sum(Xi-m)^2) >1$

V est décroissante et $Tn=\frac{1}{n}\sum(Xi-m)^2$ l'estimateur du max de vraisemblance

sous H0 nTn /σ2 suit un loi de khi-deux à paramètre n

sous H1 Tn prends des valeurs plutôt plus petit sous H0 que sous H1

et la zone de rejet $R=\{\frac{nTn}{\sigma_0^2} < F^{-1}_{X(n)}(a)\}$ donc la puissance de ce teste est

$P_{\sigma_0}(R)$=a

Teste de HO:σ =σo contre H1 :σ ≠ σ o

la zone de rejet est $R=\{\frac{nTn}{\sigma_0^2} < F^{-1}_{X(n)}(\frac{a}{2})$ ou $\frac{nTn}{\sigma_0^2} < F^{-1}_{X(n)}(1-\frac{a}{2})\}$

si on teste sur la moyenne à variance inconnue la statistique suit un loi de studente

si on teste sur la variance à moyenne inconnue la statistique suit un loi de khi-deux