Formation à l'analyse de réseaux

Statistiques descriptives

S. Donnet, F. Massol, N. Verzelen

Statistiques descriptives

- Sur la base de statistiques résumées (“métriques”) ou de classifications
- Peuvent concerner les nœuds, les arêtes ou le graphe entier
- Statistiques souvent généralisables à tous les types de graphes
- Quasiment toujours : pas de distribution attendue de la statistique (donc pas de « tables de valeurs tests à x% »)… mais testables via des hypothèses sur l’espace des graphes regardés

Plan

1. Manipulation de réseaux
2. Degrés, centralités
3. Modularité
4. Randomisations et tests

MANIPULATION DE RÉSEAUX

Prise en main (données Sophie C.)

Désigner le répertoire courant

```
setwd("C:/Massol/Enseignement/Formation réseaux/Formation Resodiv Juin 2019/Données Sophie")
```

Charger les packages utiles

```
library(igraph)
```

Récupérer le jeu de données

```
data.network<-
read.table("ssna_adj_30n_all_ind_ssloop.csv",sep=",",header=T,row.names=1)
```

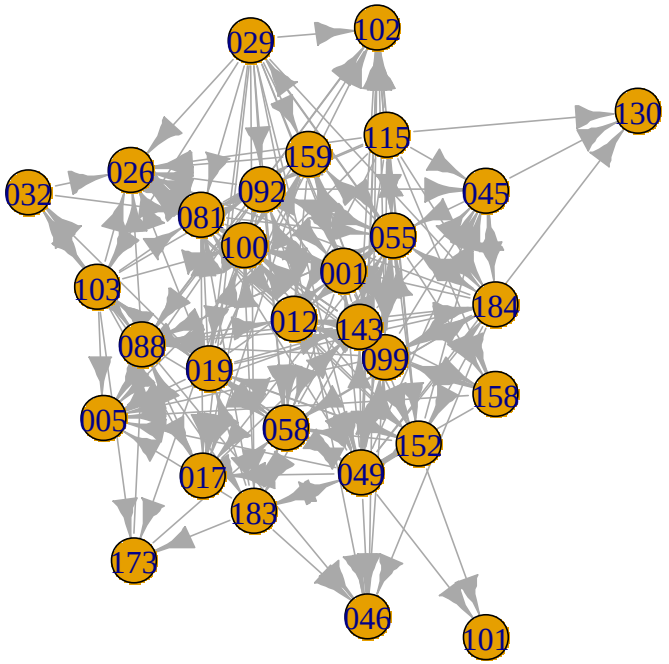
Créer le graphe dirigé pondéré

```
network<-
graph_from_adjacency_matrix(as.matrix(data.network), mode =
"directed",weighted=T)
```

Quelques définitions

- **Graphe /réseau** : ensemble de nœuds et de liens/arêtes qui les relient
- **Réseau valué** : lorsque les arêtes sont associées à des valeurs (pas forcément 1)
- **Matrice d'adjacence** : matrice représentant « qui est connecté avec qui ». Conventions :
  - a_{ij} : matrice d'adjacence binaire (non valuée)
  - c_{ij} : matrice d'adjacence valuée
  - *i* donne à *j* (convention igraph)
- **Graphe dirigé** : les arêtes ne sont pas forcément réciproques

Manipulation de réseaux



Versions binaire et symétrique de la matrice d'adjacence

```
plot(network,layout =
layout_with_kk,vertex.label=subst
r(V(network)$name,5,7))



data.network.bin<-data.network
data.network.bin[data.network.bin
>0]<-1
network.bin<-
graph_from_adjacency_matrix(as.ma
trix(data.network.bin), mode =
"directed")


network.bin.undirected<-
graph_from_adjacency_matrix(as.ma
trix(data.network.bin), mode =
"undirected")
```

Quelques définitions (suite)

- **Densité / connectance** : proportion des arêtes existantes
- **Réciprocité** : proportion de connexions mutuelles
- **Transitivité / clustering** : probabilité qu'une arête existe entre deux nœuds connectés tous deux à un troisième nœud [*mesure non dirigée*]

Calculs sur le graphe

Nombre de nœuds

```
gorder(network.bin)
```

Nombre d'arêtes

```
gsize(network.bin)
```

Calcul de la densité du graphe dirigé (et vérification)

```
edge_density(network.bin)
```

```
gsize(network.bin)/(gorder(network.bin)*(gorder(network.bin)-1))
```

Réciprocité

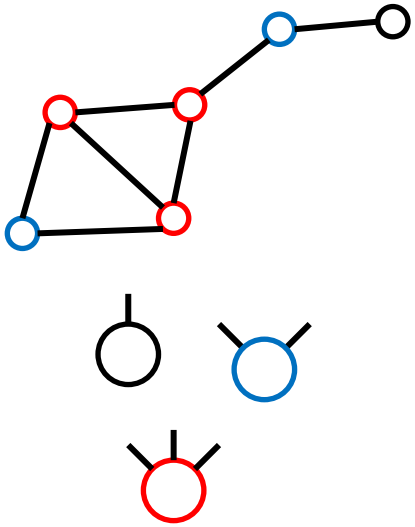
```
reciprocity(network.bin)
```

Transitivité

```
transitivity(network,type="global")
```

DEGRÉS, CENTRALITÉS

Degrés



Degré = nombre de connexions d'un nœud

$$d_i = \sum_j a_{ij}$$

[définition pour un réseau non dirigé]

Réseaux dirigés :

- degré entrant = nombre de liens entrants

$$d_i^- = \sum_j a_{ji}$$

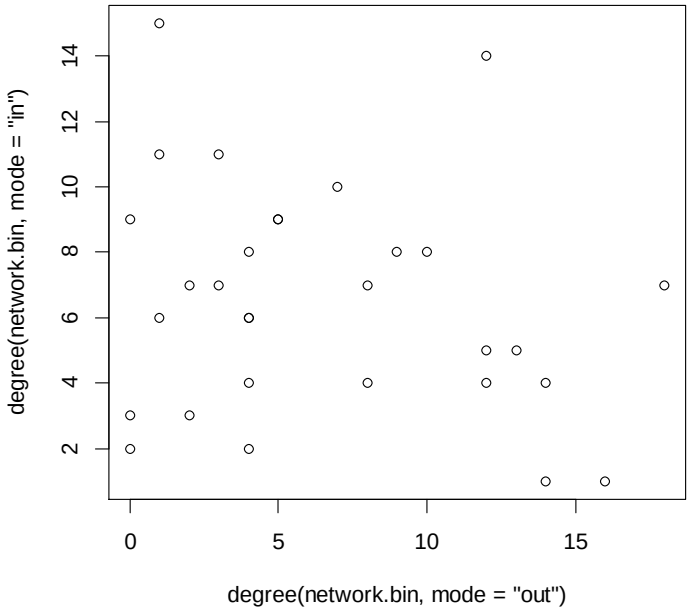
- degré sortant = nombre de liens sortants

$$d_i^+ = \sum_j a_{ij}$$

Degrés

Degrés des nœuds (tous, entrants, sortants)

Distribution des degrés (tous, entrants, sortants)




```
degree(network.bin)
degree(network.bin,mode="in")
degree(network.bin,mode="out")


degree_distribution(network.bin)
degree_distribution(network.bin,m
ode="in")
degree_distribution(network.bin,m
ode="out")



plot(degree(network.bin,mode="in"
)~degree(network.bin,mode="out"))
```

Degrés (suite)

Réseaux pondérés…?

- idem précédemment, mais avec c_{ij} au lieu de a_{ij}
- métrique *d'* (divergence KL, Blüthgen et al. 2006)

$$d_i^{+} = \sum_j \frac{c_{ij}}{C_i} \ln\left[\frac{c_{ij} C}{C_i C_j}\right]$$

$$d_i'(+) = \frac{d_i^{+} - d_{min}}{d_{max}(+) - d_{min}}$$

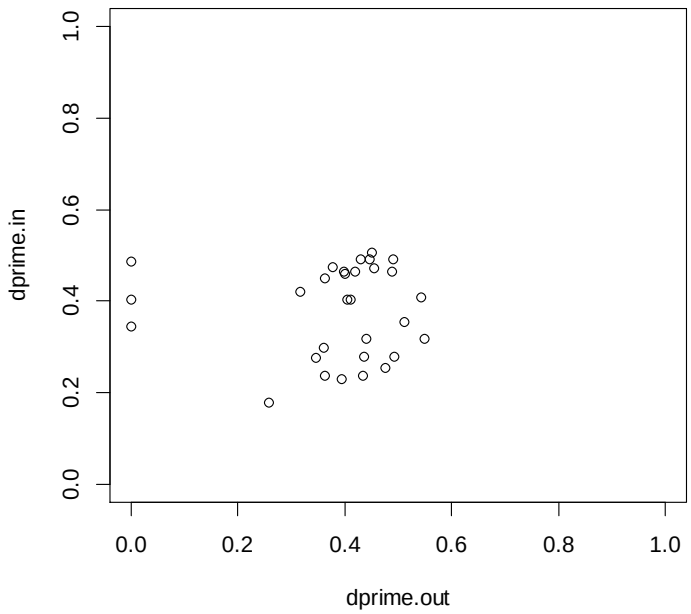
en théorie, $d_{min} = 0$, et $d_{max}(+) = \ln(C/C_i)$

Fonctions pour d’

```
pseudolog<-function(x) ifelse(x==0,0,log(x))
notzero<-function(x) ifelse(x==0,1,x)

dfun<-function(mat) {
   n<-dim(mat)[1]
   a<-sum(mat)
   ai<-notzero(apply(mat,1,sum))
   aj<-notzero(apply(mat,2,sum))
   ai_mat<-matrix(rep(ai,n),nrow=n,ncol=n,byrow=F)
   aj_mat<-matrix(rep(aj,n),nrow=n,ncol=n,byrow=T)
   elem<-(mat/ai_mat)*pseudolog(a*mat/(ai_mat*aj_mat))
   d<-apply(elem,1,sum)
   dmax<-pseudolog(a/ai)
   d/dmax
}
```

Degrés / d’

Calcul des d’ (out et in)




```
dprime.out<-dfun(data.network)

dprime.in<-dfun(t(data.network))

plot(dprime.in~dprime.out,xlim=c(
0,1),ylim=c(0,1))
```

Centralité

Définition classique (**eigen-centrality**) :

$$x_i = \frac{1}{\lambda} \sum_j a_{ij} x_j$$

avec λ la valeur propre associée à un vecteur propre positif de A

Autres définitions possibles :

– **closeness centrality** ≡ 1 / somme des distances les plus courtes à tous les autres nœuds du graphe

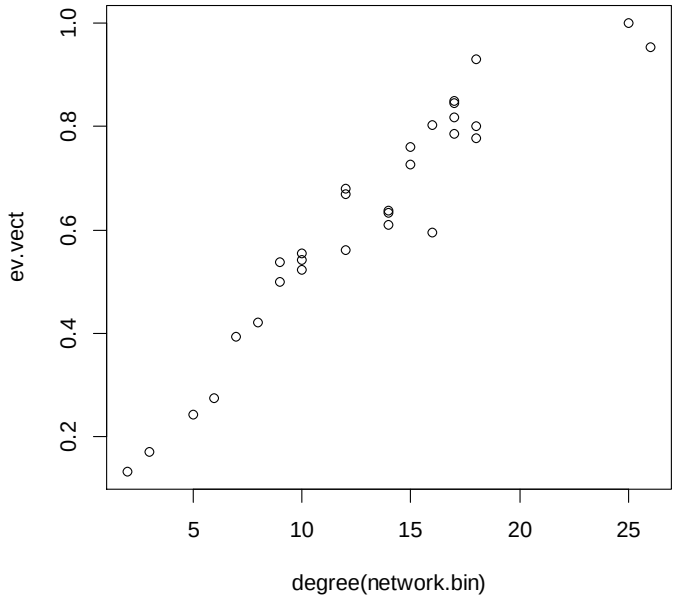
– **betweenness centrality** ≡ somme des proportions de plus courts chemins entre deux autres nœuds passant par le focal

Centralités

Eigenvector centrality

Closeness centrality

Betweenness centrality




```
ev.cent<-
eigen_centrality(network.bin.undi
rected)


clo.cent<-
closeness(network.bin.undirected,
normalized=T)


be.cent<-
betweenness(network.bin.undirecte
d,normalized=T)


ev.vect<-ev.cent$vector


plot(ev.vect
~degree(network.bin))
```

Centralité (graphes dirigés)

Définition Katz-Bonacich :

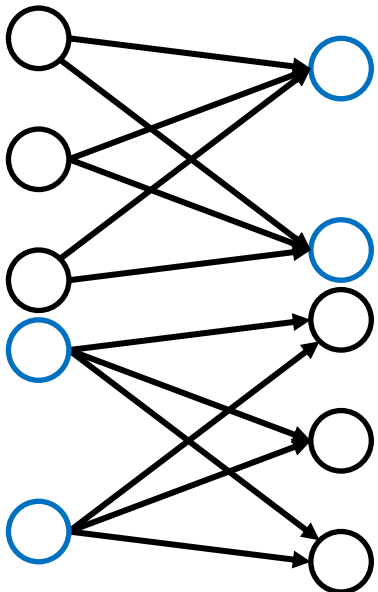
$$x_i = \alpha \sum_j a_{ij} x_j + \varepsilon$$

Problème : valeur maximale admissible de α

Autorités (in) / centres (out) [Kleinberg 1999] :

– score d'autorité : vecteur propre de $A^T.A$

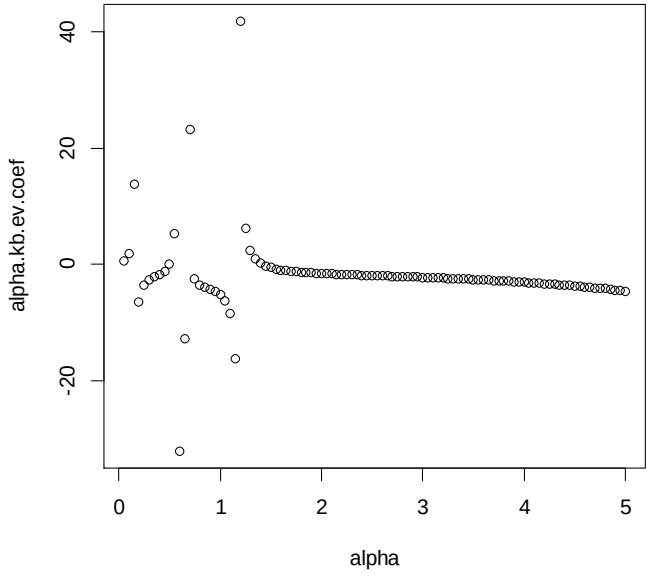
– score de centralité : vecteur propre de $A.A^T$



Centralités sur graphes dirigés

Katz-Bonacich centrality pour différentes valeurs de α

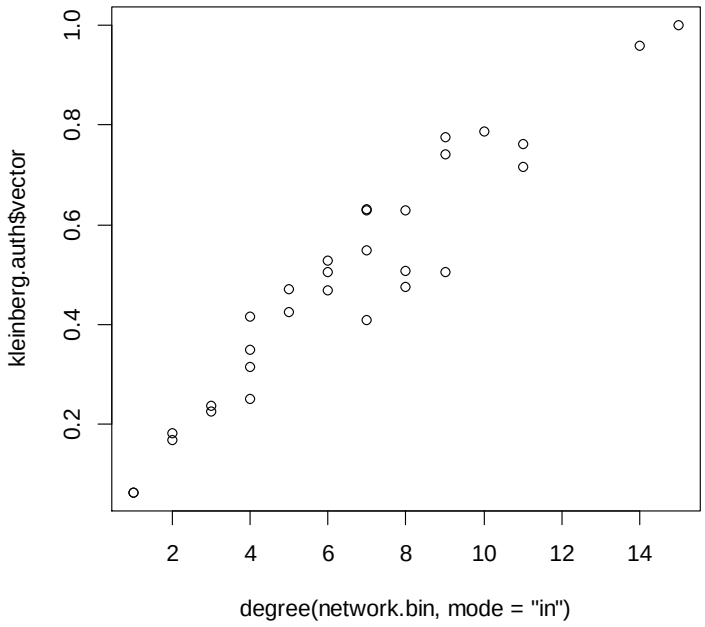
Régression avec la eigen centrality




```
kb.cent<-
sapply((1:100)/20,function(x)
alpha_centrality(network.bin,
alpha = x, exo = 1))
alpha<-(1:100)/20


regressioncoef<-function(var) {
gls(var ~ ev.vect)$coef[2]
}


alpha.kb.ev.coef<-
sapply(1:100,function(y)
regressioncoef(kb.cent[,y]))


plot(alpha.kb.ev.coef~alpha)
```

Centralités sur graphes dirigés

Scores d'autorité et de centre




```
kleinberg.auth<-
authority_score(network.bin)

kleinberg.hub<-
hub_score(network.bin)

plot(kleinberg.auth$vector~degree
(network.bin,mode="in"))
```

MODULARITÉ

Modules et modularité

Modularité (Newman)

$$Q = \frac{1}{A}\sum_{i,j}\left[a_{ij} - \frac{d_i d_j}{A}\right]\delta_{ij}$$

Principe : comparer a_{ij} à son « espérance » au vu des degrés, et ne prendre que les éléments de la somme qui correspondent à des paires ne nœuds d'un même groupe

Modules = groupes qui permettent d'obtenir la plus grande valeur de *Q*

Modules et modularité

Fonctionne pour des graphes non dirigés

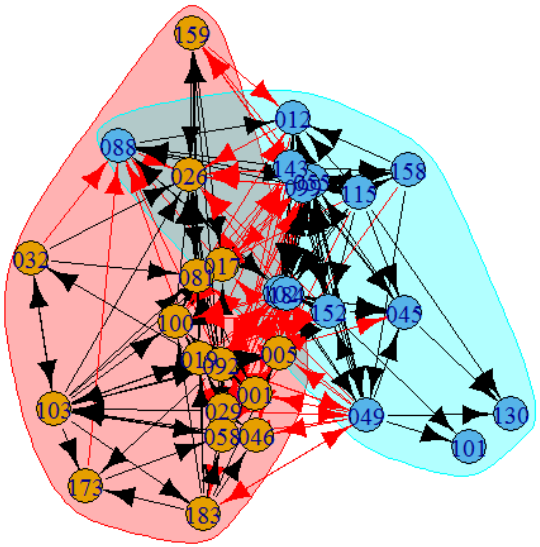
Plusieurs algorithmes (edge-betweenness, leading eigenvector, fast greedy, multilevel/louvain…)

Non adapté aux graphes dirigés

- « symétriser » le réseau
- utiliser une autre définition de la recherche de modularité

Modularité

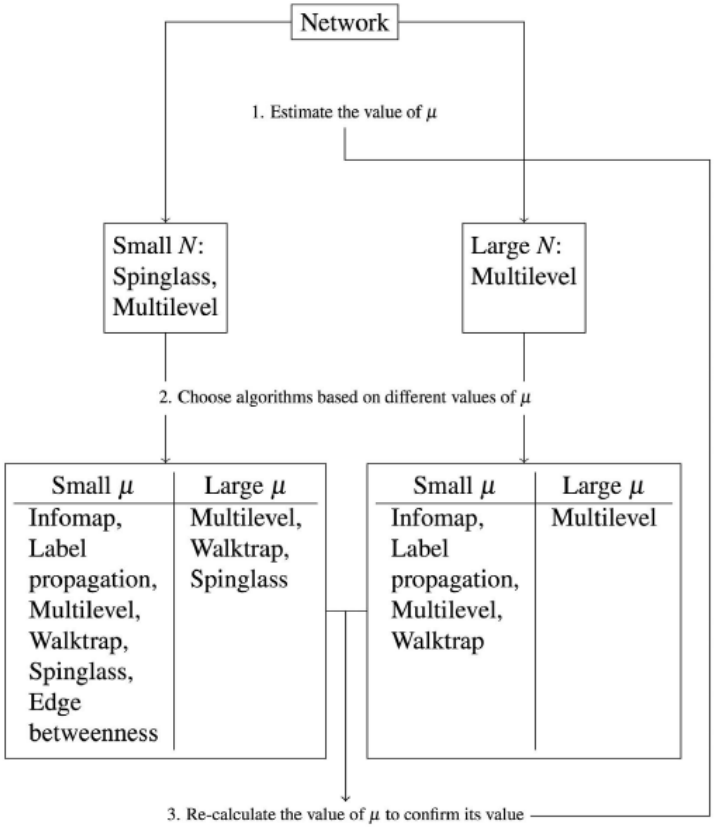
3 algorithmes (edge-betweenness, leading eigenvector, multilevel)




```
EB.mod<-
cluster_edge_betweenness(network.
bin.undirected)
LE.mod<-
cluster_leading_eigen(network.bin
.undirected)
ML.mod<-
cluster_louvain(network.bin.undir
ected)


plot(LE.mod,network.bin,layout =
layout_with_mds,vertex.label=subs
tr(V(network)$name,5,7))
```

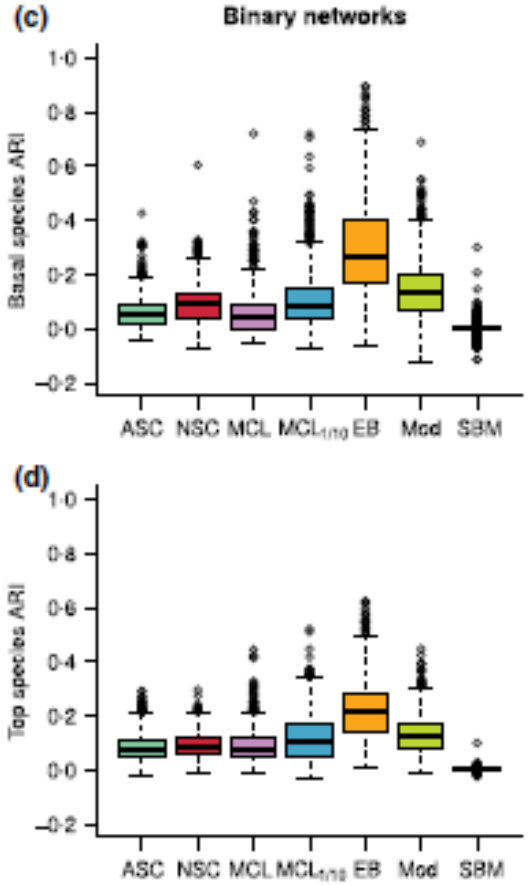
Quel algorithme choisir ?

Réseaux unipartites



Yang et al. (2016) *SciRep*

Réseaux bipartites



Leger et al. (2015) *MethEcolEvol*

Modules en réseau dirigé

Méthode `infomap` proposée par Rosvall & Bergstrom (2008)

Maps of random walks on complex networks reveal community structure

Martin Rosvall*† and Carl T. Bergstrom*‡

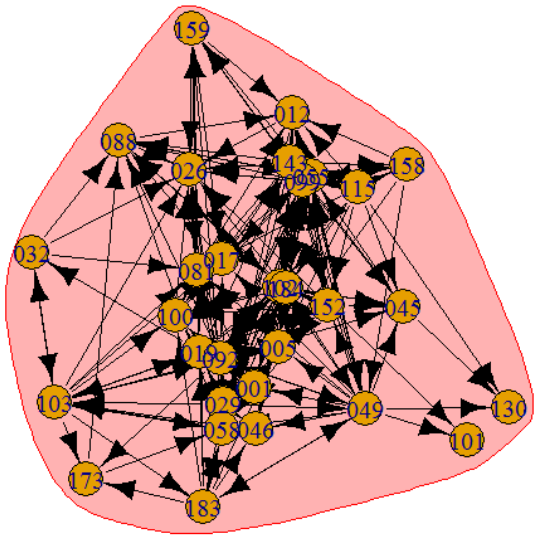
Principe : simplifier le codage d'un mouvement Brownien sur le graphe

Un module = un préfixe permettant de simplifier l'information « la particule est dans le module X »

Critère d'optimisation = minimiser le nombre de bits nécessaires pour coder une trajectoire

Modularité sur graphe dirigé

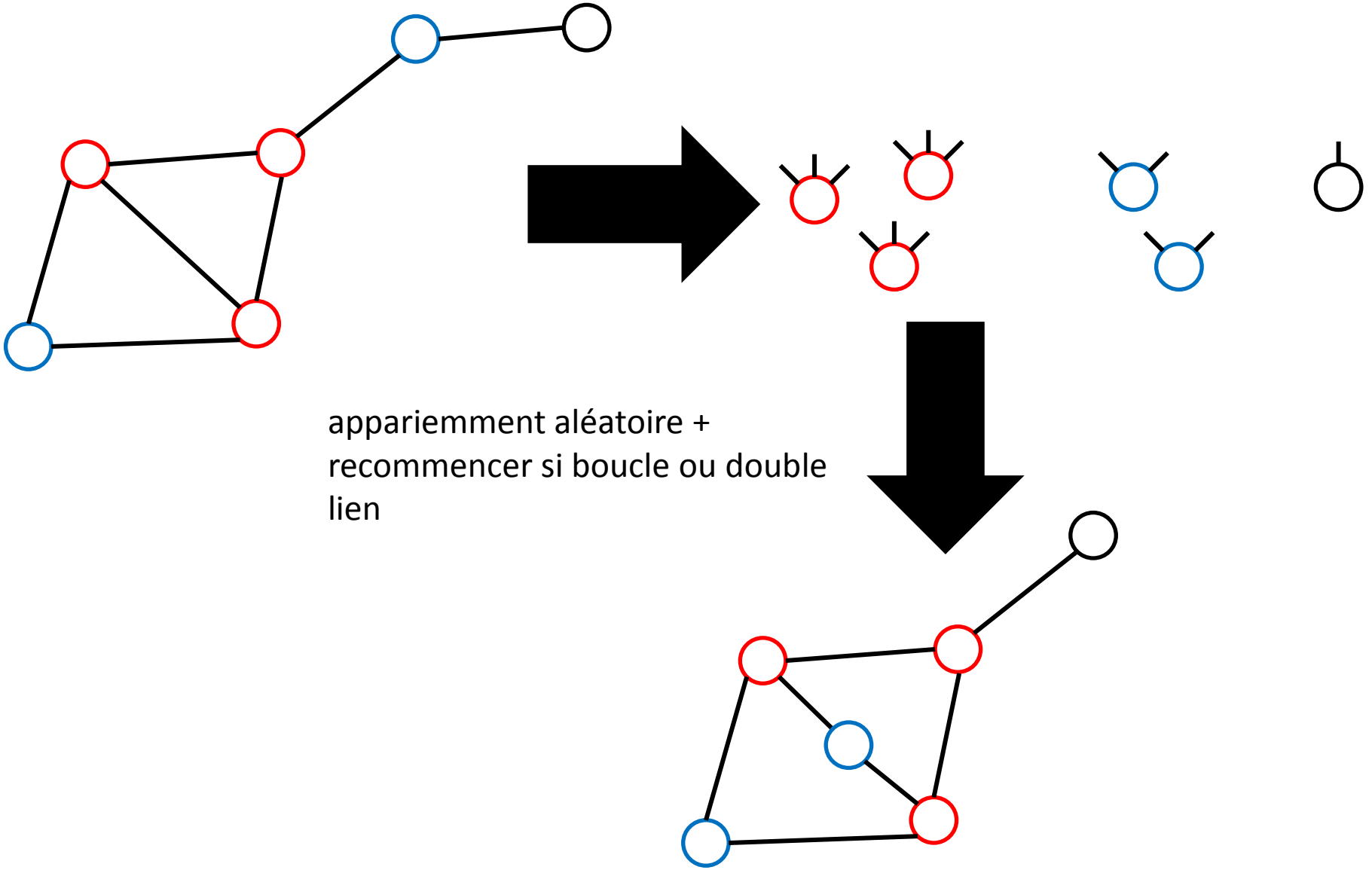
Modularité infomap




```
IM.mod<-
cluster_infomap(network.bin)


plot(IM.mod,network.bin,layout =
layout_with_mds,vertex.label=subs
tr(V(network)$name,5,7))
```

RANDOMISATION ET TESTS

Modèle de configuration



Modèle de configuration

Randomisations de graphe dirigé (non uniforme)

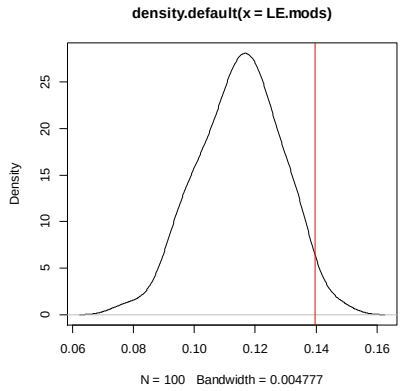
```
sample.config.directed<-
lapply(1:100,function(x)
sample_degseq(degree(network.bin,
mode="out"),
degree(network.bin,mode="in"),
method = "simple.no.multiple"))
```

Randomisations de graphe non dirigé (uniforme)

```
sample.config.undirected<-
lapply(1:100,function(x)
sample_degseq(degree(network.bin.
undirected), method = "vl"))
```

Tests

- Valeur de modularité (leading eigenvalue)
- Toutes les modularités simulées
- Distribution cumulée des modularités simulées
- p-valeur du test (one-tailed)




```
LE.mod$mod



LE.mods<-sapply(1:100,function(x)
cluster_leading_eigen(sample.conf
ig.undirected[[x]])$mod)

LE.mod.distrib<-ecdf(mods)



1-LE.mod.distrib(LE.mod$mod)

plot(density(LE.mods))
abline(v=LE.mod$mod,col="red")
```

Randomisation bipartite

Pour les réseaux bipartites, existence d'un algorithme optimal (rapide et échantillonnage uniforme) : **curveball**

ARTICLE

Received 27 Dec 2013 | Accepted 14 May 2014 | Published 11 Jun 2014

DOI: 10.1038/ncomms5114

A fast and unbiased procedure to randomize ecological binary matrices with fixed row and column totals

Giovanni Strona[1], Domenico Nappo[1], Francesco Boccacci[1], Simone Fattorini[2] & Jesus San-Miguel-Ayanz[1]

Randomisation bipartite

Charger les données d'inventaire d'espèces

Le graphe

Sa modularité

```
data.bip<-
read.table("inventory_species.csv
",sep=",",header=T,row.names=1)


bip.bin<-
graph_from_incidence_matrix(as.ma
trix(data.bip),weighted=NULL)


bip.mod<-
cluster_leading_eigen(bip.bin)$mo
d
```

Randomisation bipartite

Charger le package vegan

Générer les matrices d'incidence randomisées

Collection des modularités

Distribution cumulée des modularités

p-valeur (one-tailed test)

```
library(vegan)


sample.bip.config<-
simulate(nullmodel(data.bip,"curv
eball"),nsim=1000)


dim(sample.bip.config)


sample.mods<-
sapply(1:1000,function(x)
cluster_leading_eigen(graph_from_
incidence_matrix(as.matrix(sample
.bip.config[,,x])))$mod)


sample.mod.distrib<-
ecdf(sample.mods)


1-sample.mod.distrib(bip.mod)
```

Caveats

- Les randomisations de réseaux unipartites n'échantillonnent pas toutes uniformément l'espace des configurations

- Les randomisations des réseaux valués sont plus subtiles… (garder les zéros, leur nombre, leurs positions ?)

PAR MANQUE DE TEMPS…

Statistiques descriptives non abordées

- Sur les arêtes :
  - edge betweenness
- Sur les nœuds et dyades :
  - positions dans les motifs
  - connectivité
- Sur le graphe :
  - comptages de motifs
  - centralisation
- Autres classifications :
  - cliques
  - blocs cohésifs
- …

Références

Blüthgen, N., Menzel, F. & Blüthgen, N. (2006) Measuring specialization in species interaction networks. *BMC ecology,* **6,** 9.

Kleinberg, J. M. (1999) Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM),* ***46, 604-632.***

Leger, J.-B., Daudin, J.-J. & Vacher, C. (2015) Clustering methods differ in their ability to detect patterns in ecological networks. *Methods in Ecology and Evolution,* ***6, 474-481.***

Newman, M. E. J. (2006) Modularity and community structure in networks. *Proceedings of the National Academy of Sciences,* **103**, 8577-8582.

Rosvall, M. & Bergstrom, C. T. (2008) Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences,* **105**, 1118-1123.

Strona, G., Nappo, D., Boccacci, F., Fattorini, S. & San-Miguel-Ayanz, J. (2014) A fast and unbiased procedure to randomize ecological binary matrices with fixed row and column totals. *Nat Commun,* ***5.***

Yang, Z., Algesheimer, R. & Tessone, C. J. (2016) A Comparative Analysis of Community Detection Algorithms on Artificial Networks. *Scientific Reports,* ***6, 30750.***