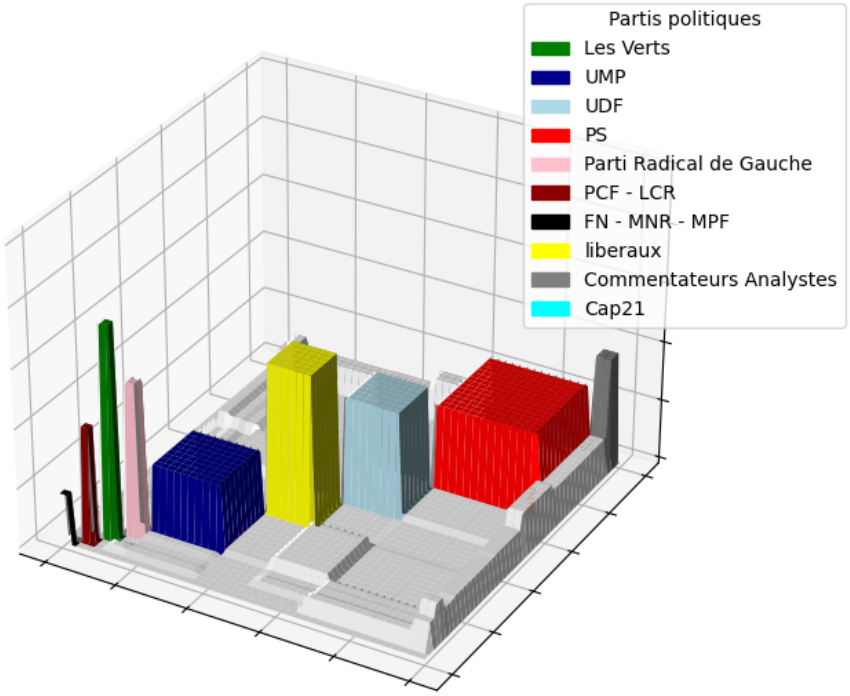
Etude de graphons

Rapport de Modal



Adélie Benhaim, Victor Mialot, Mathias Ollu

Tuteur : Pierre Latouche

Table des matières

1 Modèle du W-graphe

Ce projet porte sur l'étude des graphes et plus précisément sur l'étude des W-graphes. Dans un premier temps nous définissons le cadre théorique de ce modèle ainsi que ses propriétés élémentaires.

Les réponses au questions du sujet d'origine seront indiquées par un encadré rouge 1 suivi du numéro de la question.

1.1 Définition

Un **W-graphe** est défini comme la limite d'une suite de graphes aléatoires lorsque le nombre de sommets augmente. Un graphe est dit **dense** si le nombre d'arêtes est quadratique par rapport au nombre de sommets.

La suite de graphes aléatoires est définie comme suit pour un graphe à N sommets :

1. On commence par associer à chaque sommet une variable U_i tirée d'une loi uniforme sur l'intervalle $[0, 1]$.

$$U_i \sim U([0,1]) \quad \forall i \in [1, N]$$

2. La probabilité d'apparition d'une arête entre deux sommets i et j est alors donnée par la variable X_{ij}, qui suit une loi de Bernoulli de paramètre $W(U_i, U_j)$, où W est une fonction appelée ***graphon***.

$$X_{ij} \mid (U_i, U_j) \sim \mathrm{B}(W(U_i, U_j)) \quad \forall (i,j) \in [1,N]^2,\ i \neq j$$

Les variables $(U_i)_{i\in[1,N]}$ sont indépendantes et identiquement distribuées (iid), et les arêtes sont indépendantes conditionnellement.

1.2 Premières propriétés

1.2.1 Lois marginales

1.1 Commençons par déterminer les lois marginales des X_{ij}. X_{ij} est une variable aléatoire à valeurs dans $\{0, 1\}$, elle suit donc une loi de Bernouilli. De plus, on a :

$$P(X_{ij} = 1) = \int_0^1 \int_0^1 P(X_{ij} = 1 | U_i, U_j)\, dU_i\, dU_j = \int_0^1 \int_0^1 W(x, y)\, dx\, dy$$

Ainsi X_{ij} suit une loi de Bernoulli de paramètre $\int_0^1 \int_0^1 W(x, y)\, dx\, dy$.

1.2.2 Indépendance

1.2 Intéressons nous maintenant à l'indépendance entre les $(U_i)_{i\in[1,N]}$ connaissant la matrice d'adjacence $X = (x_{ij})_{1\leq i,j\leq N}$ et la fonction graphon W. On note f_U la densité de probabilité de U et $w = \int_0^1 \int_0^1 W(x,y)\,dx\,dy$ le paramètre de la loi de Bernouilli que suivent les X_{ij}.
Ainsi, on a la densité conditionnelle de U :

$$\begin{aligned} f_{U|X,W}(u_1,u_2,\dots,u_n,x_{11},\dots,x_{nn}) &= \frac{f_{U,X,W}(u_1,u_2,\dots,u_n,x_{11},\dots,x_{nn})}{f_{X,W}(x_{11},\dots,x_{nn})} \\ &= \frac{f_{X|U,W}(x_{11},\dots,x_{nn})f_{U,W}(u_1,u_2,\dots,u_n)}{f_{X,W}(x_{11},\dots,x_{nn})} \\ &= \frac{\prod_{i<j} W(u_i,u_j)^{x_{ij}}\big(1-W(u_i,u_j)\big)^{1-x_{ij}} 1_{[0,1]}(u_1),\dots,1_{[0,1]}(u_1)}{\prod_{i<j} w^{x_{ij}}(1-w)^{1-x_{ij}}} \\ &= \frac{\prod_{i<j} W(u_i,u_j)^{x_{ij}}\big(1-W(u_i,u_j)\big)^{1-x_{ij}}}{\prod_{i<j} w^{x_{ij}}(1-w)^{1-x_{ij}}} \end{aligned}$$

Où la troisième égalité vient du fait que U_i suit une loi uniforme sur $[0,1]$ pour tout i, on connaît donc la fonction de densité de U.

Dans la forme finale de la dernière égalité, il est clair qu'on ne peut séparer le résultat en produit chacun ne dépendant que d'un seul U_i. On n'en déduit qu'il n'y a pas d'indépendance.

1.2.3 Densité du graphe

1.7 Montrons que chaque W-graphe est dense ou vide. Supposons que notre graphe soit non vide donc que :

$$w = \int_0^1 \int_0^1 W_2(x,y)\,dx\,dy \neq 0$$

et montrons que le graphe est dense.

Le nombre moyen d'arête du graphe est (on considère les arêtes pour $i \leq j$ pour compter chaque arrête qu'une fois) :

$$\mathbb{E}[\sum_{i<j} X_{ij}] = \binom{n}{2} w$$

Pour n grand on a donc :

$$\mathbb{E}[\sum_{i<j} X_{ij}] \sim \frac{n^2}{2} w$$

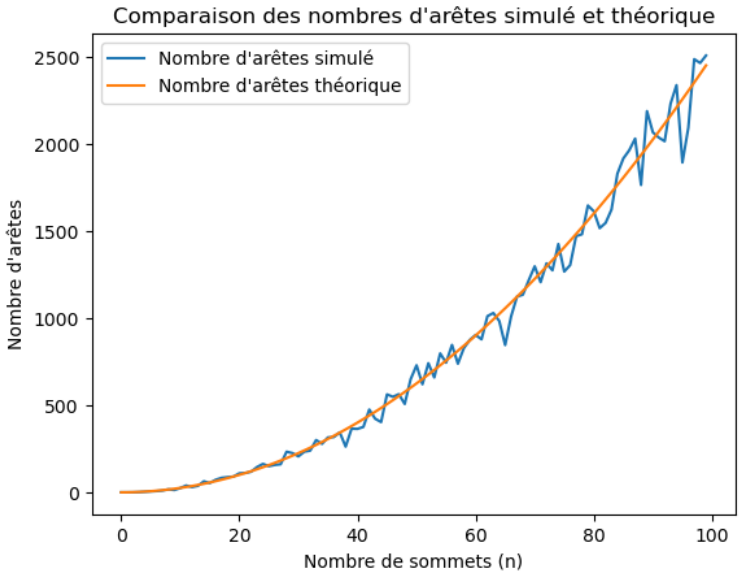


FIGURE 1 – *Simulation d'un W graphe pour le graphon* $W(x,y) = \frac{x+y}{2}$
En abscisses le nombre de sommets et en ordonnées le nombre d'arrêtes

1.8 Nous avons simulé un W-graphe pour vérifier ce résultat théorique (figure 1) : en bleu, le nombre d'arêtes de notre graphe simulé et en orange la fonction $y = \frac{x^2 w}{2}$ qui correspond au résultat théorique que nous venons de montrer.

On observe que le nombre d'arêtes empirique et l'espérance suivent une progression polynomiale très similaires. Le résultat semble donc valide.

1.2.4 Identifiabilité du modèle

1.4 On cherche ensuite à étudier les conditions d'identifiabilité du modèle. D'après 1.2.1, les arrêtes de la matrice d'adjacence issues de deux modèles de graphons associés W_1 et W_2 suivent la même loi si :

$$\int_0^1 \int_0^1 W_1(x,y)\,dx\,dy = \int_0^1 \int_0^1 W_2(x,y)\,dx\,dy \qquad (*)$$

Ainsi, pour toute une catégorie de f qui conserve la valeur de l'intégrale on pourrait avoir W_1 et $W_2 = f(W_1)$ qui détermine le même modèle.

Par exemple $W_1(x,y) = x$ et $W_2(x,y) = 1 - x$ détermine clairement le même modèle. Ainsi, il n'y a pas d'itenfiabilité possible.

Pour étudier l'identifiabilité nous étudions sous quelles conditions (*) implique $W_1 = W_2$

1.5 Il nous avait été demandé de montrer qu'il l'était sous la condition que l'intégrale de la fonction graphon soit croissante. Nous devions donc démontrer que

si deux fonctions graphons W_1 et W_2 satisfont les conditions suivantes :

— Les fonctions marginales $g_1(x) = \int_{[0,1]} W_1(x,y)\,dy$ et $g_2(x) = \int_{[0,1]} W_2(x,y)\,dy$ sont croissantes,

— $\int\int_{[0,1]^2} W_1(x,y)\,dx\,dy = \int\int_{[0,1]^2} W_2(x,y)\,dx\,dy$,

Alors cela implique que $W_1 = W_2$.

Cependant, ce résultat est faux. Nous avons trouvé un contre-exemple qui démontre que ces hypothèses ne suffisent pas pour garantir l'égalité des graphons. Considérons :

$$W_1(x,y) = xy, \quad W_2(x,y) = \sin\left(\frac{\pi}{2}x\right)\sin\left(\frac{\pi}{2}y\right)\frac{\pi^2}{16}$$

On calcule les intégrales sur $[0,1]^2$:

$$\int\int_{[0,1]^2} W_1(x,y)\,dx\,dy = \int_0^1\int_0^1 xy\,dx\,dy = \frac{1}{4}$$
$$\int\int_{[0,1]^2} W_2(x,y)\,dx\,dy = \int_0^1\int_0^1 \sin\left(\frac{\pi}{2}x\right)\sin\left(\frac{\pi}{2}y\right)\frac{\pi^2}{16}\,dx\,dy = \frac{1}{4}.$$

Les deux graphons ont donc la même intégrale double.
On calcule les fonctions marginales $g_1(x)$ et $g_2(x)$:
Pour W_1 :

$$g_1(x) = \int_0^1 W_1(x,y)\,dy = \int_0^1 xy\,dy = \frac{x}{2}$$

Pour W_2 :

$$g_2(x) = \int_0^1 W_2(x,y)\,dy = \int_0^1 \sin\left(\frac{\pi}{2}x\right)\sin\left(\frac{\pi}{2}y\right)\frac{\pi^2}{16}\,dy = \sin\left(\frac{\pi}{2}x\right)\frac{\pi}{8}$$

Les deux fonctions marginales $g_1(x)$ et $g_2(x)$ sont bien croissantes sur $[0,1]$.

Nous avons donc consulté la littérature scientifique sur le sujet, dont [1]. La démonstration de l'identifiabilité repose sur des résultats avancés issus du livre [4]. Toutefois, cette démonstration semble faire appel à des concepts au delà de notre portée, et de celle de cette étude.

Nous avons néanmoins pu identifier l'origine du blocage qui fut le nôtre. En effet, notre raisonnement était basé uniquement sur l'égalité des probabilités qu'il existe une arête entre deux sommets, ce qui implique que nous n'avons pris en compte que deux sommets à la fois.
Autrement dit, nous avons supposé que deux modèles M_1 et M_2 sont identiques si :

$$\forall i,j \in \{1,\ldots,n\}, \quad P_{M_1}(X_{ij}=1) = P_{M_2}(X_{ij}=1)$$

Cependant, l'identifiabilité dans ces travaux porte sur l'ensemble des données, c'est-à-dire sur toutes les relations entre les sommets du graphe. Par exemple, pour vérifier pleinement l'identifiabilité, il faut également considérer des configurations impliquant plusieurs sommets, et non juste 2.
Nous avons alors considéré la probabilité que pour 3 sommets $\{1, 2, 3\}$ il y ait une arrête entre 1 et 2 et entre 1 et 3. En conditionnant, on constate que la probabilité d'un tel cas est donné par :

$$\int_0^1 \int_0^1 W_1(x, y)\, dy \int_0^1 W_1(x, z)\, dz\, dx = \int_0^1 g_1(x)^2\, dx$$

En prenant en compte d'autre cas, l'identifiabilité nécessite également :

$$\int_0^1 g_1(x)^2\, dx = \int_0^1 g_2(x)^2\, dx$$

Reprenons alors notre contre exemple et calculons les intégrales au carré :

$$\int_0^1 g_1(x)^2\, dx = \int_0^1 \frac{x^2}{4}\, dx = \frac{1}{12}$$

$$\int_0^1 g_2(x)^2\, dx = \int_0^1 \frac{\pi^2}{64} \sin^2(\frac{\pi}{2}x)\, dx = \frac{\pi^2}{128}$$

Ces intégrales ne sont pas égales. Ainsi en élargissant la définition d'identifiabilité, nous constatons que notre contre exemple n'en n'est plus un.

Ainsi, nous avons compris que la définition de l'identifiabilité était cruciale et pour la suite de notre travail nous avons admis le résultat prouvé dans [4], c'est à dire que la croissance de l'intégrale des fonctions graphons suffit à l'identifiabilité.

2 Cas particuliers de W-graphe

2.1 Le modèle d'Erdös-Rényi

Le modèle d'Erdös-Rényi est un des modèles de graphe aléatoire les plus simples que l'on puisse imaginer. Dans celui ci, toutes les arêtes sont tirées de manière indépendante et identiquement distribuée (i.i.d.) selon une loi de Bernoullli de paramètre p.

$$X_{ij} \sim B(p) \quad \forall (i, j) \in [1, N],\ i \neq j$$

2.1 Ce graphe est un cas particulier de W-graphe, en effet, il suffit de prendre $W(x, y) = p$ pour tout $(x, y) \in [0, 1]^2$.

2.2 Le modèle à blocs stochastiques

2.2.1 Motivations

Cette partie se concentre sur l'étude d'un modèle plus riche que le précédent. En effet le principal problème du modèl d'Erdös-Rényi est qu'il ne permet pas des profils de connexion différents. Selon ce modèle, tout sommet aurait la même probabilité de se connecter à n'importe quelle autre sommet.

Ce constat est difficilement applicable à la réalité, ou au moins aux interactions humaines. En effet, un élève de l'école Polytechnique a par exemple bien plus de chance de se connecter à un autre élève de l'école qu'à un fermier en Alaska.

Le modèle à blocs stochastiques est un modèle qui se rapproche davantage de la réalité en prenant en compte des "communautés" et en octroyant une probabilité différente de se connecter à deux individus en fonction de leur communauté d'appartenance. Appliqué à la politique, cela implique par exemple qu'un adhérent du Parti Socialiste (PS) a une probabilité différente de se connecter à un adhérent Europe-Ecologie-Les-Verts (EELV) qu'à un adhérent Rassemblement Nationale (RN).

2.2.2 Définition

Le modèle est définit comme suit : les sommets du graphe sont répartis en K blocs. Chaque bloc k suit un modèle d'Erdös-Rényi de probabilité π_k. De plus, deux sommets issus des blocs i et j ont une probabilité μ_{ij} d'être connectés.

Formellement, pour chaque sommet $i \in [1, N]$ du graphe, un vecteur $Z_i = (Z_{i1}, Z_{i2}, ..., Z_{iK})$ est tiré à partir d'une loi multinomiale de paramètre $(1,\ \pi)$ où $\pi = (\pi_1, ..., \pi_K)$. Par définition, $\forall i \in [1, N],\ \forall k \in [1, K]$:

$$P(Z_{ik} = 1) = \pi_k \qquad \text{et} \qquad \sum_{k=1}^{K} Z_{ik} = 1,$$

.
Puis, si le sommet i est dans le bloc k et le sommet j dans le bloc l, la présence d'une arête (i, j) dans le graphe est tirée à partir d'une loi de Bernoulli de paramètre μ_{kl}.

$$X_{ij}|(Z_{ik} = 1, Z_{jl} = 1) \ \sim \ B(\mu_{kl})\ \forall (i, j) \in [1, N]^2,\ i \neq j$$

Tous les vecteurs Z_i sont tirés aléatoirement de manière iid. Les arêtes sont indépendantes conditionnellement.

2.2.3 Lien avec les W-graphes

2.3 Ce modèle est également un cas particulier de W-graphe.
Pour le montrer, on construit une fonction graphon W qui permet de répliquer le tirage du vecteur aléatoire Z_i pour chaque sommet.

Or dans le modèle du W-graphe, on tire pour tout sommet i une variable aléatoire uniforme sur $[0,1]$. On peut alors partitionner cet intervalle en K sous-intervalles dont les tailles sont les probabilités de tirer les blocs k, i.e.π_k, dans le modèle à blocs stochastiques. La fonction graphon suivante implémente cette idée visualisée par la figure 2 :

$$W(u,v) = \sum_{l_1=1}^{K}\sum_{l_2=1}^{K} \mu_{l_1,l_2} 1_{\{u\in[\sum_{k=1}^{l_1-1}\pi_k, \sum_{k=1}^{l_1}\pi_k], v\in[\sum_{k=1}^{l_2-1}\pi_k, \sum_{k=1}^{l_2}\pi_k]\}}(u,v) \tag{1}$$

FIGURE 2 – Partition de l'intervalle $[0,1]$ selon les probabilités π_k.

On a bien que X_{ij} suit une loi de Bernouilli de paramètre μ_{l_1,l_2} si et seulement si $U_i \in [\sum_{k=1}^{l_1-1}\pi_k, \sum_{k=1}^{l_1}\pi_k]$ et $U_j \in [\sum_{k=1}^{l_2-1}\pi_k, \sum_{k=1}^{l_2}\pi_k]$ ce qui survient avec probabilité :

$$\mathbb{P}(U_i \in [\sum_{k=1}^{l_1-1}\pi_k, \sum_{k=1}^{l_1}\pi_k]) = \sum_{k=1}^{l_1}\pi_k - \sum_{k=1}^{l_1-1}\pi_k = \pi_{l_1}$$

et

$$\mathbb{P}(U_j \in [\sum_{k=1}^{l_2-1}\pi_k, \sum_{k=1}^{l_2}\pi_k]) = \sum_{k=1}^{l_2}\pi_k - \sum_{k=1}^{l_2-1}\pi_k = \pi_{l_2}$$

Ainsi, on a bien une fonction graphon qui réplique le fonctionnement du modèle à blocs stochastiques.

2.2.4 Identifiabilité du modèle

Dans la continuité de la discussion en 1.2.4, on étudie ici les conditions d'identifiabilité du modèle à blocs stochastiques.

Dans ce cas particulier, la croissance de l'intégrale simple peut être atteinte en choisissant le partitionnement de $[0,1]$ nécessaire à la définition de la fonction

graphon de façon adéquate. Dans le cas du modèle à blocs stochastiques, pour une fonction graphon de type (1), on a :

$$g(u) = \int_{[0,1]} W_1(u,y)\,dy = \sum_{l_1=1}^{K} 1_{\{u\in[i_{l_1-1},i_{l_1}]\}} \sum_{l_2=1}^{K} \mu_{l_1,l_2}\pi_{l_2}$$

avec $i_k = \sum_{j=1}^{k} \pi_j$.

Pour que g soit croissant, il suffit de réarranger les i_j tel que $\sum_{l_2=1}^{K} \mu_{j,l_2}\pi_{l_2}$ soit croissant en j. La figure représente $g(u)$ implémenté avec la formule ci-dessus pour les données étudiées partie 4. On obtient bien une fonction croissante en escalier. **2.2-4**

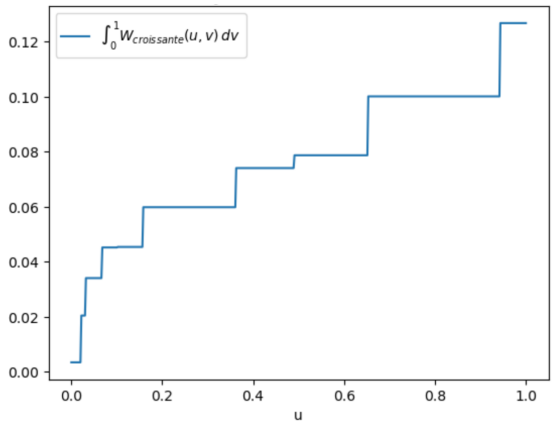


FIGURE 3 – Graphe de $\int_0^1 W_{croissante}(u,v)\,dv$ pour $u \in [0,1]$, μ, π estimés sur les données des blogs politiques (cf partie 4.1).

3 Méthode d'estimation du graphon

Maintenant que nous avons définit les W-graphes et décrit leurs propriétés saillantes, nous proposons une méthode d'estimation du modèle sur données réelles, dans le but d'étudier un réseau social en partie 4.

Cette partie se borne au cadre théorique de l'estimation des paramètres et des blocs d'un modèle à blocs stochastiques pour trois configurations différentes d'information à disposition.

On désigne par $X \in \{0,1\}^{n,n}$ la matrice d'adjacence du graphe à n sommets étudié, et par $Z \in \{0,1\}^{n,K}$ la matrice des K blocs d'appartenance.

3.1 Blocs connus

Nous envisageons ici le cas le plus simple, c'est à dire celui dans lequel les blocs d'appartenance des sommets du graphe sont connus.

Il ne reste alors qu'à estimer les probabilités d'appartenance à chaque bloc π et les probabilités de connexion inter et intra-blocs μ. Les estimateurs naturels sont les moyennes empiriques :

$$n_k = \sum_{i=1}^{n} Z_{ik} \quad \text{et} \quad \hat{\pi}_k = \frac{n_k}{n}, \quad \forall k \in \{1, \dots, K\},$$

$$\mu_{kk} = \frac{1}{n_k(n_k - 1)} \sum_{i \neq j}^{n} Z_{ik} Z_{jk} X_{ij}, \quad \forall k \in \{1, \dots, K\},$$

$$\mu_{kl} = \frac{1}{n_k n_l} \sum_{i \neq j}^{n} Z_{ik} Z_{jl} X_{ij}, \quad \forall k \neq l.$$

3.2 Nombre de blocs K connu

Dans cette partie, qui constitue le coeur de la méthode d'estimation, nous élargissons notre approche au cas dans lequel Z n'est pas connue. Nous supposons toutefois connu le nombre de blocs existant K. Ainsi notre rôle est ici d'implémenter un algorithme de *clustering* capable d'estimer les blocs, ou communautés, simplement à partir de la matrice d'adjacence X.

On se propose ici d'implémenter l'algorithme d'estimation issu de [3]. Cette partie se restreint au principe de fonctionnement de l'algorithme, en omettant les détails techniques de programmation disponibles dans le notebook associé à ce rapport.

3.2.1 Méthode d'estimation

Cette méthode d'estimation repose sur la construction d'une fonction objectif à maximiser, qui est une borne inférieure de la vraisemblance que l'on ne peut calculer directement car la vraisemblance réelle, $\mathbb{P}(Z|X)$, est inconnue.

A : Construction de la fonction objectif

La fonction objectif est une borne inférieure de la vraissemblance donnée dans [3] :

$$J(R_X) = log(L(X)) - \mathbb{KL}[R_X(.)|Pr(.|X)]$$

avec L la vraissemblance du modèle et KL la divergence de Kullback-Leibler discrète :

$$KL(P|Q) = \sum_{x \in X} P(x) \log \frac{P(x)}{Q(x)}$$

Ainsi cette quantité est une borne inférieure de la vraisemblance, KL étant positive, et nulle si et seulement si $P = Q$. De plus, elle est égale à la vraisemblance si et seulement si $R_X(.) = Pr(.|X)$, c'est à dire si et seulement si notre approximation est parfaite. Ainsi, maximiser cette quantité permet de trouver la meilleure estimation de la distribution conditionnelle de Z sachant X, et de maximiser la vraisemblance.

Pour la suite, comme dans l'article [3], on restreint R_X à adopter la forme suivante :

$$R_X(Z) = \prod_i h(Z_i; \tau_i)$$

où $\tau_i = (\tau_{i1}, \ldots, \tau_{iQ})$, et $h(\cdot; \tau)$ désigne la distribution multinomiale de paramètre τ. Ainsi, τ_{iq} mesure la probabilité que le sommet i appartienne au bloc q, c'est une approximation de Z.
La borne inférieure à minimiser prend la forme suivante :

$$\begin{aligned}
J(R_X) &= \log(L(X)) - \mathbb{KL}[R_X(.) \mid Pr(. \mid X)] \\
&= \log(L(X)) - \sum_Z R_X(Z) \log\left(\frac{R_X(Z)}{Pr(Z \mid X)}\right) \\
&= \log(L(X)) - \sum_Z R_X(Z) \log R_X(Z) + \sum_Z R_X(Z) \log Pr(Z \mid X) \\
&= -\sum_Z R_X(Z) \log R_X(Z) + \sum_Z R_X(Z) \log\left(Pr(Z \mid X) Pr(X)\right) \\
&= -\sum_Z R_X(Z) \log R_X(Z) + \sum_Z R_X(Z) \log\left(Pr(Z, X)\right)
\end{aligned}$$

D'une part, d'après la forme que l'on a contraint R_X à respecter, on a :

$$\sum_Z R_X(Z) \log R_X(Z) = \sum_i \sum_q \tau_{iq} \log \tau_{iq}$$

D'autre part :

$$\log L(Z, X) = \log L(Z) + \log L(Z|X)$$

Or, on sait que, dans un modèle à blocs stochastiques, chaque nœud a une probabilité π_q d'appartenir au bloc q si bien que :

$$log(L(Z)) = \sum_i \sum_q Z_{iq} log(\pi_q)$$

Toutefois comme nous approximons Z par τ nous obtenons :

$$log(L(Z)) = \sum_i \sum_q \tau_{iq} log(\pi_q)$$

De même, dans ce modèle, $X_{ij} \mid (Z_{ik} = 1, Z_{jl} = 1) \sim \mathcal{B}(\mu_{kl}), \quad \forall (i,j),\ i \neq j$. Donc la probabilité qu'une arête entre i et j existe est μ_{ql} pour (q,l) les blocs de (i,j), et $1 - \mu_{rf}$ pour les autres blocs (r,f). Ainsi :

$$log(L(X|Z)) = \frac{1}{2} \sum_{i \neq j} \sum_{q,l} Z_{iq} Z_{jl} log(b(X_{ij}; \mu_{ql}))$$

avec

$$b(x, \pi) = \pi^x (1 - \pi)^{1-x}$$

ce qui devient, dans notre approximation :

$$log(L(X|Z)) = \frac{1}{2} \sum_{i \neq j} \sum_{q,l} \tau_{iq} \tau_{jl} log(b(X_{ij}; \mu_{ql}))$$

Ainsi, en combinant ces équations nous obtenons la fonciton objectif :

$$\mathcal{J}(R_{\mathcal{X}}) = \sum_i \sum_q \tau_{iq} \log \pi_q + \frac{1}{2} \sum_{i \neq j} \sum_{q,\ell} \tau_{iq} \tau_{j\ell} \log b(X_{ij}; \mu_{q\ell}) - \sum_i \sum_q \tau_{iq} \log \tau_{iq} \quad (1)$$

B : Paramètres optimaux

On cherche alors à maximiser cette vraisemblance. On peut obtenir des conditions de premier ordre sur (π, μ, τ), en utilisant la méthode du lagrangien pour maximiser $J(R_X)$ sous les contraintes $\sum_q \pi_q = 1$ (i) et $\sum_q \tau_{iq} = 1$. Ces conditions garantissent que l'on obtienne bien des probabilités pour les paramètres régissant l'appartenance aux blocs et les probabilités de connexion.

Les conditions du premier ordre obtenues sont :

$$\hat{\pi_q} = \frac{1}{n} \sum_i \hat{\tau}_{iq} \quad (2)$$

$$\hat{\mu}_{ql} = \frac{\sum_{i \neq j} \hat{\tau}_{iq} \hat{\tau}_{jl} X_{ij}}{\sum_{i \neq j} \hat{\tau}_{iq} \hat{\tau}_{jl}} \quad (3)$$

Seule la condition sur τ pose problème en raison de la présence du multiplicateur de lagrange λ_i dans la solution qui ne permet que d'avoir une formule à une constante près :

$$log(\pi_q) \sum_{j \neq i} \sum_l b(X_{ij}; \mu_{ql}) - log(\hat{\tau}_{iq}) + 1 + \lambda_i = 0$$

ce qui donne :

$$\hat{\tau}_{iq} = exp(1 + \lambda_i) \pi_q \Pi_{j \neq i} \Pi_l b(X_{ij}; \mu_{ql})^{\hat{\tau}_{jl}}$$

Afin de surmonter le problème posé par $exp(1 + \lambda_i)$, on peut chercher un point fixe, en renormalisant les colonnes de τ_{iq} à chaque itération. La relation finale obtenue est ainsi :

$$\hat{\tau}_{iq} \propto \pi_q \Pi_{j \neq i} \Pi_l b(X_{ij}; \mu_{ql})^{\hat{\tau}_{jl}} \tag{4}$$

La normalisation permet d'assurer la contrainte d'obtenir une probabilité à chaque itération.

3.2.2 Implémentation de l'estimation

Dans cette sous-section, on précise l'algorithme d'estimation dont le code *python* est contenu dans le *notebook* adjoint à ce rapport. Cet algorithme repose sur 4 fonctions principales :

1. ***upate_tau*** : code la recherche de point fixe selon la relation (4), en contrôlant la durée de calcul via des restrictions sur le nombre d'itérations et la proximité entre deux itérations

Algorithm 1: Recherche de point fixe

Input: paramètres (π, μ, τ), ensemble d'adjacence X, nombre max d'itérations $nmax$, n le nombre de noeuds et K le nombre de blocs
Output: τ mis à jour
Initialiser $\tau\_new \leftarrow$ copie de τ ;
while *itérations* $\leq nmax$ **do**
  $\tau\_old \leftarrow$ copie de τ_new ;
  for i ***from*** 1 ***to*** n **do**
    for q ***from*** 1 ***to*** K **do**
      $\tau\_new[i, q] \leftarrow f(\tau, \pi, \mu, \mu)$, où f est la fonction (4) ;
    end
  end
  for i ***from*** 1 ***to*** n **do**
    normalisation de τ_new selon $\sum_q \tau\_new[i, q] = 1$;
  end
  if $||\tau\_old - \tau\_new||_2 \leq 10^{-3}$ **then**
    Sortir de la boucle **while** ;
  end
end
return τ_new

2. ***vraisemblance*** : code la formule de la vraissemblance à maximiser (1) à partir de X, μ, π, τ.

3. ***K_clust*** : code l'algorithme des K-moyennes, qui retourne K-clusters à partir de la matrice d'adjacence. Cela permet d'avoir un point de départ de l'algorithme plus efficace qu'un départ aléatoire.

4.***estimp_Qfixe*** : code l'estimation des paramètres et des blocs d'appartenance du graphe.

Algorithm 2: Estimation des blocs

Input: ensemble d'adjacence X, nombre max d'itérations $nmax$, nombre max d'itérations pour le point fixe $nmax\_ptf$, sensibilité de la convergence $emax$

Output: Blocs appartenance estimés τ

Initialisation :
$\tau \leftarrow K\_clust(K, X)$. *Point de départ des K-moyennes*
$\mu \leftarrow$ matrice vide de taille KxK
$\pi \leftarrow$ liste vide de taille K
$V\_old \leftarrow -10^{-11}$ (Initialiser vraissemblance)

while *Condition d'arrêt non vérifiée ou itérations* $\leq nmax$ **do**
  $\tau\_old \leftarrow$ copie de τ ;
  for q *from* 1 ***to*** K **do**
    $\pi[q] \leftarrow$ moyenne de la colonne q de τ (2)
  end
  for q *from* 1 ***to*** K **do**
    for l *from* 1 ***to*** K **do**
      $\mu[q, l]$ mis à jour selon la formule (3) ;
    end
  end
  $\tau = update\_tau(\pi, \mu, \tau\_old, X, nmax\_ptf)$
  V=vraiss(π, μ, τ)
  if $||V\_old - V||_2 \leq emax$ **then**
    Sortir de la boucle **while**
  end
  $V\_old \leftarrow V$
end
for i ***from*** *1* ***to*** n **do**
  $j \leftarrow argmax(\tau[i])$
  $\tau[i][q] = 0 \ \forall q$ **from** 1 **to** K
  $\tau[i][j] = 1$
end
return τ

La deuxième boucle après le while sert simplement à garder le bloc ayant la plus grande probabilité pour chaque sommet.

3.2.3 Optimisation de l'algorithme d'estimation

Afin de permettre à l'algorithme d'estimer les blocs en un temps raisonnable, nous avons optimisé deux aspects de son fonctionnement :

A : Calcul optimisé des τ à jour

La partie de l'algorithme la plus coûteuse en temps de calcul est la recherche du point fixe.

Afin d'accélérer le calcul, on évite de calculer $b()$ à chaque itération, c'est à dire qu'on ne teste pas si $X[i][j]$ est nulle. Pour cela, on implémente une liste d'adjacence à la place de la matrice d'adjacence.

Plus précisément, on calcule au début de l'algorithme d'estimation un dictionnaire d'adjacence pour lequel on associe à chaque sommet (clés) une liste de ses voisins (valeurs). Ainsi, on peut directement utiliser $\mu[q, l]$ ou $(1 - \mu[q, l])$ selon si $X[i, k] = 1$ ou 0 sans devoir le vérifier à chaque fois.

B : Choix de point de départ

Le point de départ de toute optimisation de fonction est un élément crucial de son efficacité, puisqu'il conditionne la vitesse de convergence et la qualité du résultat obtenu, c'est à dire le risque d'atteindre un "mauvais" minimum local. Dans notre cas, le point de départ est l'initialisation de la matrice τ qui estime pour chaque nœud la probabilité d'appartenance à chaque bloc.

Nous avons commencé par implémenter des points de départs aléatoires, c'est à dire des matrices (n, K) dont les lignes étaient des mesures de probabilités uniformément distribuées. Nous avons combiné cette approche à une méthode de Monte Carlo : nous faisions tourner l'algorithme entre 5 et 10 fois pour différents points de départ aléatoires, en choisissant le résultat atteignant la meilleure vraisemblance. Cette méthode a porté ses fruits, mais au prix d'un temps de calcul très long dans certains cas "malchanceux".

Nous avons donc adopté une seconde approche basée sur l'algorithme des K-moyennes, qui estime les *clusters* (ou blocs) d'appartenance de chaque sommet selon une méthode moins avancée que la nôtre, mais qui permet d'avoir un bon point de départ. Cet algorithme devant lui-même être initialisé au hasard, nous lançons à chaque fois l'algorithme K_means 50 fois et prenons le meilleur essai.

3.3 Nombre de blocs K inconnu

La méthode d'estimation présentée en 3.2 reste inadapté à l'application à des données réelles en ce qu'elle nécessite la connaissance du nombre de blocs K. En réalité, ce nombre est inconnu, et dépend uniquement des propriétés d'interconnexion des sommets. Il ne doit pas correspondre à une réalité sociale, biologique ou autre - comme on le précisera en partie 4.

3.3.1 Mesure adaptée de vraissemblance

Pour estimer les paramètres avec K inconnu, on estime les paramètres pour tout K $\in [1, N]$ avec N fixé, puis on compare la vraissemblance des modèles estimés pour chaque K selon une mesure adaptée, ***l'Integrated Classification Likelihood*** (**ICL**) :

$$ICL(X,k) = max_{\pi,\,\mu} \log(L(X, Z_1|\pi, \mu, K)) - \frac{K(K+1)}{4} \log(\frac{n(n-1)}{2}) - \frac{K-1}{2} \log(n)$$

où :

$$\log L(X, Z_1|\pi, \mu, K) = \log L(Z_1|\pi, \mu, K) + \log L(X|Z, \pi, \mu, k)$$

avec :

$$logL(Z_1|\pi, \mu, K) = \sum_{q=1}^{K} n_q \log(n_q) - n \log(n)$$

où n_q est le nombre de sommets dans la classe q.

Cette mesure permet de compenser le biais de la vraisemblance par rapport au nombre de blocs. Celle-ci a en effet tendance à être d'autant plus grande que le nombre de paramètres, c'est à dire ici de blocs, est grand. L'étude exacte de cet indice va au delà de l'objet de cette étude. On retiendra néanmoins que l'ICL permet de respecter le *rasoir d'Occam* selon lequel la solution la plus simple, i.e. la moins paramétrée, sera toujours la meilleure toutes choses égales par ailleurs.

3.3.2 Implémentation de la recherche de blocs

Il reste alors à décrire l'algorithme qui estime τ pour plusieurs nombres de blocs K différents, et retient celui qui a le meilleur ICL :

Algorithm 3: Recherche du nombre de blocs idéal

Input: nombre maximal de blocs N,ensemble d'adjacence X, nombre max d'itérations $nmax$ nombre max d'itérations pour le point fixe $nmax\_ptf$, sensibilité maximale de convergence $emax$

Output: Blocs d'appartenance estimés τ

liste_ICL $\leftarrow$ tableau vide

liste_Z $\leftarrow$ tableau vide

for $K \leftarrow 1$ ***to*** N **do**

  Faire tourner 10 fois estimp_Qfixe(X, nmax, 20, 10^{-2})

  $liste\_ICL$ **add** meilleur ICL sur les 10 essais

  $liste\_Z$ **add** la matrice Z correspondante

end

ICL $\leftarrow$ max(liste_ICL)

$K \leftarrow$ indice de ICL dans liste_ICL

$Z \leftarrow liste\_Z[K]$.

return Z,ICL, K

3.4 Test du fonctionnement de l'algorithme

Maintenant que l'on dispose d'un algorithme d'estimation fonctionnel et optimisé, on s'assure dans cette section que celui-ci fonctionne bel et bien sur des graphes simples.

On suit pour cela l'approche d'Etienne Come et Pierre Latouche dans [2]. On construit des paramètres de modèle à blocs stochastiques simples, l'idée étant de vérifier qu'à partir d'un certain niveau d'intra-connexion des blocs, l'algorithme les estime aisément. Tous les blocs sont équiprobables, ont une probabilité de connexion inter-blocs de 10^{-2}, et intra-blocs de $\beta \in [0, 1]$:

$$\pi = \left(\frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}\right)$$

et

$$\mu = \begin{bmatrix} \beta & 0.01 & \cdots & 0.01 \\ 001 & \ddots & \cdots & 0.01 \\ \vdots & & \ddots & \vdots \\ 0.01 & \cdots & 0.01 & \beta \end{bmatrix}$$

On génère ensuite pour différentes valeurs de β des graphes issus d'un modèle à blocs stochastiques pour les paramètres (π, μ).

Pour vérifier l'efficacité de notre programme, nous utilisons l'information mutuelle normalisée (IMN) des matrices Z^e estimée et Z^s simulée pour des modèles simples.

$$IMN(Z^s, Z^e) = \frac{I(Z^s, Z^e)}{max(H(Z^s), H(Z^e))}$$

où

$$I(Z^s, Z^e) = \sum_{k,l}^{K} p_{kl} log\left(\frac{p_{kl}}{p_k^s p_l^s}\right)$$

$$H(Z) = -\sum_{k}^{K} p_k log(p_k)$$

avec $p_{kl} = \frac{1}{N}\sum_{i,j}^{N} Z_{ik}^e Z_{jl}^s$ et $p_k = \frac{1}{N}\sum_{i}^{N} Z_{ik}$

Celle-ci compare les blocs estimés aux blocs réels du modèle, avec un score de 1 si ils sont identiques. D'après l'article d'Etienne Come et Pierre Latouche, nous nous attendons à une valeur de 1 à partir de $\beta = 0.45$ environ.

Afin de minimiser la variabilité des résultats pour chaque β, on génère 20 graphes différents dont on estime le modèle sous-jacent, puis on prend la moyenne des score obtenus. La figure 2 présente les résultats qui sont bien en accord avec le papier pris comme modèle : l'algorithme est efficace partir de $\beta > 0.45$.

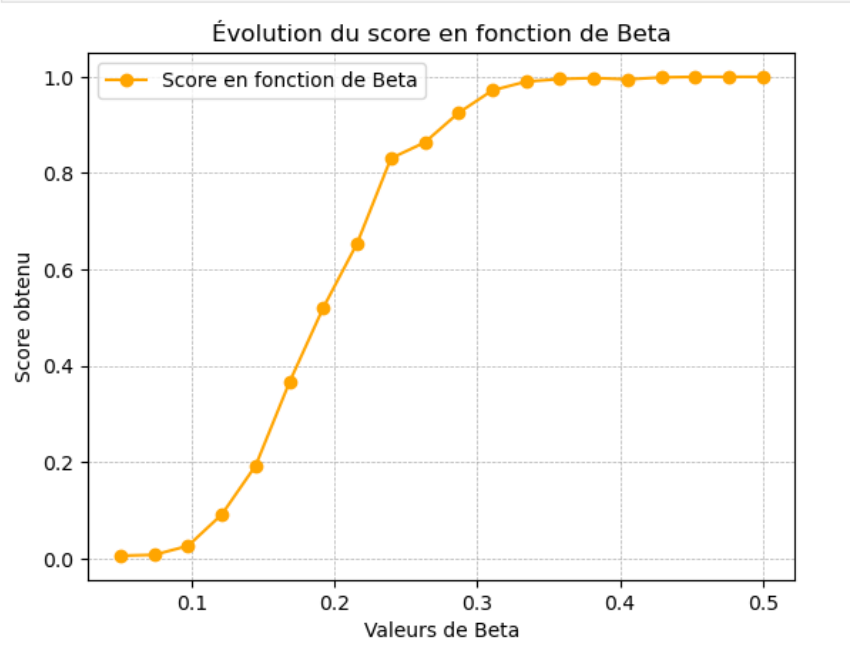


FIGURE 4 – *IMN en fonction de la valeur de β*

Notre courbe est très proche de celle obtenue dans l'article, notre algorithme semble donc efficace.

4 Application aux réseaux de blogs politiques

4.1 Présentation des données

Les données que nous avons analysées correspondent à des informations d'interconnexion de blogs politiques datant de 2007. Nous recensons 196 blogs.

Des politologues se sont accordés pour lier ces blogs à des partis politiques parmi les suivants : EELV, PS, Union pour un Mouvement Populaire (UMP), Union pour la Démocratie Française (UDF), Rassemblement National - Mouvement National Républicain - Mouvement Populaire Français (RN-MNR-MPF), Parti Communiste Français - Ligue Communiste Révolutionnaire (PCF-LCR), Parti Radical de Gauche, Libéraux, Cap21 et Commentateurs Annalystes (ce dernier ne correspond pas à un parti, mais nous allons l'étudier comme tel). Ainsi, pour chaque blog nous avons accès à son parti de rattachement.

Nous considérons alors les blogs comme les sommets d'un graphe, et les partis comme les blocs d'appartenance de ces sommets. Les arrêtes entre les blogs correspondent à la présence d'un lien hypertexte d'un blog vers un autre. Dans le cadre de cette étude, l'orientation des liens est omise. On obtient donc un graphe non orienté, et donc une matrice d'adjacence symétrique.

4.2 Analyse avec Z connu

Nous avons estimé le graphon sous jacent dans différentes cas suivant l'information à disposition. **3.1-2**

En premier lieu, nous avons supposé les blocs connus en prenant les partis associés à chaque sommet par les politiques. Il s'agit donc de retrouver les probabilités d'appartenance à chaque bloc π et les probabilités de de connexion inter et intra-blocs. Suivant les formules de 3.1, on obtient une première estimation du graphon représentée en 3 dimensions par la figure 5. Il s'agit d'une représentation graphique de la matrice $\hat{\mu}$. Pour le PS par exemple, la haueur du bloc rouge représente la probabilité des blogs PS d'êtres connectés à d'autres blogs PS, tandis que les hauteurs des autres blocs de la ligne et colonne du bloc rouges sont les probabilités de connexion à des blogs d'une autre famille politique.

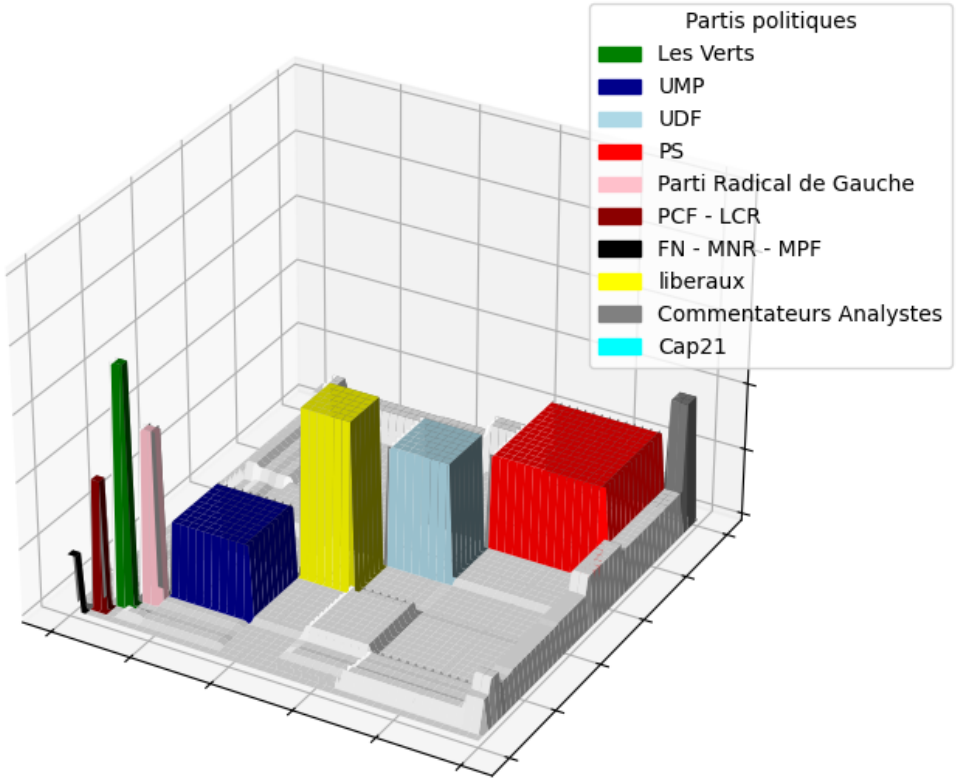


FIGURE 5 – *Représentation en 3 dimensions du graphon, données de blogs politiques*

On procède ensuite à l'affichage du graphe en utilisant l'algorithme de disposition "force-directed" de la bibliothèque *NetworkX*. Celui-ci affiche les sommets très connectés proches les uns des autres, et ceux qui le sont moins de façon éloignée, en appliquant un modèle physique d'attraction-répulsion sur l'exemple des charges électriques. Le graphe en découlant est présenté par la figure 6.

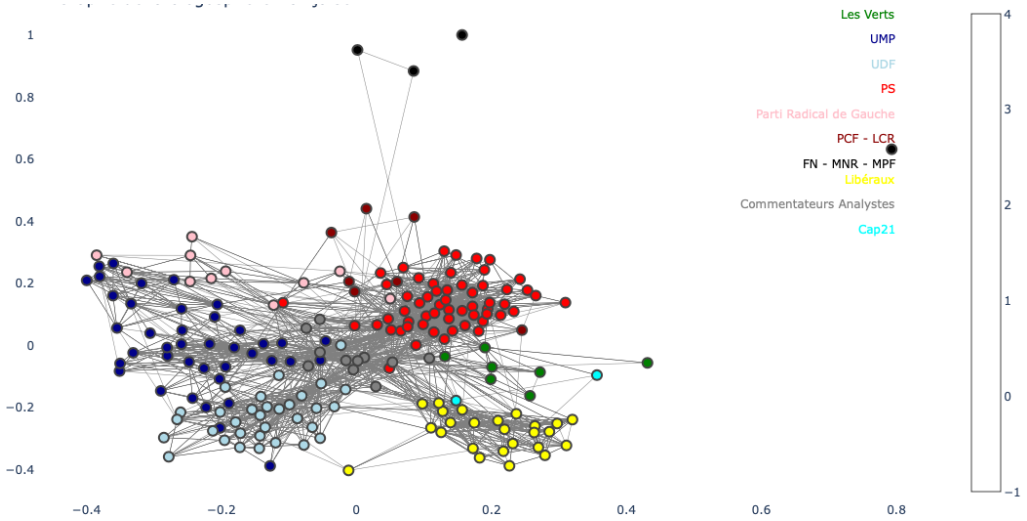


FIGURE 6 – *Graphe des blogs politiques avec la connaissance des blocs*

4.3 Analyse avec Z inconnu

L'objet de cette étude étant l'estimation des blocs, on considère dans la suite Z inconnu. L'information sur les partis d'appartenance servira pour l'analyse des résultats. Il n'y a en revanche pas de raison particulière que les blocs estimés à partir de l'information d'interconnexion des blogs soient identiques à l'information qualitative issue d'une étude de politologues.

4.3.1 Analyse à 10 partis

Cette partie est dédiée à l'estimation des blocs du graphe de blogs politiques pour $K = 10$, ce qui correspond au nombre de partis. Il s'agit donc de faire tourner l'algorithme décrit en section 3.2 qui semble fiable suite aux tests effectués en 3.4. Pour améliorer notre estimation, nous avons fait tourner 10 fois notre algorithme, et avons retenu l'essai conduisant au meilleur ICL. Cette approche est similaire à l'algorithme 3, où on ne prend qu'une seule valeur $K = 10$. On obtient ainsi 10 blocs estimés, représentés par la figure 7.

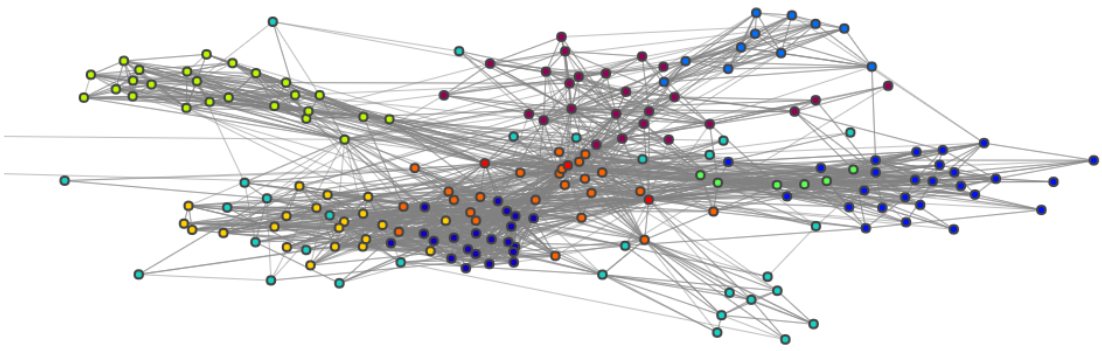


FIGURE 7 – *Graphe des blocs politiques estimés*

Il est difficile d'analyser ce graphe simplement via la visualisation de la figure 7. Cependant, à première vue, le *clustering* semble proche des blocs de partis politiques. Un tableau croisé, présenté en figure 8, permet d'analyser plus précisément le comportement de l'algorithme. Celui-ci indique pour chaque parti le nombre de sommets contenus dans chaque bloc estimé.

L'analyse approfondie du tableau croisé est contenue la section suivante. On peut néanmoins déjà constater que l'algorithme semble pertinent. La majorité des partis ont été correctement identifiés ou alors, comme dans le cas du PS ou de l'UMP, séparés en plusieurs sous partis.

L'algorithme semble particulièrement bien fonctionner pour des grands blocs très interconnectés, en adéquation avec le test en 3.4. Les plus petits groupes, dont EELV ou le PCF, semblent en effet moins clairement séparés dans l'algorithme.

On note également la présence d'un parti qu'on pourrait qualifier de *"fourre tout"* qui comprend beaucoup de blog de petits partis que l'algorithme détecte mal.

	UMP	PS	UDF	liberaux	Commentateurs Analystes	Les Verts	parti Radical de Gauche	PCF - LCR	FN - MNR - MPF	Cap21
cluster1	24	0	1	0	0	0	0	0	0	0
cluster2	11	0	0	0	0	0	0	0	0	0
cluster3	0	21	0	0	0	0	0	0	0	0
cluster4	2	10	1	0	6	2	2	1	0	0
cluster5	2	0	24	0	0	0	0	0	0	0
cluster6	0	0	6	0	0	0	0	0	0	0
cluster7	0	0	6	0	0	0	0	0	0	0
cluster8	0	0	0	24	1	0	0	0	0	0
cluster9	0	0	0	0	3	0	0	0	0	0
cluster10	1	5	0	1	1	5	9	6	4	2

FIGURE 8 – *Tableau croisé entre notre clustering en 10 partis et la réalité d'après la labélisation manuelle des experts*

4.3.2 Analyse à nombre de partis inconnu

L'étape suivante a été de déterminer un *clustering* sans connaître le nombre de blocs à trouver. Pour cela nous appliquons la méthode détaillée en 3.3.

Suivant l'algorithme 3, nous avons fait tourner notre algorithme avec K partis politiques 10 fois pour $K \in \{1, ..., 13\}$, et nous avons à chaque itération conservé le meilleur essai par rapport à l'ICL. L'évolution de l'ICL suivant K est présentée par la figure 9.

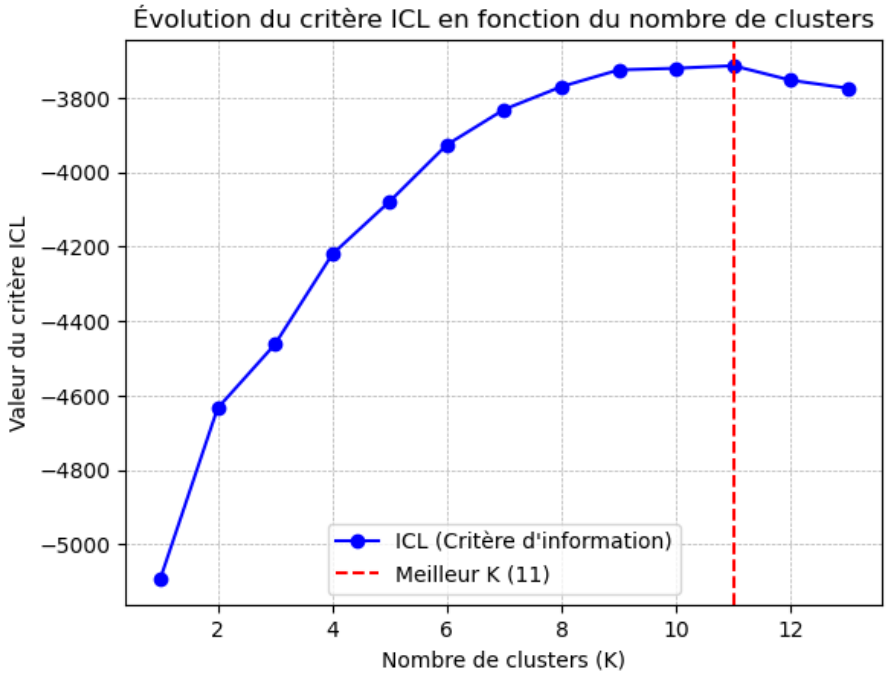


FIGURE 9 – *ICL en fonction du nombre de cluster*

On observe une forte régularité de l'ICL suivant K : il est croissant jusqu'au nombre de blocs de blocs optimal, 11, puis décroissant. On constate bien ici la différence entre blocs estimés et partis, puisque l'algorithme estime un bloc de plus qu'il n'y a de partis. L'information d'interconnexion n'équivaut pas à l'information politique qualitative de l'étude des politologues.

De même que dans le cas $K = 10$, nous avons affiché le graphe et réalisé un tableau croisé présentés par les figures 10 et 11.

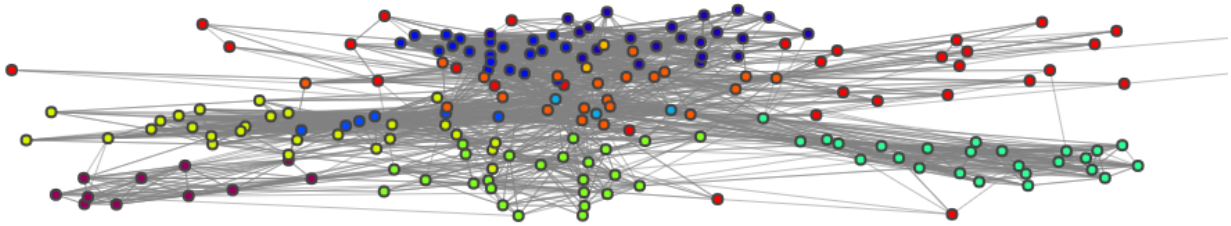


FIGURE 10 – *Graphe des blocs estimés pour* $K = 11$

	PS	UMP	UDF	liberaux	Commentateurs Analystes	Les Verts	Parti Radical de Gauche	PCF - LCR	FN - MNR - MPF	Cap21
cluster1	22	0	0	0	0	0	0	0	0	0
cluster2	20	0	0	0	0	0	0	0	0	0
cluster3	2	0	0	0	0	0	0	0	0	0
cluster4	0	24	1	0	0	0	0	0	0	0
cluster5	0	11	0	0	0	0	0	0	0	0
cluster6	0	2	24	0	1	0	0	0	0	0
cluster7	0	0	6	0	0	0	0	0	0	0
cluster8	0	0	0	24	1	0	0	0	0	0
cluster9	0	0	0	0	3	0	0	0	0	0
cluster10	8	2	1	0	5	2	2	1	0	0
cluster11	5	1	0	1	1	5	9	6	4	2

FIGURE 11 – *Tableau croisé pour* $K = 11$

4.3.3 Analyse des résultats obtenus

Cette partie est dédiée à l'analyse des résultats obtenus dans le cas $K = 11$ (4.3.2), qui constitue le meilleur résultat pour l'ICL.

La figure 11 indique que les blocs estimés correspondent à la plupart des partis politiques, certains de manière unique, d'autres sont séparés en plusieurs clusters. On observe néanmoins certaines "anomalies" :

— Le blocs estimé 3 comprend 24 UMP et 1 UDF. Comme cela nous semblait étrange, nous sommes allés regarder de plus près quel était ce blog UMP : yvesjego.typepad.com. En consultant Wikipedia, Yves Jégo est effectivement affilié à l'UDF, il semble donc y avoir une erreur d'étiquetage.
— Au contraire, on observe aussi que l'un des blocs comprend 24 UDF pour 2 UMP. En regardant de plus près, on trouve qu'ils correspondent aux blogs suivants : www.andre-santini.net et denisvinckier.hautetfort.com. De même, en consultant internet, ces deux personnalités sont en fait affiliées à l'UDF.

Ainsi, notre algorithme parvient à détecter des **erreurs d'association** réalisées par des politologues.

Nous constatons également la présence de 2 blocs qui rassemblent plusieurs partis :

— Le bloc 9 qui comprend notamment des membres du PS, EELV, Analystes ou Radicaux de gauche. Notre analyse est que ce parti est le parti que l'on pourrait désigner de "gauche modérée/centriste". On peut donc s'attendre, au vu de la proximité du positionnement de ces acteurs, à ce qu'ils soient très interconnectés.
— Le bloc 10 qui est plus difficilement identifiable, contient des blogs de tous partis, du FN au PCF. Nous pensons que ce bloc peut correspondre à un parti *"fourre tout"* pour lequel l'algorithme n'arrive pas très bien à classifier notamment car ses blogs ne forment pas assez de connexions avec d'autres blogs ou alors appartiennent à des partis trop petits.

Nous avons également voulu pousser l'analyse plus loin en comprenant les divisions que réalisait notre algorithme au sein de partis fortement représentés comme l'UMP et le PS.

L'UMP :

En étudiant en détail les noms des blogs dans les deux sous-partis de l'UMP, nous avons constaté que le sous-parti à 11 blogs est uniquement composé de blogs associés à Dominique de Villepin (son nom complet étant Dominique Galouzeau de Villepin). Voici les blogs concernés :

- libanvision.com/villepin.htm
- villepin-2007.net
- villepin.over-blog.com
- 2villepin.free.fr
- de-villepin.org
- colombespourvillepin.over-blog.com
- devillepin-renouveaugaulliste.over-blog.com
- galouzeau.hautetfort.com
- blog.villepin.free.fr
- laplumeetlepee.canalblog.com
- dominiquedevillepin.over-blog.com

La division au sein de l'UMP est donc une division entre de Villepin et le reste de l'UMP.

Le PS :

Pour le PS, en étudiant les deux grandes divisions effectuées par notre algorithme, nous n'avons pas réussi à identifier de tendance claire, car nous ne connaissons pas les personnalités politiques impliquées dans la plupart des cas. Cependant, nous avons remarqué que deux blogs PS se retrouvent seuls dans un parti distinct. Voici les blogs concernés : www.parti-socialiste.fr et annuaire.parti-socialiste.fr

Ce résultat semble cohérent. En effet, en termes de connexion avec les autres blogs, un annuaire de grand parti et le site officiel du parti auront évidemment un comportement très différent par rapport à un blog d'un élu PS. On aurait donc pu s'attendre à ce que ces deux sites soient isolés.

Références

[1] Peter Bickel and Aiyou Chen. A nonparametric view of network models and newman-girvan and other modularities. *Proceedings of the National Academy of Sciences of the United States of America*, 106 :21068–73, 11 2009.

[2] Etienne Côme and Pierre Latouche. Model selection and clustering in stochastic block models with the exact integrated complete data likelihood. *Statistical Modelling*, 15, 03 2013.

[3] Jean-Jacques Daudin, Franck Picard, and Stéphane Robin. A mixture model for random graph. *Statistics and Computing*, 18 :173–183, 06 2008.

[4] O. Kallenberg. Probabilistic symmetries and invariance principles. 01 2005.