Identification de communautés

Ziru Niu, Jiale Ning

May 23, 2021

1 Introduction

Dans le problème de l'identification de communautés, on cherche à diviser un ensemble de données en différents sous-ensembles (data clustering) de la manière que les données de chaque sous-ensemble partagent le plus de caractéristiques communes et que les données de sous-ensembles différents sont les plus dissemblables possibles. Les types de données que l'on est amené à partitionner peuvent avoir des caractéristiques très diverses et ainsi les contextes de l'utisisation du clustering sont très variées. Par exemple, dans le cadre d'études de réseaux sociaux sur des liens d'amitiés comme chez Facebook, la classification d'utilisateurs par des caractéristiques est utile dans la mise en place de la publicité ciblée.

On s'intéresse ici au problème du clustering d'un ensemble $V = \{v_1, ..., v_n\}$ d'individus intégrés dans un réseau d'amitiés : un graphe non orienté $G = (V, E)$, où E est l'ensemble des liens d'amitié (i.e. $E \subset V \times V$). L'algorithme qu'on ulitise ici est le clustering spectral dont on compare aussi la version normalisée et non-normalisée.

2 Modèle

Le modèle qu'on utilise est le stochastic block model qui prend les paramètres suivantes :

• Un ensemble de données $V = \{v_1, ..., v_n\}$

• Une partition en k classes $C_1, ..., C_k$

• Une matrice k × k symétrique P à coordonnées dans $[0, 1]$ qui représente la probabilité d'existence des arrêts

Le graphe G est obtenu de la façon telle que les différentes arêtes existent indépendemment et que pour tout $v, w \in V$, lorsque $(v, w) \in C_i \times C_j$, la probabilité qu'il existe une arête entre v et w est $P_{i,j}$. On tentera donc d'identifier les classes $C_1, ..., C_k$ par observation de la matrice d'adjacence $A = [\mathbb{1}_{(v_i,v_j)\in E}]_{i,j=1,...,n}$. On se focalise ici sur le cas où les termes diagonaux de P sont tous égaux à un nombre $p_{in} = c_{in}/n$ et où les termes non diagonaux de P sont tous égaux à un nombre $p_{out} = c_{out}/n$.

On visualise un graphe G en prenant les paramètres $n = 50, k = 2, c_{in} = 35, c_{out} = 20$ et p=0.5 (on sépare les données en deux communautés de taiile $n \times p$ et $n \times (1 - p)$ respectivement)

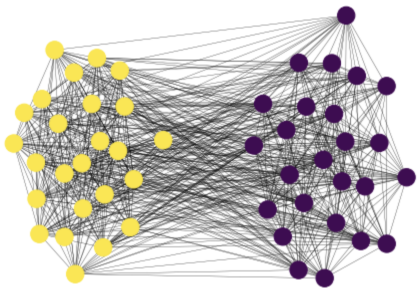

Figure 1: Graphe G

3 Spectral Clustering

On implémente l'algorithme du clustering spectral pour identifier les communautés et en plus on compare le clustering spectral non-normalisé et le clustering spectral normalisé en termes du nombre d'erreurs et on obtient

les résultats ci-dessous dans le cas où $n = 50, k = 2, c_{in} = 35, c_{out} = 20$ et p=0.5.

L'esperance du nombre d'erreurs est pour unnormalised spectral clustering: 9.11 et pour normalised spectral clustering: 1.34

L'intervalle de confiance asymptotique de probabilité de couverture 0.95 est pour unnormalised spectral clustering : [7.23,10.99] et pour normalised spectral clustering : [1.13,1.55]

On visualise aussi la distribution du nombre d'erreurs :

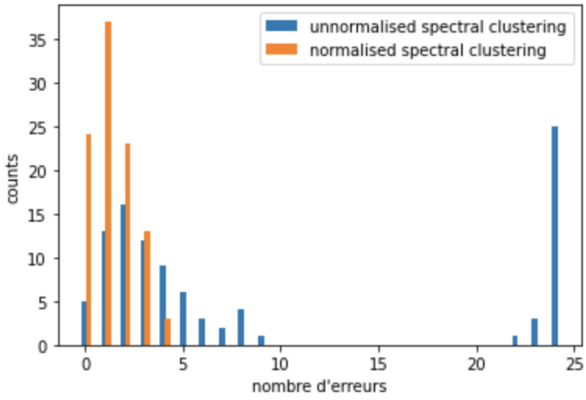

Figure 2: Distribution du nombre d'erreurs

On voit qu'il y a moins d'individus mal classés dans le clustering spectral normalisé que le clustering spectral non-normalisé. Et puis, on compare le nombre d'erreurs des deux algorithmes en fonction de différents niveaux qu'on définit comme le ratio $(c_{in} - c_{out})/\sqrt{log(n)(c_{in} + c_{out})}$ et on obtient le résultat ci-dessous.

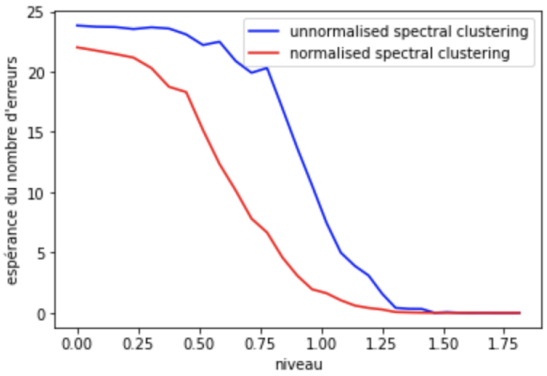

Figure 3: Nombre d'erreurs en fonction de niveaux

On voit que quand le niveau est inférieur que 1.5, le clustering spectral normalisé fonctionne mieux que le clustering spectral non-normalisé et quand le niveau dépasse 1.5, les deux algorithmes ont la même performance et ils ont bien classé tous les individus.

4 Simulation d'un évènement rare

Dans la deuxième partie, on va étudier la probabilité qu'un ensemble fixé de sommets, de cardinal faible (on commence par 1), soit mal clusterisé. On utilise le spectral clustering normalisé dans la suite.

4.1 Méthode de Monte-Carlo naïve

D'abord, on utilise la méthiode de monte-carlo pour estimer la probabilité qu'un point fixé soit mal classé avec $c_{in} = 40$, $c_{out} = 20$ et 100 expériences. Puisque dans ce cas l'évènement qu'on veut simuler a une probalité faible, on obtient les résultats ci-dessous et on voit que l'espérance est nulle. C'est-à-dire, le nombre d'expériences est trop faible pour que l'évènement apparaisse.

```
méthode de monte-carlo
avec 100 experience:
l'esperence empirique est 0.0
l'écart-type est 0.0
l'interval de confiance [0.0,0.0]
```

Figure 4: Résultats de la méthode de Monte-Carlo

4.2 L'échantillonnage d'importance

Par la suite, on pense à appliquer la méthode de l'échantillonnage d'importance et à faire un changement de probabilité. Le but du changement de proba étant d'augmenter ou maximiser la probabilité que ce point soit mal classés, on n'a du coup modifié que les p_{in} qui concernent ce point fixé telles que $p_{in} = p_{out}$. Une amélioration possible est de réduire l'écart entre p_{in} et p_{out} au lieu de mettre p_{in} égale à p_{out} tout un coup et au fur et à mesure de l'approche de p_{in} et p_{out}, on pourrait ainsi sélectionner le meilleur changement de probabilité d'après les résultats.

On rappelle la formule d'IS: Soient $X_1, \cdots, X_n$ des v.a. indépendantes de loi de Bernoulli $B(p_1), \ldots, B(p_n)$; et $Z_1, \ldots, Z_n$ des v.a. indépendantes de lois de Bernoulli $B(q_1), \ldots, B(q_n)$. Pour une fonction $g : \{0,1\}^n \to \mathbb{R}$ mesurable bornée, on a

$$\begin{aligned}
\mathbb{E}\left[g(X_1, \ldots, X_n)\right] &= \mathbb{E}\left[g(Z_1, \ldots, Z_n) \prod_{i=1}^{n} \left({}_{Z_i=0} \frac{1-p_i}{1-q_i} + 1_{Z_i=1} \frac{p_i}{q_i} \right)\right] \\
&= \left(\prod_{i=1}^{n} \frac{1-p_i}{1-q_i}\right) \mathbb{E}\left[g(Z_1, \ldots, Z_n) \prod_{i=1}^{n} \left(\frac{p_i(1-q_i)}{q_i(1-p_i)}\right)^{Z_i}\right]
\end{aligned}$$

En réalisant nos idées dans nos codes, on obtient les résultats ci-dessous sur la probabilité de mal classer ce point fixe. On voit qu'avec 100 expériences, l'écart-type qu'on obtient est relativement grand par rapport à l'espérance empirique ce qui semble anormal.

```
échantillonage importance
avec 100 experience:
l'esperence empirique est 0.00021131523545170987
l'écart-type est 0.0015515520395264176
l'interval de confiance [-9.278896429546795e-05,0.0005154194351988877]
```

Figure 5: Résultats de l'échantillonnage d'importance

Pour trouver d'où vient le problème, on a d'abord vérifié la correction du programme de l'échantillonnage d'importance qu'on a implémenté avec c_{in} et c_{out} proches, i.e, $c_{in} = 30$ et $c_{out} = 31$. En principe, quand c_{in} et c_{out} sont proches, cela implique que la probabilité de mal classer le point est grande et est autour de 0.5. Les résultats obtenus sont affichés ci-dessous. On voit que les résultats correspondent bien à ce qu'on estime, c'est-à-dire qui sont tous proches de 0.5, et on pourrait ainsi dire que l'on a correctement mis en oeuvre l'échantillonnage d'importance.

```
Méthode de monte-carlos avec 100 experience:
l'ésperence empirique est 0.49
l'écart-type est 0.4998999899979995
l'interval de confiance [0.3920196019603921,0.5879803980396079]
Échantillonnage d'importance (formule 1) avec 100 experience:
l'ésperence empirique est 0.4162269107478025
l'écart-type est 0.5181719127240194
l'interval de confiance [0.3146652158538947,0.5177886056417103]
Échantillonnage d'importance (formule 2) avec 100 experience:
l'esperence empirique est 0.4647378383791538
l'écart-type est 0.5418822408441046
l'interval de confiance [0.35852891917370927,0.5709467575845983]
```

Figure 6: Comparaison des résultats de MC et IS

Ensuite, on a fait une exploration pour voir si c'est à cause du manque d'expériences que la variance est grande. Donc, on a répété nos simulations de l'évènement rare que le point soit mal classé en augmentant le nombre

d'expériences pour les deux méthodes. D'après les résultats affichés ci-dessous, on observe que quand le nombre d'expériences augmente, l'espérance estimée devient de plus en plus stable et on pourrait déduire que la variance est petite si le nombre d'expériences est assez grande. Alors, il faut faire plus d'expériences pour réduire la variance.

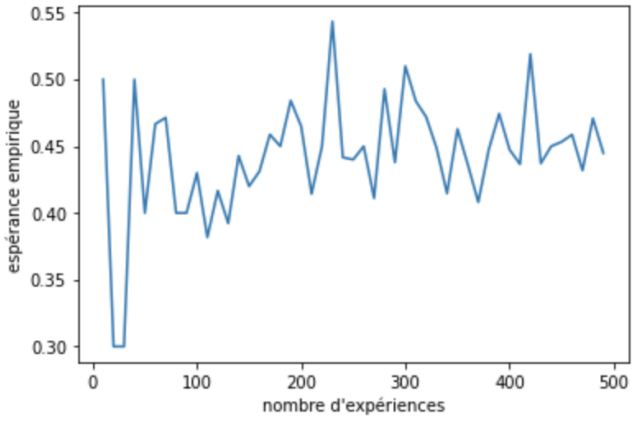

Figure 7: Évolution de l'espérance empirique en fonction du nombre d'expériences utilisant la méthode de Monte-Carlo

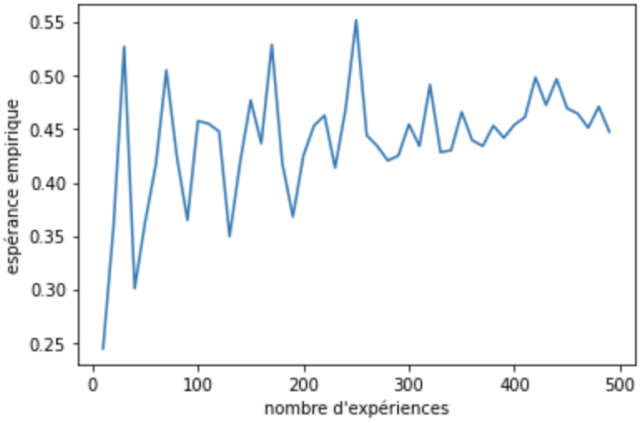

Figure 8: Évolution de l'espérance empirique en fonction du nombre d'expériences utilisant la méthode de l'échantillonnage d'importance

En revanche, ça prend environs 4 secondes avec 100 expériences ce qui est un peu lourd pour faire un nombre assez grand d'expériences tel que la variance est bien réduite. On a examiné les temps pour obtenir le vecteur U qui est ensuite utilisé pour faire le k-means clustering dans nos algorithmes du spectral clustering. On voit d'abord que k-means est un peu coûteuse puisque la classication qu'on fait avec k-means ici n'est pas compliquée étant donné que le vecteur U est un vecteur de taille n=50 d'éléments dans l'ensembel 0,1. Mais avec juste 100 expériences, k-means prend quand même 1-2s. Un autre coût qu'on analyse est celui pour obtenir le vecteur U. En effet, le clustering normalisé prend 2s en plus que le clustering non-normalisé ce qui est dû à la calculation pour normaliser une matrice, particulièrement à calculer l'inverse d'une matrice. Ces deux apects de coûts limitent ainsi la possibilité d'améliorer nos résultats en prenant un nombre d'expériences suffisamment grand.

```
time cost for calculating the matrix U in unnormalized clustering 0.6202313899993896 s
time cost for calculating the matrix U in normalized clustering 2.663491725921631 s
time cost for k-means 1.5046310424804688 s
```

Figure 9: Coûts du temps répartis dans le spectral clustering

4.3 Méthode de splitting

Dans cette partie, on considère la quesion suivante: si p_{in} et p_{out} sont relativement proches, quelle est la probabilité qu'un nombre important de points soient mal-classé ? Pour répondre à cette quetsion, on rapelle d'abord, le niveau du model défini par :

$$niveau = \frac{c_{in} - c_{out}}{\sqrt{log(n)(c_{in} + c_{out})}}$$

Comme on a vu dans la section précedante, lorsque le niveau est plus grand que 1, l'espérance du nombre d'erreurs est presque 1, cela nous inspire de fixer le niveau autour de 1 dans la suite.

Pour calculer la probabilité que plus de N points soient mal-classés, on utilise la méthode qu'on a vue dans le cours, la méthode de spliting, en utilisant la formule suivante:

$$\mathbb{P}(Erreur \geq N) = \prod_{i=1}^{N} \mathbb{P}(Erreur \geq i \,|\, Erreur \geq i-1)$$

Notons que l'on a toujours $\mathbb{P}(Erreur \geq 0) = 1$

Par cette formule, il suffit d'avoir une estimation sur chaque terme du produit à droite, et pour faire cela, on va utiliser la chaîne de markov et le théorème ergodique dans la suite. Plus précisément, on a besoin d'une chaine de markov $\{X_n\}$ où chaque membre X_n représente un graphe, qui lui-même est caraterisé par sa matrice adjacente. La matrice adjacente est une matrice stochastique dont chaque coordonné est une variable de loi bernouille du paramètre p_{in} ou p_{out} et ces variables bernouilles sont indépendantes aussi.

On note le fait que si $\{X_i\}$ et $\{Y_i\}$ sont deux chaînes indépendentes et réversibles par rapport aux mesures u et v repectivement , alors $\{(X_i, Y_i)\}$ est une chaîne de Markov réversible par rapport à la mesure $u * v$. Comme un graphe est décrit par sa matrice adjacente, elle même est rien d'autre qu'un vecteur aléatoire avec composants indépendants. On pourrait simuler chaque composant par une chaîne de Markov et puis construire une nouvelle chaîne étant la chaîne de Graphe. Du coup, pour construire la chaîne de graphe il suffit de construire des chaînes avec les conditions suivantes :

- La chaîne est définie sur l'ensemble $0, 1$
- Elle est réversible par rapport à la mesure v, qui suit une loi $Ber(q)$, où q est soit p_{in} soit p_{out}

Et on vérifie facilement que la chaîne définie par la matrice de transition ci-dessous satisfait ces conditions :

$$P = \begin{pmatrix} 1 - v(1) * (2 - \rho) & v(1) * (2 - \rho) \\ v(0) * (2 - \rho) & 1 - v(0) * (2 - \rho) \end{pmatrix}$$

où ρ est un paramètre qui joue un rôle important puisqu'il détermine à quel niveau, la valeur X_{n+1} dépend de la valeur X_n (le même rôle que le cas du gaussien dans le cours), par exemple si $\rho = 1$, alors cette chaîne est en fait une suite i.i.d, si $\rho = 2$, alors cette chaîne est une suite constante. Dans notre cas, il y a deux types de chaînes p_{in} et p_{out}, chaque type corrspond respectivement au ρ_{in} et ρ_{out}. A priori, on peut choisir ρ_{in} et ρ_{out} égaux.

On rappelle ici le théorème que l'on a vu dans le cours, qui explique la raision d'avoir une chaîne réversible:

Si $(X_i)_i$ est une chaine de Markov de transistion réversible pour v, la chaine de Markov (X_i^A) avec rejet dans A est réversible pour v_A

4.3.1 Programmation et résultats

A partir des arguments ci-dessus, on réalise un programme qui calcule la probabilité que plus de 5 points soient mal-classés, pour $p_{in} = 0.8$ et $p_{out} = 0.5$, puis pour le choix de ρ, on a choisi $\rho_{in} = \rho_{out} = 1.5$.

Dans ce cas, on a besoins de calculer 4 probabilités conditionelles, du coup 4 chaîne à simuler. Pour réaliser une chaîne, il faut aussi choisir une valeur initiale X_0, par exemple pour calculer la chaîne qui calcule la probabilité $P(err \geq 2 | err \geq 1)$, on a besoin d'avoir X_0 tel que le graphe correspondant a au moins une erreur après le clustring. Cependant, la convergence dans le théorème ergodiquene ne dépende pas du choix de X_0, du coup une fois que X_0 est trouvé, on pourrait la réutiliser dans la suite.

Une fois que les valeurs initiales des chaînes soient bien choisit, on pourra faire évoluer la chaîne autant de fois qu'on veut, et finalement on stocke toutes les probabilités qu'on a calculées dans un tableaux et le produit des élements dans ce tablaux donne la probabilité que plus de 5 points soient mal-classés. Voici les résultats pour les chaînes qui s'évoluent $N = 200$ fois.

Déjà, on voit facilement dans les deux dernières chaînes, on se trouve presque toujours dans l'état de réjet, c'est à dire, on a géneré peu de nouvelles données. En fait, on espère qu'il y a moins de rejets possibles, pour résoudre ce problème, on devrait choisir les paramètre ρ_{in} et ρ_{out} avec soin.

Si on étudie précisément le rôle de ρ, lorsque le ρ tend vers 2, il est de plus en plus difficile pour X_{n+1} de faire un changement par rapport à X_n, et on note qu'il y a 1250 variables bernouilles au total pour chaque élément dans la chaîne, un changement de niveau 0.1 sur ρ peut avoir une influence non négligeable finale sur notre chaîne. Le deuxième point à analyser, c'est de comprendre pourquoi il y a un rejet, en fait, un rejet veut dire que le prochain graphe généré dans la chaîne est mieux classé, et on sait qu'un graphe soit mal-classé ce qui est équivalent de dire qu'il y a un nombre important d'arrêts entre deux communautés et il manque des arrêts à l'intérieur de chaque une communauté.

A partir de discussions ci-dessus, on propose le changement suivant, soit $\rho_{in} = 1.5$ et $\rho_{out} = 1.95$:

```
rho_in = 1.5
rho_out = 1.5
!!!
voici les résultats du 1ème chaîne
on enregistre le processus de rejet, 1 reprensente rejet, 0 sinon
[1. 0. 0. 0. 1. 1. 0. 0. 0. 1. 0. 0. 1. 0. 0. 0. 0. 1. 1. 1. 1. 0. 1. 1.
 0. 1. 1. 1. 1. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 1. 1. 0. 1.
 0. 0. 1. 0. 0. 1. 0. 0. 1. 0. 0. 1. 0. 1. 1. 1. 0. 0. 0. 0. 0. 1. 1. 0.
 1. 0. 1. 0. 1. 1. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 0. 0. 0. 0. 1. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 1. 0. 0. 0. 1. 0. 1. 1. 0. 1. 1. 1. 0. 0.
 1. 0. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 1. 0. 0. 1. 0. 0.
 0. 1. 1. 0. 0. 0. 1. 1. 0. 0. 0. 0. 1. 1. 0. 0. 1. 0. 0. 0. 1. 0. 0. 0.
 1. 0. 1. 0. 1. 1. 1. 1. 0. 1. 0. 0. 0. 0. 1. 0. 0. 0. 1. 1. 0. 1. 1. 1.
 0. 0. 0. 0. 1. 1. 1. 1.]
le nombre de rejet est 78
voici les résultats du 2ème chaîne
on enregistre le processus de rejet, 1 reprensente rejet, 0 sinon
[0. 0. 1. 0. 0. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 0. 1. 1. 0. 0. 1. 1. 1. 1.
 1. 0. 0. 0. 1. 1. 1. 0. 1. 1. 1. 0. 1. 0. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1.
 1. 0. 1. 1. 1. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 0.
 1. 1. 1. 1. 0. 1. 1. 1. 0. 0. 1. 1. 1. 0. 1. 1. 1. 1. 0. 1. 0. 1. 1. 0.
 1. 0. 1. 0. 1. 1. 1. 0. 0. 1. 1. 0. 0. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 0. 0. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 0. 1. 1. 0. 1.
 1. 1. 1. 0. 1. 1. 1. 0. 1. 1. 0. 1. 0. 1. 1. 0. 1. 1. 1. 1. 1. 0. 1. 1.
 1. 1. 1. 1. 1. 1. 0. 0. 0. 1. 1. 1. 1. 0. 1. 1. 0. 1. 1. 1. 1. 1. 0. 1.
 1. 1. 1. 0. 1. 1. 1. 1.]
le nombre de rejet est 147
voici les résultats du 3ème chaîne
on enregistre le processus de rejet, 1 reprensente rejet, 0 sinon
[0. 0. 0. 1. 1. 1. 1. 0. 0. 1. 0. 0. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1.
 1. 1. 0. 1. 1. 1. 1. 0. 1. 1. 1. 0. 1. 0. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 0. 0. 0. 1. 1. 0. 1. 1. 0. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 0. 1. 1. 0. 1. 1. 0. 1. 1. 1. 1. 1.
 1. 1. 1. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 0. 1.
 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 0. 0. 1. 1. 0. 1. 1.]
le nombre de rejet est 167
voici les résultats du 4ème chaîne
on enregistre le processus de rejet, 1 reprensente rejet, 0 sinon
[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 0. 1. 1. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 0. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 0. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1.
 1. 0. 0. 1. 1. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 0. 1.
 0. 0. 1. 1. 1. 1. 1. 1.]
le nombre de rejet est 180
```

```
rho_in = 1.5
rho_out = 1.95
!!!
voici les résultats du 1ème chaîne
on enregistre le processus de rejet, 1 reprensente rejet, 0 sinon
[1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 0. 1. 0. 0. 1. 1. 1. 0. 1. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 1. 1. 0. 0. 1. 0. 1. 0. 0.
 1. 0. 0. 1. 1. 1. 0. 1. 0. 0. 0. 0. 1. 1. 0. 1. 0. 1. 1. 1. 0. 1. 1. 0.
 0. 1. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 1. 0. 0. 1. 0. 1. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 1. 1.
 0. 1. 0. 1. 1. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 1. 1. 1. 1. 0. 1. 0.
 1. 1. 1. 0. 1. 0. 0. 0. 1. 0. 1. 0. 0. 0. 0. 0. 0. 1. 1. 0. 0. 1. 0. 0.
 0. 0. 0. 0. 0. 0. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 0. 1. 0. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 0. 1. 0. 1.]
le nombre de rejet est 83
voici les résultats du 2ème chaîne
on enregistre le processus de rejet, 1 reprensente rejet, 0 sinon
[1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0.
 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 0. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 0. 1. 1.
 1. 1. 0. 0. 0. 0. 0. 1. 1. 0. 0. 1. 0. 0. 1. 1. 1. 0. 0. 1. 0. 1. 0. 0.
 0. 0. 1. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 0. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 0. 0. 0. 0. 1. 0.
 0. 0. 0. 0. 1. 1. 1. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 0. 1.]
le nombre de rejet est 149
voici les résultats du 3ème chaîne
on enregistre le processus de rejet, 1 reprensente rejet, 0 sinon
[0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 0. 1. 1.
 1. 1. 1. 1. 1. 1. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 0. 1. 1. 1. 1. 1. 0. 1. 0. 1. 1. 1. 0. 0. 1. 1. 1. 1. 1. 1. 0.
 0. 0. 0. 0. 1. 1. 1. 0. 1. 0. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1.
 0. 1. 1. 0. 0. 1. 0. 1. 1. 1. 0. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1.]
le nombre de rejet est 171
voici les résultats du 4ème chaîne
on enregistre le processus de rejet, 1 reprensente rejet, 0 sinon
[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1.
 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 0. 0. 1. 1. 0. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 0. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1.
 1. 1. 1. 1. 1. 1. 1. 1.]
le nombre de rejet est 187
```

Figure 10: Une comparaison

Par la figure ci-dessus, on voit que ce changement n'a pas beaucoup aidé, voire aller dans un sens contraire, mais on voit que, en tout cas, tout se passe bien jusqu'au moment où on vise à calculer $P(err \geq m | err \geq m-1)$ pour un m élevé, c'est à dire qu'il y a de plus en plus de rejets lorsque m augumente. Pour expliquer cela, on a des explications possibles, soit la probabilité $P(err \geq m | err \geq m-1)$ elle-même est très petite, soit notre chaîne n'est pas très efficace.

On a une approche simple pour vérifier si la chaîne est efficace ou pas. En fait, lorsque $\rho_{in} = \rho_{out} = 1$, cette chaîne est une suite i.i.d. et dans ce cas, si le processus de rejet n'a pas beaucoup de différences avec les deux résultats ci-dessus, on est sûr que la chaîne n'est pas efficace car une suite i.i.d est aussi performante. Et malheureusement, c'est exactement le cas dans notre cas.

Du coup, le problème est plustôt la chaîne est un peu trop simple et il faut construire une chaîne plus efficace pour ce problème.

Malgré tout, on va donner un résultat par notre approche,

```
voici les résultats du 1ème chaîne
on enregistre le processus de rejet, 1 reprensente rejet, 0 sinon
[0. 0. 1. 0. 0. 0. 0. 0. 0. 1. 0. 0. 1. 1. 0. 1. 0. 0. 0. 0. 1. 0. 1. 0.
 1. 0. 0. 1. 0. 1. 0. 0. 0. 1. 1. 0. 1. 0. 0. 1. 1. 0. 0. 0. 0. 1. 0. 0.
 0. 0. 0. 0. 0. 1. 1. 0. 0. 0. 1. 0. 0. 0. 0. 1. 0. 1. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 1. 0. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 1. 1. 0. 0. 0. 1. 0. 0. 1. 0. 0. 0. 0. 0. 0. 1. 0. 1. 0. 1. 1. 1.
 0. 0. 1. 1. 0. 1. 0. 0. 0. 1. 1. 0. 1. 1. 1. 1. 0. 0. 1. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 1. 1. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 1. 0. 1. 1. 0.
 1. 0. 0. 0. 1. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 1. 0. 1. 0. 0. 1. 0. 0. 1. 0.
 0. 0. 1. 0. 0. 0. 0. 1.]
le nombre de rejet est 60
voici les résultats du 2ème chaîne
on enregistre le processus de rejet, 1 reprensente rejet, 0 sinon
[1. 1. 1. 1. 0. 1. 0. 0. 0. 0. 1. 0. 1. 1. 0. 0. 1. 1. 1. 1. 0. 1. 0. 1.
 1. 0. 0. 1. 0. 0. 0. 0. 0. 1. 1. 1. 1. 0. 1. 1. 1. 0. 1. 1. 1. 0. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 0. 1. 0. 0. 1. 1. 1. 1. 1. 1. 0. 0. 1. 0.
 1. 0. 0. 1. 0. 1. 1. 1. 1. 1. 0. 1. 1. 1. 0. 1. 1. 1. 0. 1. 1. 1. 1. 0. 1.
 0. 1. 1. 0. 1. 1. 0. 1. 1. 0. 1. 1. 1. 1. 1. 0. 1. 0. 1. 1. 1. 1. 1. 0. 1. 1.
 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 0. 1. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 0. 0. 0. 1. 1. 0. 1. 0. 1. 1. 1. 1. 1. 0. 1. 0. 1. 1. 1. 0. 0. 1. 1. 1. 1. 0.
 0. 1. 1. 1. 1. 1. 1. 1. 1. 0. 0. 1. 1. 0. 0. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1.
 1. 0. 1. 0. 1. 1. 0. 0.]
le nombre de rejet est 135
voici les résultats du 3ème chaîne
on enregistre le processus de rejet, 1 reprensente rejet, 0 sinon
[1. 1. 1. 1. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 0. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 0. 0. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 0. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 0. 1. 1. 1. 1. 1. 1. 1. 0. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 0. 1. 1. 1. 1. 1. 0. 1.
 1. 1. 1. 1. 0. 1. 0. 1. 1. 1. 1. 1. 1. 1. 0. 1. 0. 0. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 0. 1. 1.]
le nombre de rejet est 174
voici les résultats du 4ème chaîne
on enregistre le processus de rejet, 1 reprensente rejet, 0 sinon
[1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 0. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1.
 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1.]
le nombre de rejet est 190
```

Figure 11: Une comparaison

```
rho_in = 1.0
rho_out = 1.9
!!!
voici les résultats du 1ème chaîne
on enregistre le processus de rejet, 1 reprensente rejet, 0 sinon
le nombre de rejet est 71
voici les résultats du 2ème chaîne
on enregistre le processus de rejet, 1 reprensente rejet, 0 sinon
le nombre de rejet est 131
voici les résultats du 3ème chaîne
on enregistre le processus de rejet, 1 reprensente rejet, 0 sinon
le nombre de rejet est 163
voici les résultats du 4ème chaîne
on enregistre le processus de rejet, 1 reprensente rejet, 0 sinon
le nombre de rejet est 186
le tablaux qui contient les 5 probas
[0.6965174129353234, 0.44776119402985076, 0.3532338308457711 5, 0.373134328358209, 0.6616915422885572]
la probabilité que plus de 5 points mal-classé est 0.02719953698170712
```

Figure 12: Une comparaison

4.3.2 Remarque

Ici, le problème peut être résolu par une augumentation de capacité de calculation afin d'effectuer un nombre important d'expériences, comme dans une chaîne de Markov, la loi de X_n converge exponentiellement vite vers la loi stationnaire. On pense qu'il est possible d'avoir des résultats beaucoup meilleurs par cette approche, mais faute de temps, on n'a pas pu le vérifier.

5 Conclusion

Dans ce projet, on vise à résoudre deux questions principales, la première étant de calculer la probabilité qu'un point fixé soit mal classé, la deuxième étant de calculer la probabilité que le nombre de points mal-classés est plus de 5. Après avoir fait ce projet, on s'est rendu compte que la méthode de monte-carlo ne marche pas très bien pour des évènement rares, et puis des méthodes qu'on a vues dans le cours ne sont pas parfaites non plus, par exemple, comment choisir un changement de probabilité, comment construire une chaîne de Markov efficace. Ces questions n'ont pas une réponse standarde. Ici, un point difficile repose sur la fonction qui prend un graphe comme entrée et deux communautés comme sorties. Cette fonction est difficile à caractériser. La seule façon de connaître cette fonction est de faire des expériences empiriques ce qui ressemble comme une 'boite noire'.