# Projet de Graph Mining : Diffusion et maximisation d'influence sur les réseaux sociaux

Nicolas Noblot

13 avril 2023

# Introduction

- Réseau social = moyen efficace de diffuser de l'information
- Maximisation d'influence = identifier les noeuds d'un réseau qui contribuent le plus à la diffusion de l'information
- Étude de diffusion de l'information et de maximisation d'influence sur Facebook et LastFM Asia
- Plan :
  1. Rappels théoriques
  2. Présentation des datasets
  3. Résultats et discussion

# Notations

- Soit $\mathcal{G} = (V, E)$, si $v \in V$, on note $\mathcal{N}(v)$ l'ensemble des voisins de $v$.
- $\mathcal{S}$ : ensemble des graines, noeuds infectés au départ
- $I_t(\mathcal{S})$ : ensemble des noeuds infectés à l'instant $t$ dans un processus de diffusion à partir de $\mathcal{S}$
- $I(\mathcal{S}) = \bigcup_{t \geq 0} I_t(\mathcal{S})$ : ensemble des noeuds infectés durant un processus de diffusion à partir de $\mathcal{S}$
- $\mathfrak{S}(\mathcal{S}) = \mathbb{E}(|I(\mathcal{S})|)$ : espérance du nombre de noeuds infectés à partir de $\mathcal{S}$

# Modèle à cascades indépendantes

**Algorithme 1 :** Algorithme de diffusion par cascades

**Entrées :** $\mathcal{G} = (V, E)$, $\mathcal{S}$
$I(\mathcal{S}) \leftarrow \mathcal{S}$, $\mathcal{A} \leftarrow \mathcal{S}$
**Tant que** $\mathcal{A} \neq \varnothing$ **faire**
  $L \leftarrow \varnothing$
  **pour chaque** $a \in \mathcal{A}$ **faire**
    **pour chaque** $v \in \mathcal{N}(a)$
     **faire**
      Choisir un nombre aléatoire $p$ selon $\mathcal{U}([0,1])$
      **si** $p < \frac{1}{deg(v)}$ **alors**
        $L \leftarrow L \cup \{v\}$
      **sinon**
        Ne rien faire
      **fin**
    **fin**
  **fin**
  $\mathcal{A} \leftarrow L \backslash I(\mathcal{S})$
  $I(\mathcal{S}) \leftarrow I(\mathcal{S}) \cup \mathcal{A}$
**FinTantque**

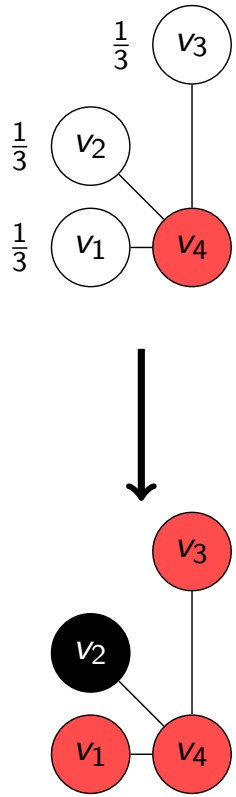



Figure – Exemple de cascade. En rouge, les noeuds infectés . En noir, les noeuds sains.

# Modèle de seuillage

**Algorithme 2 :** Algorithme de diffusion par seuillage

**Entrées :** $\mathcal{G} = (V, E)$, $\mathcal{S}$
Initialiser un vecteur $\theta$ de seuils aléatoires entre 0 et 1
$I(\mathcal{S}) \leftarrow \mathcal{S}$, $\mathcal{A} \leftarrow \mathcal{S}$
**Tant que** $\mathcal{A} \neq \varnothing$ **faire**
  $L \leftarrow \varnothing$
  **pour chaque** $a \in \mathcal{A}$ **faire**
    **pour chaque** $v \in \mathcal{N}(a)$ **faire**
      Calculer la proportion $p$ de voisins de $v$ infectés
      **si** $p > \theta_v$ **alors**
        $L \leftarrow L \cup \{v\}$
      **sinon**
        Ne rien faire
      **fin**
    **fin**
  **fin**
  $\mathcal{A} \leftarrow L \backslash I(\mathcal{S})$
  $I(\mathcal{S}) \leftarrow I(\mathcal{S}) \cup \mathcal{A}$
**FinTantque**

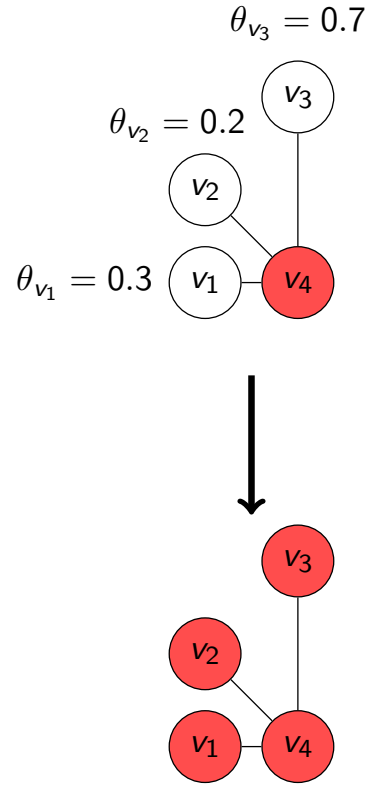



Figure – Exemple d'exécution du modèle de seuillage. Les noeuds infectés sont en rouge.

# Heuristique gloutonne de maximisation de l'influence

**Algorithme 3 :** Heuristique gourmande de maximisation d'influence

**Entrées :** $\mathcal{G}(V, E)$, *budget* $k$

$\mathcal{S} \leftarrow \varnothing$

**Tant que** $|\mathcal{S}| \leq k$ **faire**

  $v^* = \text{argmax}_{v \in V \setminus \mathcal{S}} \left(\mathfrak{S}(\mathcal{S} \cup \{v\})\right) - \mathfrak{S}\mathcal{S}$

  $\mathcal{S} \leftarrow \mathcal{S} \cup \{v^*\}$

**FinTantque**

- Algorithme très calculatoire en pratique.
- Estimation de $\mathfrak{S}(\mathcal{S})$ à l'aide de la moyenne empirique du nombre de noeuds infectés sur plusieurs exécutions d'un algorithme de diffusion.

# Dataset ego-Facebook

- Dataset de cercle d'amis sur Facebook
- Les noeuds sont les utilisateurs. Un sommet $A$ est relié à un sommet $B$ si $A$ est ami avec $B$.
- Graphe non-pondéré et non-orienté
- Statistiques générales :

| Statistique | Valeur |
|---|---|
| Nombre de noeuds | 4 039 |
| Nombre d'arêtes | 88 234 |
| Densité | 0.01 |
| Transitivité | 0.52 |
| Coefficient de clustering moyen | 0.606 |
| Diamètre | 8 |
| Rayon | 4 |

- Source : J. McAuley and J. Leskovec. Learning to Discover Social Circles in Ego Networks. NIPS, 2012.

# Dataset LastFM Asia

- Graphe d'utilisateurs en Asie du site LastFM (site de recommandation de musiques)
- Les noeuds sont les utilisateurs et une arête existe entre $A$ et $B$ s'ils se suivent mutuellement.
- Graphe non-pondéré et non-orienté
- Statistiques générales :

| Statistique | Valeur |
|---|---|
| Nombre de noeuds | 7 624 |
| Nombre d'arêtes | 27 806 |
| Densité | 0.00096 |
| Transitivité | 0.18 |
| Coefficient de clustering moyen | 0.22 |
| Diamètre | 15 |
| Rayon | 8 |

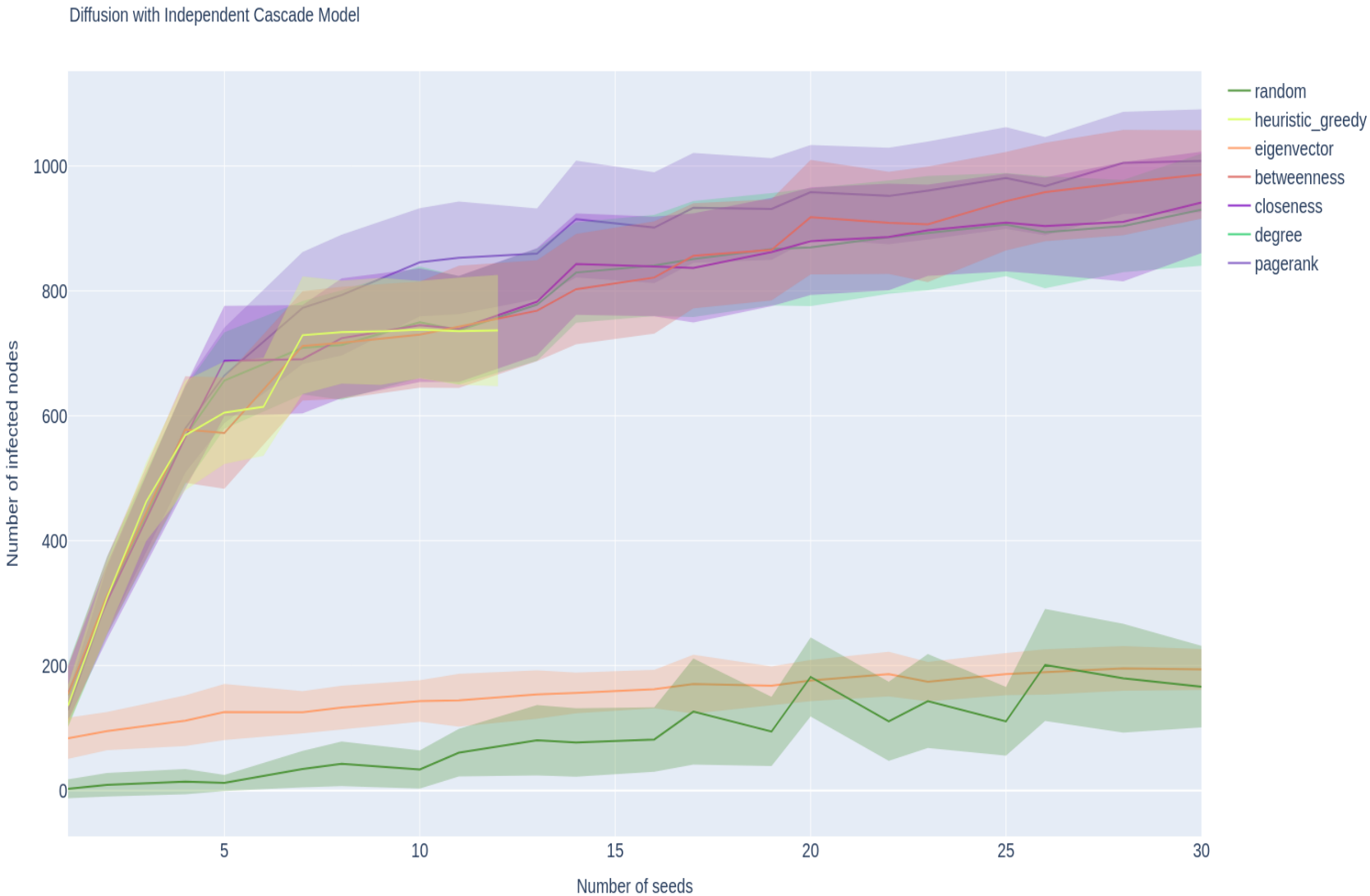
- Source : B. Rozemberczki and R. Sarkar. Characteristic Functions on Graphs: Birds of a Feather, from Statistical Descriptors to Parametric Models. 2020.

# Expériences

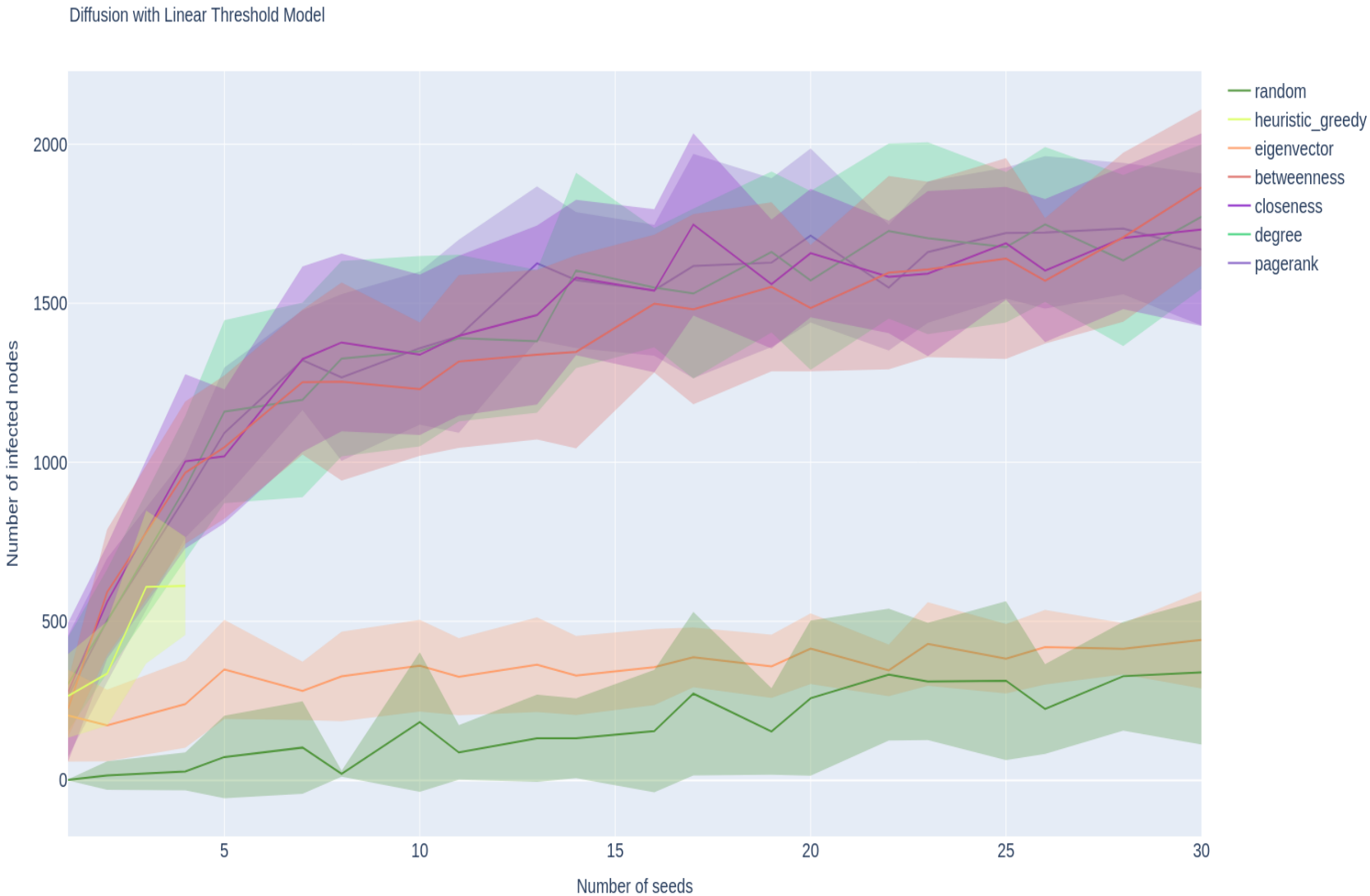
Sur chaque dataset :

- Maximisation d'influence avec un budget inférieur à 30 noeuds
- Utilisation des modèles de seuillage et de cascade avec 6 stratégies de sélection des noeuds de départ différentes : aléatoire, centralité pagerank, centralité closeness, centralité spectrale, centralité intermédiaire, degré
- Comparaison avec l'heuristique gourmande

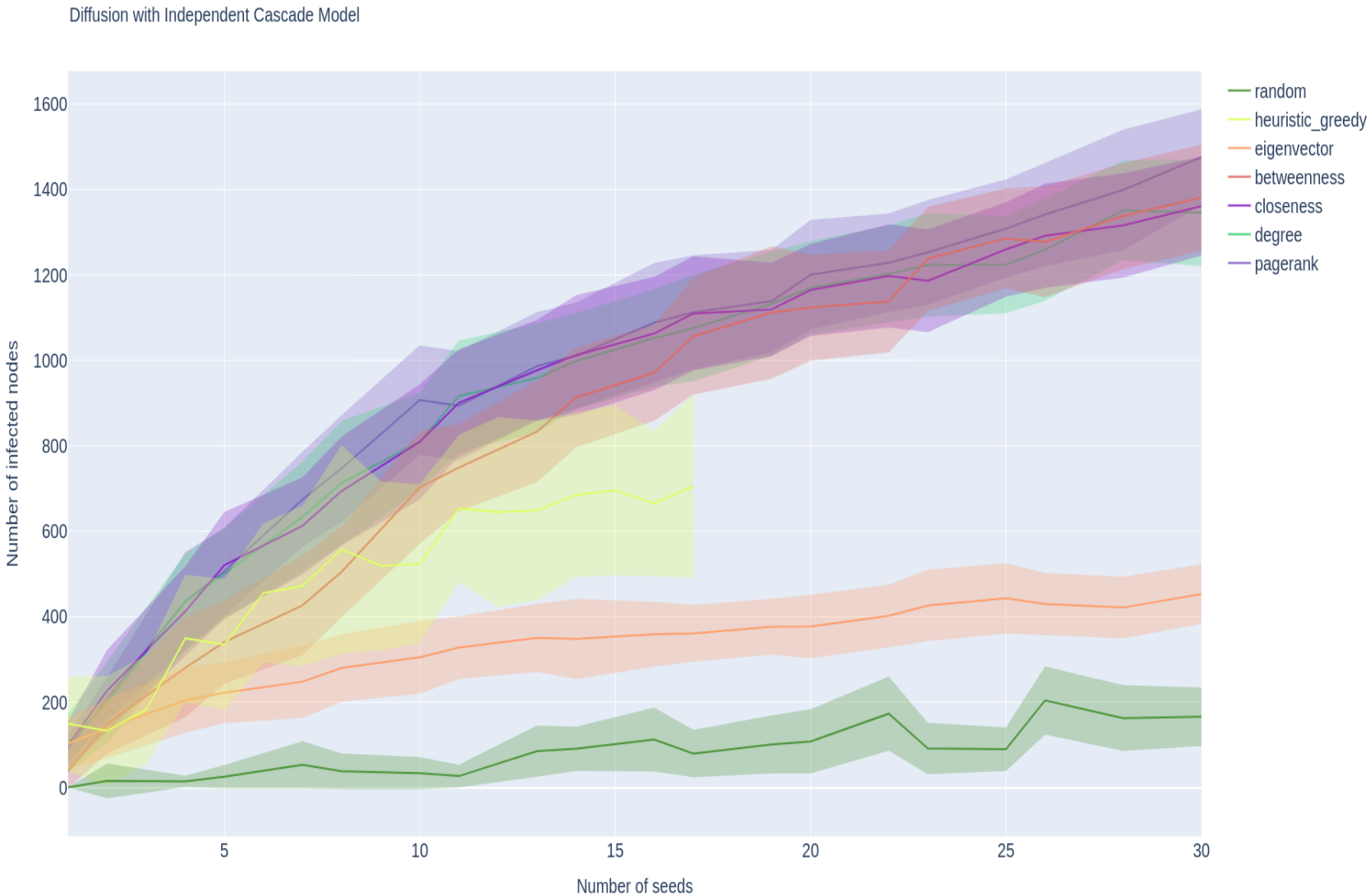
# Résultats de la diffusion sur le dataset Facebook (1/2)



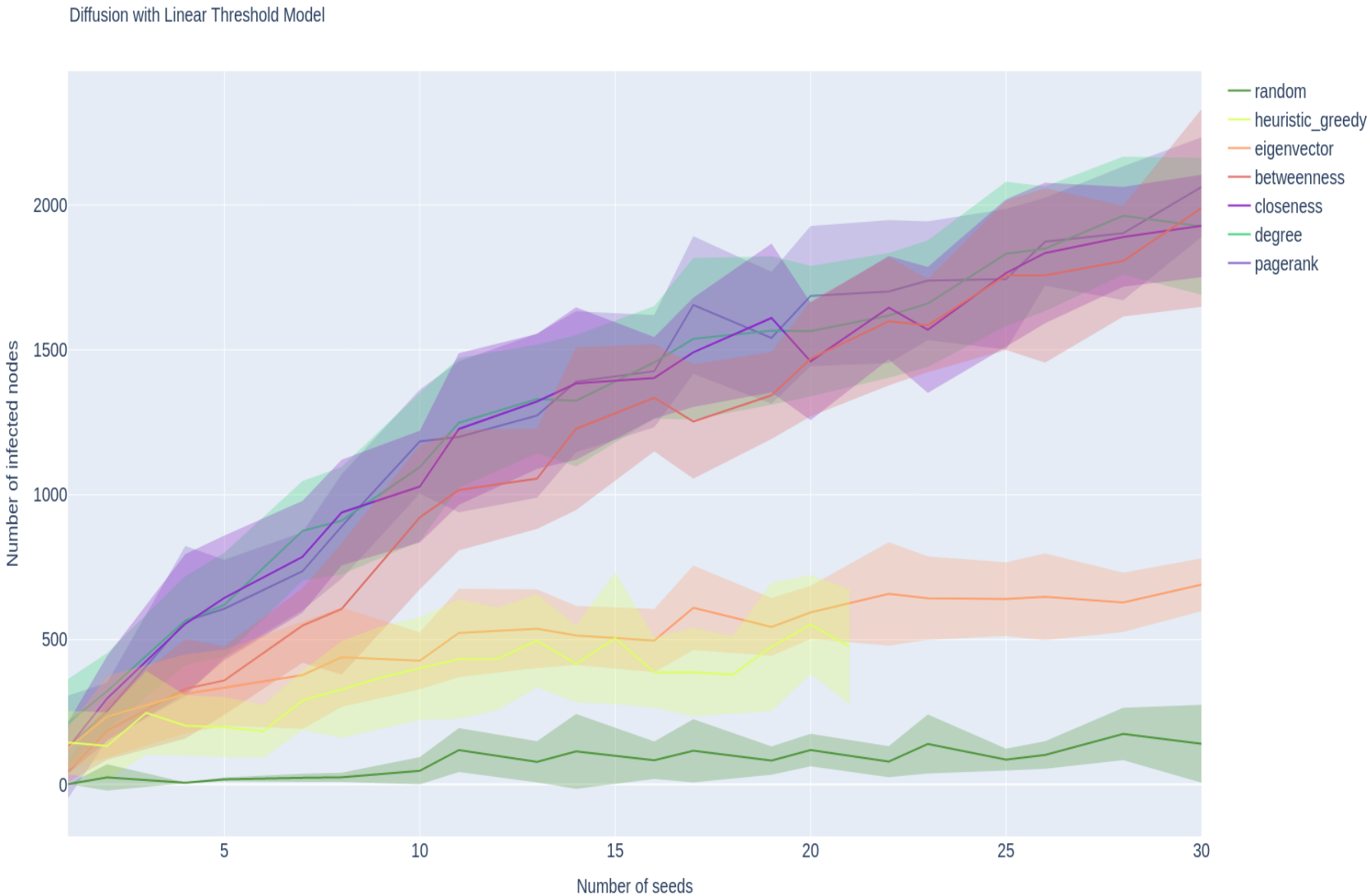
# Résultats de la diffusion sur le dataset Facebook (2/2)



# Résultats de la diffusion sur le dataset LastFM Asia (1/2)



# Résultats de la diffusion sur le dataset LastFM Asia (2/2)



# Visualisation des graphes

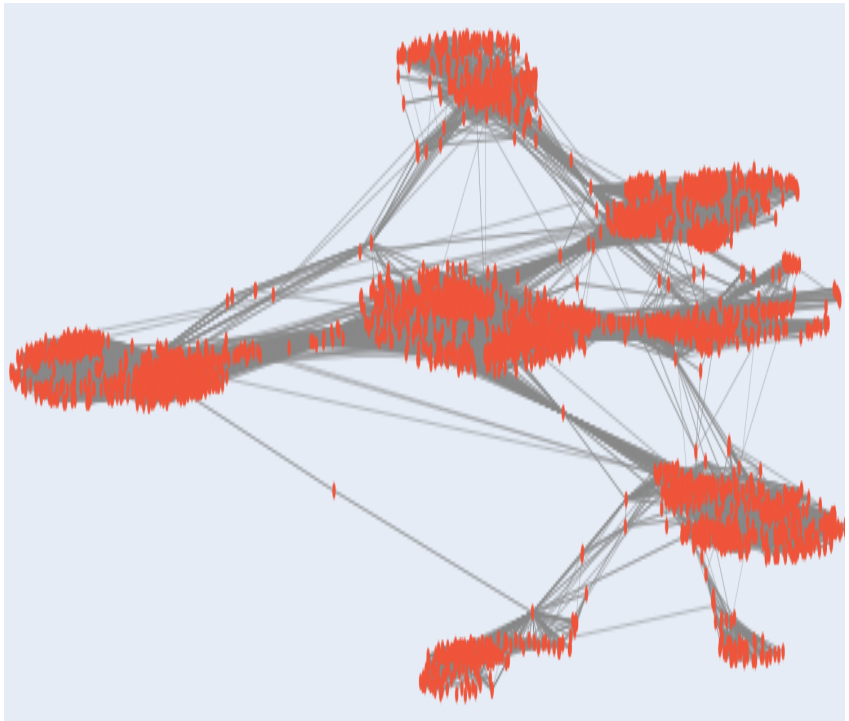


Figure – Projection 2D du graphe de Facebook

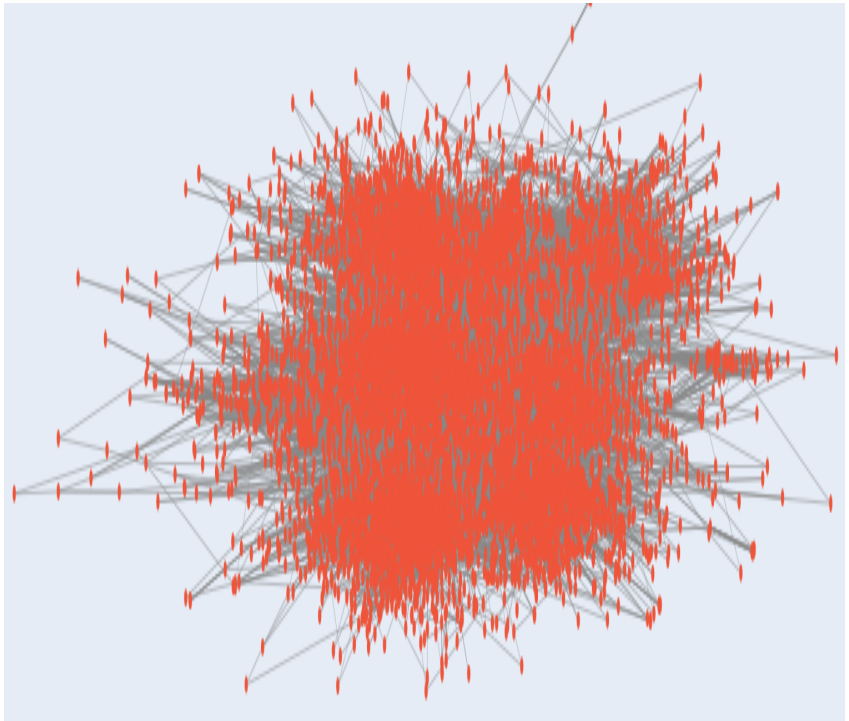


Figure – Projection 2D du graphe de LastFM Asia

# Visualisation de centralité

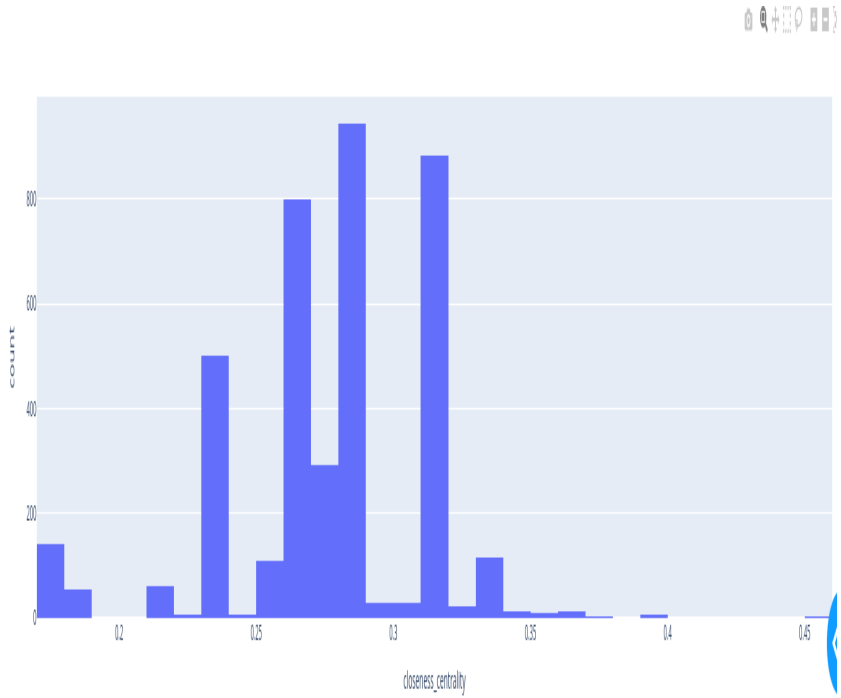



Figure – Distribution de la centralité closeness sur le dataset de Facebook

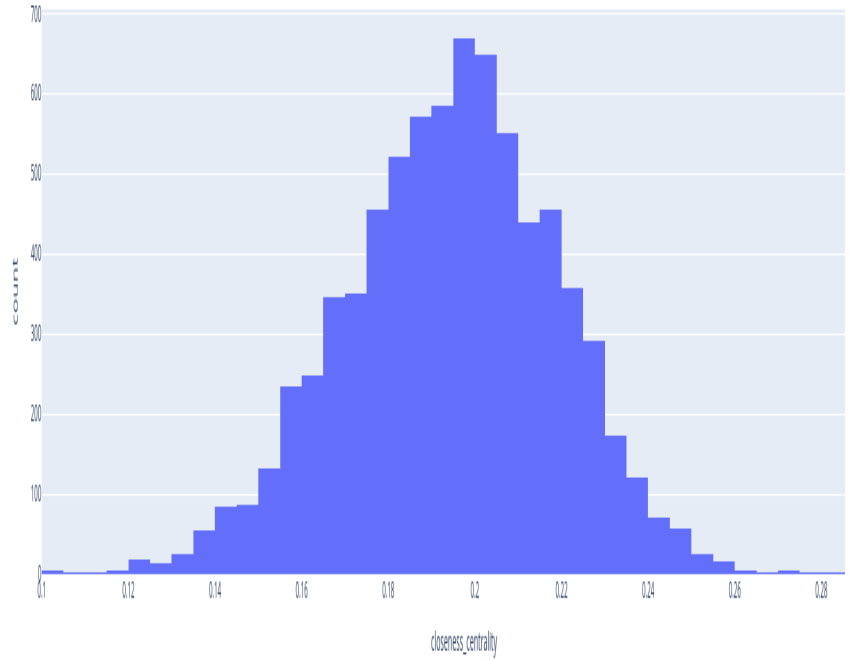



Figure – Distribution de la centralité closeness sur le dataset de LastFM Asia

Sur les deux jeux de données, les autres centralités ont une distribution de type loi exponentielle.

# Conclusion

- La diffusion d'information sur les réseaux sociaux dépend de :
  - la stratégie de sélection des germes utilisée
  - du modèle de diffusion
  - de la topologie du réseau
- Algorithmes de diffusion peuvent être coûteux en temps et en ressources même sur des petits réseaux
- Code du projet disponible sur github : `https://github.com/noblotni/graph_mining_labs`