

Projet LINFO1114 – Mathématiques discrètes

« Ranking de réseaux sociaux et pages web : PageRank »

Professeur Marco Saerens marco.saerens@uclouvain.be
Bureau b.125
Adresse Université catholique de Louvain
Place des Doyens 1
1348 Louvain-La-Neuve
Belgique
Assistants Diego Eloi diego.eloi@uclouvain.be
Alexis Airson alexis.airson@uclouvain.be
Date 13 Novembre 2025

Objectif : Le but de ce projet est d'implémenter et de tester un algorithme permettant de classer (to rank) les noeuds d'un graphe en assignant un score d'importance à chacun de ces noeuds. Cet algorithme est celui de PageRank avec téléportation et vecteur de personnalisation. Vous travaillerez par groupe de *trois* étudiant-e-s impérativement (merci de vous inscrire dans un groupe sur Moodle). Chaque groupe trouvera un vecteur de personnalisation différent sur Moodle, sur lequel il devra se baser pour calculer les scores PageRank. La résolution doit être détaillée dans le rapport en utilisant trois méthodes différentes (1) en résolvant un système d'équations linéaires en Python (voir slides du cours), (2) en implémentant la "power method" en Python à l'aide de Numpy (bibliothèque Python pour le calcul scientifique et matriciel) et (3) en simulant une marche aléatoire sur le graphe (random walk) en partant d'un noeud spécifique de votre choix. Il faudra suivre rigoureusement ces algorithmes, comme détaillé au cours. Vous trouverez toutes les informations utiles dans les slides du cours mais aussi dans des ouvrages de "link analysis" (par exemple l'ouvrage *Google's PageRank and Beyond* de Amy Langville et Carl Meyer).

Fonctions à implémenter (Python 3.5+) :

Dans le cadre de l'implémentation de la résolution du système d'équations linéaires, la signature de la fonction est :

```
def pageRankLinear(A~: np.matrix, alpha~: float, v~: np.array) -> np.array
```

- **Input :** Une matrice d'adjacence¹ **A** d'un graphe dirigé, pondéré et régulier *G*, un vecteur de personnalisation **v**, ainsi qu'un paramètre de téléportation α compris entre 0 et 1 (0.9 par défaut et pour les résultats à présenter).
Toutes ces valeurs sont non-négatives.
- **Output :** Un vecteur **x** contenant les scores d'importance des noeuds ordonnés dans le même ordre que les lignes de la matrice d'adjacence (représentant les noeuds).

Dans le cadre de l'implémentation de la power method, la signature de la fonction est :

```
def pageRankPower(A~: np.matrix, alpha~: float, v~: np.array) -> np.array
```

- **Input :** Une matrice d'adjacence **A** d'un graphe dirigé, pondéré et régulier *G*, un vecteur de personnalisation **v**, ainsi qu'un paramètre de téléportation α compris entre 0 et 1 (0.9 par défaut et pour les résultats à présenter).
- **Output :** Un vecteur **x** contenant les scores d'importance des noeuds ordonnés dans le même ordre que les lignes de la matrice d'adjacence (représentant les noeuds).

Dans le cadre de l'implémentation de la marche aléatoire, la signature de la fonction est :

```
def randomWalk(A~: np.matrix, alpha~: float, v~: np.array) -> np.array
```

- **Input :** Une matrice d'adjacence **A** d'un graphe dirigé, pondéré et régulier *G*, un vecteur de personnalisation **v**, ainsi qu'un paramètre de téléportation α compris entre 0 et 1 (0.9 par défaut et pour les résultats à présenter).
- **Output :** Un vecteur **x** contenant les scores d'importance des noeuds ordonnés dans le même ordre que les lignes de la matrice d'adjacence (représentant les noeuds).

Vous devez donc calculer les scores PageRank de trois façons différentes :

- En Python, en résolvant un système d'équations linéaires. Pour cette technique, vous pouvez utiliser les fonctions numpy permettant de résoudre un système d'équations linéaires, en utilisant impérativement l'algèbre matriciel (formules matricielles et non "elementwise").
- En Python, en calculant le vecteur propre dominant de gauche de la matrice Google, en utilisant la "power method". Pour cette technique, vous devez implémenter vous-même la power method, et donc la boucle permettant de calculer le vecteur propre dominant de gauche de la matrice Google (en utilisant les formules matricielles).
- En Python, en simulant une marche aléatoire de longueur fixée à au moins 10000 pas sur le graphe dirigé et pondéré. On part d'un noeud initial (fixé au noeud A), puis à

1. Aussi parfois notée **W** (pour Web ou Weight) dans les slides du cours.

chaque étape on choisit le noeud suivant selon les probabilités de transition définies par la matrice Google. Le score PageRank approximatif de chaque noeud est alors obtenu en comptant la fréquence relative de visite de chaque page au cours de la marche aléatoire.

Comme nous considérons une téléportation *non-uniforme*, vous devrez calculer la matrice Google à partir de la matrice de probabilités de transitions du graphe et un vecteur de personnalisation non-uniforme. Notez que, dans l'algorithme basé sur la "power method", il faudra initialiser les scores par le degré entrant (indegree) de chaque noeud (la somme de chaque colonne j de la matrice d'adjacence, $\sum_{i=1}^n a_{ij}$).

Ainsi que déjà mentionné, évitez d'utiliser les fonctions permettant de calculer directement des vecteurs propres² car le but du projet est justement de comprendre et d'implémenter la power method. Donc, dans ce cadre, vous ne pouvez utiliser cette librairie que pour effectuer des opérations matricielles simples telles que la transposée, la multiplication matricielle, etc. Par contre, pour la résolution à l'aide d'un système d'équations, nous vous autorisons à utiliser des fonctions spécialisées calculant la solution du système après l'avoir mis sous forme matricielle.

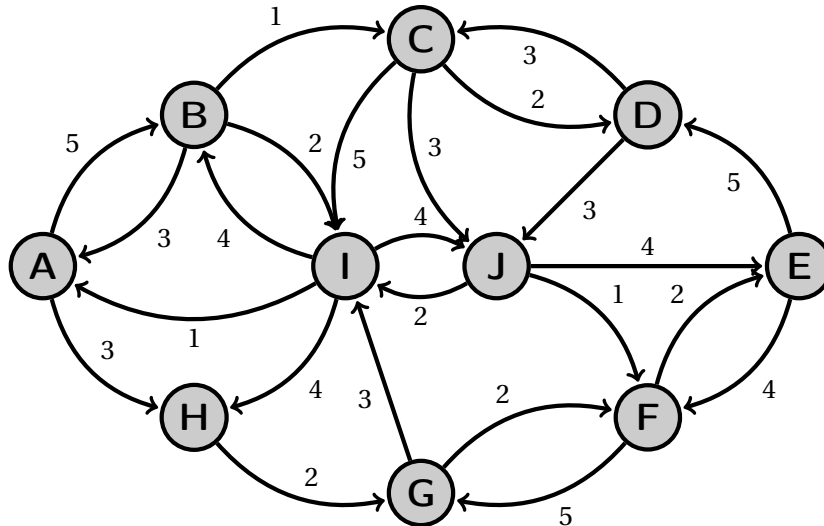
De plus, il vous est demandé d'ajouter à votre code une fonction "main" qui lit un fichier csv contenant la matrice d'adjacence **A** du graphe présenté ci-dessous (séparez vos valeurs par une virgule et chaque ligne de la matrice par un passage à la ligne), qui exécute le calcul de PageRank (via vos fonctions pageRankPower, pageRankLinear et randomWalk) et imprime les résultats à l'écran. Nous allons exécuter ce code lors de l'évaluation du projet. Tous les fichiers doivent se trouver dans un même répertoire.

Données :

– Un graphe dirigé et pondéré par le nombre d'hyperliens entre les paires de noeuds, représentant un réseau de 10 noeuds (matrice d'adjacence).

– Ainsi qu'un vecteur de personnalisation **v**, propre à chaque groupe, disponible sur Moodle dans le dossier "Vecteur".

2. Par contre, vous pouvez les utiliser pour vérifier les résultats de votre implémentation.



Rapport : Le rapport est un fichier PDF (7 pages maximum tout inclu ; écrit en LaTeX) qui se compose de :

- Un rappel théorique expliquant brièvement (avec équations ; environ une page) comment calculer le vecteur de scores PageRank (en résolvant un système d'équations linéaires et en implémentant la power method). Discuter également l'impact du paramètre de téléportation α et du vecteur de personnalisation \mathbf{v} sur le score.
- La présentation numérique en LaTeX du système d'équations linéaires (sous forme matricielle) permettant de calculer le score PageRank du graphe ci-dessus avec le vecteur de personnalisation qui vous a été assigné.
- Présentez le score PageRank final obtenu par power method et système d'équations (les deux méthodes devraient fournir les mêmes valeurs).
- Proposez un graphe x - y illustrant l'évolution de l'erreur à chaque pas (time step k) du marcheur aléatoire entre le score exact (calculé par power method ou système d'équations) et le score approximé par la marche aléatoire en fonction du temps (voir plus bas pour les détails).
- Proposez une représentation du graphe où les noeuds sont colorés en fonction de leur score PageRank.
- Une discussion des résultats obtenus.

Code complet : Le code complet comprendra deux fichiers .py, un fichier contenant vos fonctions pour la power method, la résolution du système d'équations linéaires et la marche aléatoire, et un fichier "main" qui lit vos deux fichiers csv (matrice d'adjacence et vecteur de personnalisation) et calcule les score PageRank en faisant appel à vos fonctions définies dans l'autre fichier. Votre fichier contenant les fonctions doit donc contenir au moins ces trois fonctions :

- **La power method :** Dans le cadre de l'implémentation en Python de la power method, nous vous demandons plusieurs impressions à l'écran lorsque le code est exécuté :
 - de la matrice d'adjacence \mathbf{A} ,

- de la matrice de probabilités de transition \mathbf{P} ,
- de la matrice Google \mathbf{G} ,
- des trois premières itérations de la power method (vecteur de scores), et
- du résultat final, c'est à dire le score PageRank final après convergence (output de la fonction).
- **La résolution de système d'équations linéaires** : Dans le cadre de l'implémentation en Python de la résolution du système d'équations, nous demandons l'impression à l'écran :
 - du résultat final, c'est à dire le score PageRank final obtenu (output de la fonction).
- **La simulation de la marche aléatoire sur le graphe** : Dans le cadre de l'implémentation en Python de la marche aléatoire, nous demandons l'impression à l'écran :
 - du résultat final, c'est à dire le score PageRank final obtenu (output de la fonction)
 - un graphe x - y à deux dimensions illustrant l'évolution de l'erreur moyenne entre le score exact et le score approximé par la marche aléatoire en fonction du pas de temps k (time step) :

$$\epsilon(k) = \frac{1}{n} \sum_{i=1}^n |p_i^{\text{rw}}(k) - p_i^*|$$

où n désigne le nombre de noeuds, $p_i^{\text{rw}}(k)$ le score PageRank estimé par la marche aléatoire pour le noeud i et p^* le score PageRank exact, qui ne varie pas.

Attention, respectez bien ces consignes car nous nous baserons sur celles-ci pour calculer la note du projet de manière semi-automatique. N'oubliez pas de bien commenter vos codes !

Langage de programmation : L'implémentation devra impérativement être codée en Python 3.5+ (en essayant de rester back-compatible) en respectant les consignes et les signatures des fonctions énoncées ci-dessus. N'imprimez pas de valeurs intermédiaires inutiles. Comme déjà mentionné, vous devez utiliser une librairie Python externe de calcul et de manipulation matricielle/vectorielle nommée numpy. Cette librairie vous évitera d'implémenter la multiplication matricielle, ou la transposition, de manière à vous concentrer sur l'algorithme proprement dit.

Evaluation et consignes : Le projet est obligatoire et à réaliser par groupes de trois étudiantes et étudiants. L'évaluation portera sur le contenu du rapport (maximum 7 pages) et le code (lisibilité, structure, **commentaires**,...) et comptera pour 3 points sur 20 dans la note finale (le reste des points étant donné par l'examen écrit). Le rapport doit être très professionnel³, du type article scientifique ou technique, et doit contenir les références bibliographiques sur lesquelles vous vous êtes basés.

Les différents fichiers, c'est-à-dire les fichiers de code source (un fichier pour le main qui s'exécute et un fichier contenant les fonctions), les fichiers .csv (la matrice d'adjacence et le vecteur de personnalisation) et le rapport en pdf, tous compressés ensemble (nom du fichier compressé : "groupe" suivi du numéro du groupe (deux chiffres), et suivi par les noms de famille (majuscule à la première lettre) des membres du groupe séparés par des underscores

3. Par exemple pas de copier/coller d'images de formule mathématique ou algorithme.

et par ordre alphabétique⁴. Nous ne corrigeons pas les projets qui ne respectent pas cette consigne : vous devrez re-soumettre le projet, avec pénalités. Les projets sont à remettre impérativement sur Moodle pour le dimanche 21 décembre 2025, avant 23h59. Aucun retard ne sera accepté; dès lors soyez prévoyant et évitez de remettre votre projet dans les dernières minutes. La note sera la même pour tous les membres du groupe.

Bon travail!

4. Par exemple "groupe05_Eloi_Saerens_Airson.zip".