Math ECG 2. 2022-2023

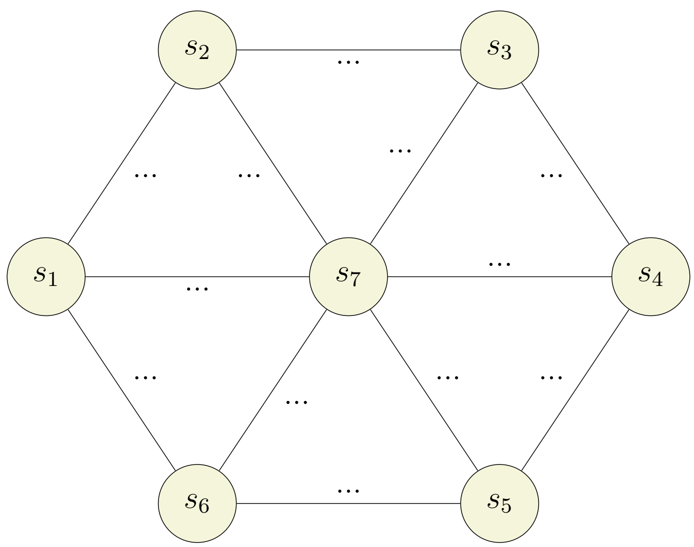
Mathématiques Appliquées - F. Gaunard
http://frederic.gaunard.com
ENC Bessières, Paris 17e.

Devoir Maison n°5

À rendre le 12/12

Exercice 1

On considère le graphe pondéré dont une représentation graphique partielle est donnée ci-dessous.



(1) (a) Recopier et compléter ce graphe pondéré de sorte que la matrice des poids soit la matrice

$$\begin{pmatrix} 0 & 3 & +\infty & +\infty & +\infty & 1 & 7 \\ 3 & 0 & 8 & +\infty & +\infty & +\infty & 2 \\ +\infty & 8 & 0 & 2 & +\infty & +\infty & 6 \\ +\infty & +\infty & 2 & 0 & 3 & +\infty & 8 \\ +\infty & +\infty & +\infty & 3 & 0 & 9 & 4 \\ 1 & +\infty & +\infty & +\infty & 9 & 0 & 5 \\ 7 & 2 & 6 & 8 & 4 & 5 & 0 \end{pmatrix}$$

(b) Ce graphe est-il complet ? Est-il connexe ?

(2) L'algorithme de Dijkstra permet de déterminer le plus court chemin entre deux sommets du graphe. On rappelle son fonctionnement *via* le tableau suivant qui présente les étapes de l'algorithme permettant de trouver le plus court chemin $s_1 - s_2 - s_7 - s_5 - s_4$ (de longueur 12) entre s_1 et s_4.

s_1	s_2	s_3	s_4	s_5	s_6	s_7	étape i	choix sommet étape $i+1$
0	$+\infty$	$+\infty$	$+\infty$	$+\infty$	$+\infty$	$+\infty$	0	s_1
0	3^{s_1}	$+\infty$	$+\infty$	$+\infty$	1^{s_1}	7^{s_1}	1	s_6
0	3^{s_1}	$+\infty$	$+\infty$	10^{s_6}	1^{s_1}	6^{s_6}	2	s_2
0	3^{s_1}	11^{s_2}	$+\infty$	10^{s_6}	1^{s_1}	5^{s_2}	3	s_7
0	3^{s_1}	11^{s_2}	13^{s_6}	9^{s_7}	1^{s_1}	5^{s_2}	4	s_5
0	3^{s_1}	11^{s_2}	12^{s_5}	9^{s_7}	1^{s_1}	5^{s_2}	5	s_3
0	3^{s_1}	11^{s_2}	12^{s_5}	9^{s_7}	1^{s_1}	5^{s_2}	6	s_4
0	3^{s_1}	11^{s_2}	12^{s_5}	9^{s_7}	1^{s_1}	5^{s_2}	7	$-$

Dresser un tableau analogue correspondant à l'application de l'algorithme de Dijkstra avec une origine en s_5. Quelle est le plus court chemin de s_5 à s_1? Quelle est sa longueur ?

(3) On suppose qu'on dispose d'une fonction Python `Dijkstra_dist(G, depart)` qui prenant en argument un graphe pondéré d'ordre n représenté par sa matrice des poids `G` et un sommet `depart` (représenté par un indice $i \in [\![0, n-1]\!]$ et correspondant au sommer s_{i+1}) renvoie la liste $l = [d_1, .., d_n]$ où d_j est la distance d'un plus court chemin de `départ` au sommet s_j. Par exemple, les instructions

```
import numpy as np

Inf=np.inf
G=np.array([[0,3, Inf, Inf, Inf, 1,  7],
            [3, 0, 8, Inf, Inf, Inf, 2],
            [Inf, 8, 0, 2, Inf, Inf, 6],
            [Inf, Inf, 2, 0, 3, Inf, 8],
            [Inf, Inf, Inf, 3, 0, 9, 4],
            [1, Inf, Inf, Inf, 9, 0, 5],
                 [7, 2, 6, 8,  4, 5, 0]])
print(Dijkstra_dist(G, 0))
```

Affichage Python

```
>>>
[0, 3, 11, 12, 9, 1, 5]
```

(a) Que va afficher l'instruction ci-dessus ?

```
print(Dijkstra_dist(G, 4))
```

Exercice 2* - Graphes aléatoires

Dans tout l'exercice n désigne un entier supérieur ou égal à 2 et p un réel de l'intervalle $]0;1[$.
Soit $S = \{s_1, ..., s_n\}$ un ensemble fini de n sommets. On s'intéresse dans cet exercice à des graphes aléatoires construits à partir de l'ensemble S.
Plus précisément, pour tout couple $(i, j) \in [\![1, n]\!]^2$ avec $i < j$, on introduit les variables aléatoires $T_{i,j}$ mutuellement indépendantes de même loi $\mathcal{B}(p)$. Les arêtes du graphe sont les paires de sommets $\{s_i, s_j\}$ pour lesquelles $T_{i,j} = 1$.
On dit qu'un sommet est isolé s'il n'y a aucune arête incidente à ce sommet.
On introduit N_n la variable aléatoire égale au nombre d'arêtes du graphe. Pour $k \in [\![1, n]\!]$, on note D_k la variable aléatoire qui prend pour valeur le degré du sommet s_k (c'est à dire le nombre d'arêtes incidentes à ce somment).
Enfin, on note X_k la variable aléatoire qui vaut 1 si le sommet s_k est isolé et 0 sinon puis Z_n celle égale

au nombre de sommets isolés du graphe.

(1) Recopier et compléter la fonction Python ci-dessous qui renvoie la liste d'adjacence d'un tel graphe aléatoire en prenant pour argument la liste des sommets `S` et p.
On rappelle que la liste d'adjacence L d'un graphe dont l'ensemble des sommets est $S = \{s_1, ..., s_k\}$ est la liste dont la composante L_i est la liste des sommets adjacents au sommet s_i.

```
import numpy as np
import numpy.random as rd

def list_adj(S, p):
    n=.....
    l=[ [ ] for k in range(n)]
    for i in range(n-1):
        for j in range(i+1, n):
            if rd.random()<p :
                l[i].append(.....)
                l[j].append(.....)
    return l
```

(2) On exécute alors la commande suivante qui génère l'affichage suivant

```
 list_adj('abcdef', 1/3)
```

Affichage Python

```
>>>
[['b', 'e', 'f'], ['a', 'c', 'f'], ['b'], [], ['a'], ['a', 'b']]
```

(a) Donner une représentation graphique du graphe aléatoire généré par cette exécution.
(b) Ce graphe contient-il des sommets isolés? Si oui, combien ?
(c) Expliciter la matrice d'adjacence de ce graphe. Quelle propriété a-t-elle ? Pourquoi ?

(3) Écrire une fonction Python d'en-tête `def nb_som_is(l):` qui, prenant en argument la liste `l` d'adjacence d'un graphe renvoie le nombre de sommets isolés de celui-ci.

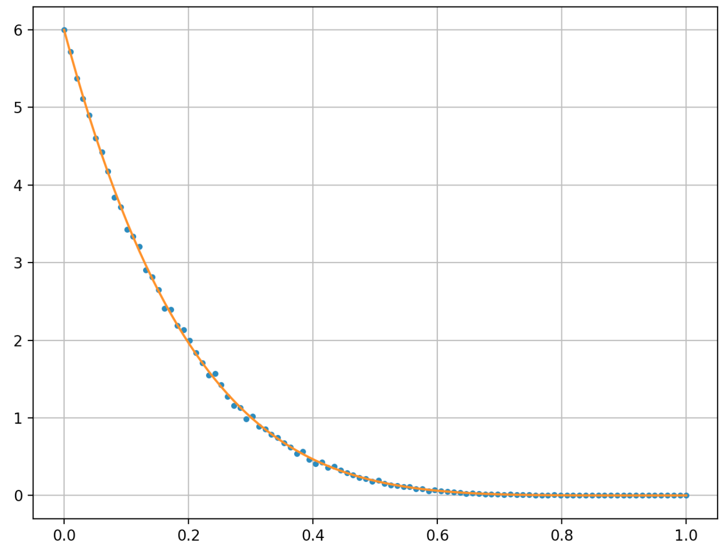
(4) On dispose de la fonction mystère suivante

```
def mystere(S, p):
    return np.mean([nb_som_is(list_adj(S,p)) for k in range(1000)])
```

(a) Que renvoie-t-elle ?
(b) On ajoute les instructions suivantes dont l'exécution produit la figure ci-contre. Interpréter et émettre une conjecture.

```
S='abcdef'
x=np.linspace(0,1, 100)
y=[mystere(S, p) for p in x]
w=[len(S)*(1-p)**(len(S)-1) for p in x]
plt.grid()
plt.plot(x,y, '.')
plt.plot(x,w)
plt.show()
```

Affichage Python



(5) (a) Justifier que $N_n(\Omega) \subset \llbracket 0, \binom{n}{2} \rrbracket$.

(b) Montrer que $P(N_n = 0) = P\left(\bigcap_{i=1}^{n-1} \bigcap_{j=i+1}^{n} [T_{i,j} = 0]\right) = (1-p)^{n(n-1)/2}$.

(c) Que vaut $P\left(N_n = \binom{n}{2}\right)$?

(6) (a) Justifier que, pour tout $k \in \llbracket 1, n \rrbracket$, on a

$$D_k = \sum_{i=1}^{k-1} T_{i,k} + \sum_{i=k+1}^{n} T_{k,i}.$$

En déduire la loi de D_k.

(b) Montrer que, pour $1 \leq k < \ell \leq n$, on a

$$\operatorname{cov}(D_k, D_\ell) = p(1-p).$$

Les variables D_k et D_ℓ sont-elles indépendantes ?

(c) Déterminer $E(Z_n)$. Est-ce cohérent avec la conjecture précédente?

(7) (a) Montrer que, pour tous $i < j$, on a

$$P([X_i = 1] \cap [X_j = 1]) = (1-p)^{2n-3}.$$

(b) En observant que

$$Z_n^2 = \sum_{i=1}^{n} X_i + 2 \sum_{1 \leq i < j \leq n} X_i X_j,$$

montrer que

$$E(Z_n^2) = n(1-p)^{n-1} + n(n-1)(1-p)^{2n-3}.$$