

Devoir Surveillé 4 - BCPST 2

durée : 3 heures

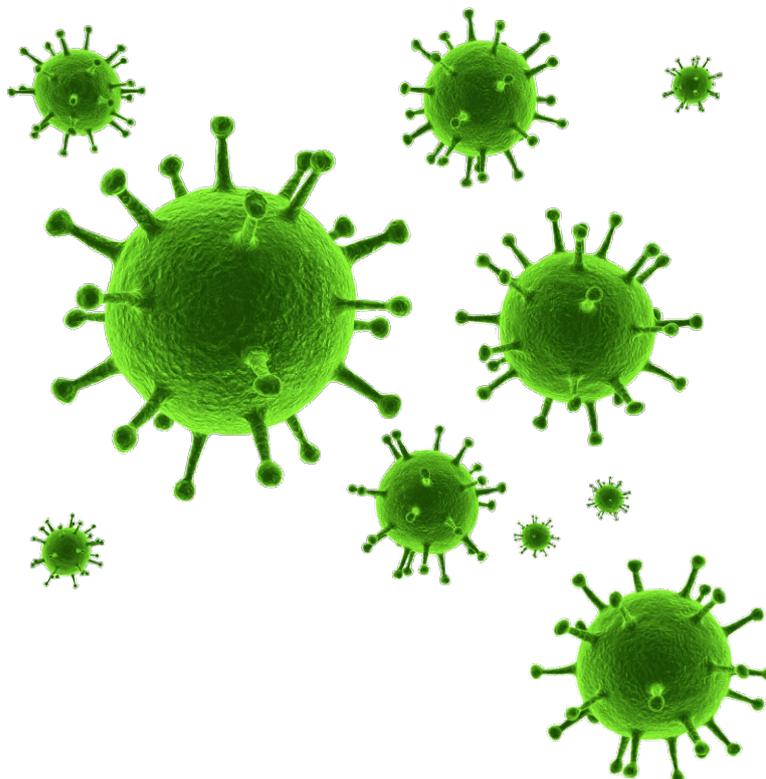
Documents et calculatrices non autorisés.

Aucune réponse non justifiée ne sera prise en compte dans la notation. Il sera tenu compte de la clarté de la rédaction.

Le sujet se compose de 4 parties largement indépendantes.

- La **Partie I** est consacrée à l'étude d'un modèle dit à compartiments, qui permet de quantifier la propagation temporelle d'un virus dans une population.
- Dans la **Partie II**, on met en œuvre la méthode de la puissance qui permet de déterminer un vecteur propre d'une matrice.
- Dans la **Partie III**, on étudie une modèle stochastique simple permettant de déterminer la probabilité d'infection d'un individu par un virus.
- La **Partie IV** est consacrée à l'étude d'un modèle permettant de quantifier la propagation spatiale d'un virus.

Quelques fonctions Python utiles sont rappelées en **Annexe** à la fin du sujet.



Partie I - Évolution d'une population touchée par un virus

Dans une population touchée par un virus, tous les individus porteurs n'ont pas le même potentiel de contagion. De plus, les personnes exposées ne deviennent pas nécessairement contagieuses et celles qui sont contagieuses ne développent pas nécessairement des symptômes.

Nous choisissons une unité de temps u et nous décidons de découper la population en contact avec le virus en 3 classes disjointes :

- la classe des personnes exposées au virus au cours de l'unité de temps
- la classe des personnes contagieuses sans symptômes au cours de l'unité de temps
- la classe des personnes contagieuses avec symptômes au cours de l'unité de temps.

Pour $n \in \mathbb{N}$, nous décidons de représenter la population après n unités de temps par la matrice $N_n \in \mathcal{M}_{3,1}(\mathbb{R})$ définie par :

$$N_n = \begin{pmatrix} e_n \\ a_n \\ s_n \end{pmatrix},$$

où :

- e_n désigne l'effectif de la classe des personnes exposées au début de la n -ième unité de temps
- a_n désigne l'effectif de la classe des personnes contagieuses asymptomatiques au début de la n -ième unité de temps
- s_n désigne l'effectif de la classe des personnes contagieuses avec symptômes au début de la n -ième unité de temps

On note :

- F_1 le nombre de personnes exposées par unité de temps pour un individu contagieux asymptomatique.
- F_2 le nombre de personnes exposées par unité de temps pour un individu contagieux avec symptômes.
- p_1 la probabilité qu'un individu exposé devienne contagieux asymptomatique au cours d'une unité de temps
- p_2 la probabilité qu'un individu contagieux asymptomatique développe des symptômes au cours d'une unité de temps.

1. Soit $n \in \mathbb{N}$.

- Exprimer e_{n+1} en fonction de a_n , s_n et de F_1 , F_2 .
- Exprimer a_{n+1} et s_{n+1} en fonction de e_n , a_n et de p_1 , p_2 .
- Déterminer alors la matrice $A \in \mathcal{M}_3(\mathbb{R})$ telle que : $N_{n+1} = AN_n$.

2. **Étude d'un exemple.**

Nous considérerons un virus et une population telle qu'une personne n'est pas en contact (exposition et infection) plus de 30 jours avec le virus et choisissons une unité de temps u de 10 jours. Après étude statistique, on estime que :

$$F_1 = 13, F_2 = 12, p_1 = \frac{1}{4}, p_2 = \frac{1}{2}.$$

Pour tout entier naturel n , $N_{n+1} = AN_n$ où

$$A = \begin{pmatrix} 0 & 13 & 12 \\ \frac{1}{4} & 0 & 0 \\ 0 & \frac{1}{2} & 0 \end{pmatrix}$$

Montrer que pour tout entier naturel n : $N_n = A^n N_0$.

3. **Réduction de la matrice A .**

- Démontrer que A possède comme valeurs propres les nombres 2 et $-\frac{1}{2}$, et déterminer la troisième valeur propre de A .
Les valeurs propres de A seront notées λ_1 , λ_2 , λ_3 telles que :

$$\lambda_1 > |\lambda_2| \geq |\lambda_3|.$$

La matrice A est-elle diagonalisable ?

- Déterminer une base de chaque sous-espace propre de A .
Pour tout entier i compris entre 1 et 3, V_i désigne une matrice de $\mathcal{M}_{3,1}(\mathbb{R})$ appartenant au sous-espace propre de la matrice A associé à la valeur propre λ_i et dont l'élément situé en 3^e ligne est égal à 1.
- Justifier que la famille $\mathcal{B} = (V_1, V_2, V_3)$ est une base de $\mathcal{M}_{3,1}(\mathbb{R})$.

4. Comportement asymptotique de la population.

Notons (c_1, c_2, c_3) les coordonnées de N_0 dans la base \mathcal{B} . On admet que $c_1 \neq 0$.

- Pour $i \in \{1, 2, 3\}$, calculer $A^n V_i$ pour tout entier naturel n .
- Montrer que : $\forall n \in \mathbb{N}, N_n = (\lambda_1)^n c_1 V_1 + (\lambda_2)^n c_2 V_2 + (\lambda_3)^n c_3 V_3$.
- En déduire que, pour tout $n \in \mathbb{N}$, on peut écrire

$$N_n = (\lambda_1)^n (c_1 V_1 + E_n)$$

où E_n est une matrice dont tous les coefficients ont pour limite 0 lorsque n tend vers $+\infty$.

- Donner des équivalents de e_n , a_n et s_n .
En déduire les limites des différents rapports $\frac{e_n}{t_n}$, $\frac{a_n}{t_n}$ et $\frac{s_n}{t_n}$ où $t_n = e_n + a_n + s_n$. On admet que $t_n \neq 0$.
- Ces résultats dépendent-ils de la répartition initiale dans les différentes classes ?
- Interpréter les résultats obtenus en termes de répartition de la population en contact avec le virus considéré.

Partie II - Calcul numérique d'un vecteur propre

Dans cette partie, on implémente une méthode numérique permettant de calculer, sous certaines conditions, un vecteur propre d'une matrice. Quelques fonctions utiles sont rappelées en annexe à la fin du sujet.

On rappelle que chaque programme doit être commenté par une phrase détaillant le raisonnement qui a conduit à son élaboration.

On importe les modules suivants :

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

À toute matrice $M = (m_{ij}) \in \mathcal{M}_{m,n}(\mathbb{R})$ on associe un nombre réel positif $\|M\|$, appelé *norme* de M , défini par

$$\|M\| = \max\{|m_{ij}| \text{ pour } i \in \llbracket 1, m \rrbracket \text{ et } j \in \llbracket 1, n \rrbracket\}.$$

Remarquons que $\|M\|$ est toujours strictement positif, sauf lorsque la matrice M est nulle.

- Écrire une fonction `Norme(M)` qui prend en argument un tableau numpy M de taille quelconque et qui calcule et renvoie le nombre $\|M\|$. *Pour cette question, on interdit le recours à une fonction `max` prédéfinie.*
- Écrire une fonction `Normalise(v)` qui étant donnée une matrice colonne $v \in \mathcal{M}_{d,1}(\mathbb{R})$ non nulle renvoie la matrice colonne $\tilde{v} \in \mathcal{M}_{d,1}(\mathbb{R})$ définie par :

$$\tilde{v} = \frac{v}{\|v\|}.$$

On considère une matrice carrée $A \in \mathcal{M}_d(\mathbb{R})$ et une colonne $v_0 \in \mathcal{M}_{d,1}(\mathbb{R})$ telle que la suite $(v_n)_{n \geq 0}$ d'éléments de $\mathcal{M}_{d,1}(\mathbb{R})$ vérifiant la relation de récurrence

$$\forall n \in \mathbb{N}, \quad v_{n+1} = \frac{Av_n}{\|Av_n\|}$$

est bien définie, c'est-à-dire, telle que pour tout entier naturel n , Av_n n'est pas nul.

On admet que si $A \in \mathcal{M}_d(\mathbb{R})$ est diagonalisable et possède les valeurs propres $\lambda_1, \dots, \lambda_d$ avec

$$|\lambda_1| > |\lambda_2| \geq \dots \geq |\lambda_d|,$$

alors les composantes de la suite $(v_n)_{n \geq 0}$ convergent vers les composantes d'un vecteur propre v de A associé à la valeur propre λ_1 , sauf pour quelques choix de v_0 . La probabilité, en choisissant aléatoirement v_0 , de tomber sur l'une de ces exceptions est nulle.

On suppose qu'on dispose d'une fonction `v0(d)` qui prend en argument un entier naturel non nul d et qui renvoie une colonne de $\mathcal{M}_{d,1}(\mathbb{R})$ dont les coefficients sont choisis aléatoirement dans $[0, 1]$.

- Écrire une fonction `VecteurPropre(A, eps)` qui prend en argument un tableau numpy A satisfaisant aux hypothèses décrites ci-dessus et un nombre $eps > 0$ et qui, partant d'un v_0 choisi au hasard, calcule les termes de la suite $(v_n)_{n \geq 0}$ jusqu'à ce que deux termes successifs vérifient $\|v_n - v_{n+1}\| \leq eps$, et renvoie alors la matrice colonne v_{n+1} .

4. (a) Écrire une fonction `matrice(v,w)` qui prend en argument deux tableaux numpy v et w de taille $(d, 1)$ et qui renvoie le tableau numpy de taille $(d, 2)$ constitué des colonnes v et w .
- (b) Écrire une fonction `colineaires(v,w)` qui prend en argument deux tableaux numpy v et w de taille $(d, 1)$ et qui renvoie `True` s'ils sont colinéaires, `False` sinon.
- (c) Écrire alors une fonction `estVecteurPropre(A,v)` qui renvoie `True` si v est vecteur propre de A , `False` sinon.
- (d) Qu'obtient-on en compilant le programme suivant :

```

v = VecteurPropre(A, eps)
print(estVecteurPropre(A, v))

```

Partie III - Un modèle stochastique discret permettant de calculer une probabilité d'infection

1. Un modèle de contamination linéaire.

Dans ce modèle, on considère une chaîne infinie d'individus indexés par \mathbb{N} .

On suppose que :

- dans la configuration initiale, l'individu 0 est infecté et que les autres individus ne le sont pas,
- la contamination est instantanée.

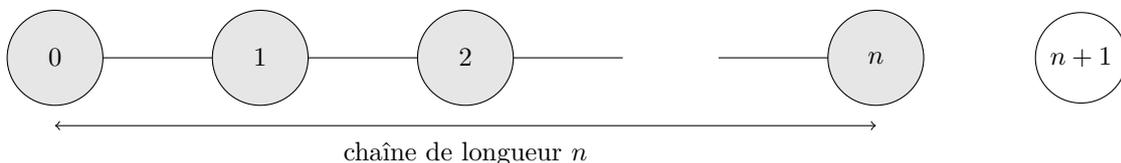
On note p la probabilité que la maladie soit transmise par l'individu n à l'individu $n+1$ et on s'intéresse à la variable aléatoire L égale à la longueur de la chaîne d'infection, c'est-à-dire le numéro du dernier individu infecté.

On introduit pour tout entier naturel n l'événement I_n « l'individu n est infecté par la maladie ».

La configuration initiale :



Une configuration finale (événement $(L = n)$ réalisé) :



- (a) Soit $X = L + 1$ le numéro du premier individu non infecté. Reconnaître la loi de X . En déduire la loi de L .
- (b) Justifier que L admet une espérance et une variance et les déterminer.
- (c) Soit $n \in \mathbb{N}$. En remarquant que $I_n = (L \geq n)$, calculer $P(I_n)$.
- (d)
 - i. Écrire en `Python` une fonction `L(p)` qui prend en argument la probabilité de contamination p et qui renvoie une simulation de la longueur d'une chaîne.
 - ii. Écrire en `Python` une fonction `esperanceL(p)` qui prend en argument la probabilité de contamination p et qui renvoie une valeur approchée de l'espérance de L .

2. Détermination de la probabilité d'infection.

On considère une population infinie comportant des individus sains et un seul individu infecté noté 0.

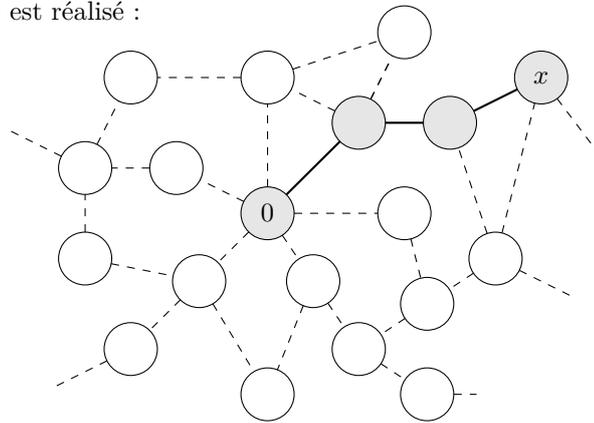
On choisit au hasard un individu x de cette population initialement sain et on souhaite déterminer la probabilité que cet individu x soit infecté à la fin de la journée.

On suppose qu'au cours d'une journée, les individus peuvent ou non se rencontrer, ce qui crée d'éventuelles chaînes de contact entre l'individu 0 et l'individu x . On admet que si l'individu x est contaminé, il l'est par la chaîne de contact la plus courte entre l'individu 0 et l'individu x . On considère alors que la contamination suit le modèle linéaire de la question précédente.

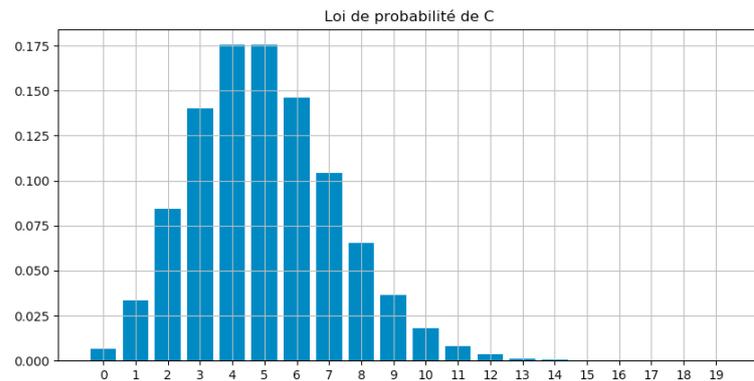
On considère :

- l'événement I « l'individu x est infecté par le virus »,
- la variable aléatoire C égale à la longueur de la chaîne de contact la plus courte entre l'individu 0 et l'individu x , si elle existe. On notera $(C = 0)$ l'événement : « il n'existe pas une telle chaîne de contact ».

Un exemple dans lequel ($C = 3$) est réalisé :



On suppose que C suit une loi de Poisson de paramètre μ où μ est un entier naturel non nul, représentée par l'histogramme suivant :



- Soit $n \in \mathbb{N}^*$. À quelle condition sur n , a-t-on $P(C = n) \geq P(C = n - 1)$?
- En déduire la valeur de μ .
- Utiliser la formule des probabilités totales pour déterminer $P(I)$.

Partie IV - Un modèle stochastique discret permettant de quantifier la propagation de l'épidémie

On considère une population infinie comportant des individus sains et un seul individu infecté noté 0.

On note N la variable aléatoire égale au nombre de personnes qui sont entrées en contact, directement ou indirectement, avec l'individu 0 au cours de la journée.

Pour tout entier naturel n , on pose $p_n = P(N = n)$. On suppose que la suite (p_n) vérifie la relation (R) :

$$(R) \quad \forall n \in \mathbb{N}, \quad p_{n+1} = \left(a + \frac{b}{n+1} \right) p_n$$

où a et b sont des réels tels que N n'est pas nulle presque sûrement, c'est-à-dire tels que $0 < p_0 < 1$.

- Rappeler les propriétés vérifiées par la série de terme général (p_n) .
- Cas particulier** $a = 0$.

(a) Montrer que :

$$\forall n \in \mathbb{N}, \quad p_n = \frac{b^n}{n!} p_0.$$

(b) En déduire p_0 et reconnaître la loi de N .

- On souhaite montrer que $a + b > 0$.

(a) En utilisant l'expression de p_1 démontrer que $a + b \geq 0$.

(b) On suppose que $a + b = 0$. Calculer p_1 puis p_n pour $n \geq 1$. En déduire p_0 et conclure.

4. On souhaite montrer que $a < 1$. Pour cela, on raisonne par l'absurde en supposant que $a \geq 1$.

(a) Montrer que :

$$\forall n \in \mathbb{N}, \quad p_{n+1} \geq \frac{n}{n+1} p_n.$$

(b) En déduire que :

$$\forall n \in \mathbb{N}^*, \quad p_n \geq \frac{p_1}{n}$$

et conclure.

5. Existence et calcul de l'espérance.

Le réel a étant strictement inférieur à 1, on considère α un réel positif tel que : $a < \alpha < 1$.

(a) Montrer qu'il existe un entier naturel n_0 tel que :

$$\forall n \geq n_0, \quad a + \frac{b}{n+1} \leq \alpha.$$

(b) En déduire que :

$$\forall n \geq n_0, \quad p_n \leq \alpha^{n-n_0} p_{n_0}.$$

(c) Justifier alors que N admet une espérance.

(d) Montrer que :

$$E(N) = aE(N) + a + b.$$

On pourra remarquer que $E(N) = \sum_{n=0}^{+\infty} (n+1)p_{n+1}$ et utiliser la relation (R).

(e) En déduire $E(N)$.

6. Quantification de la propagation.

On pose $S = \sum_{n=1}^N X_n$ où les variables X_n suivent la loi de Bernoulli de paramètre $P(I)$.

(a) Que représente la variable aléatoire S ?

(b) Que renvoie la fonction suivante :

```
def mystere(p0, a, b):
    x = rd.random()
    n = 0
    p = p0
    F = 0
    while F < x:
        n += 1
        p = (a+b/n)*p
        F += p
    return n
```

(c) Écrire en Python une fonction $S(p, p_0, a, b)$ qui prend en argument les paramètres p, p_0, a et b et qui simule la variable aléatoire S . On prendra $\mu = 5$.

Annexe - fonctions Python

On rappelle que sous réserve d'avoir fait les importations suivantes :

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

- `np.shape(A)` prend en argument un tableau numpy A et renvoie la dimension de A sous forme d'un tuple.
- `np.zeros((n,p))` prend en argument deux entiers naturels n et p et renvoie le tableau numpy nul de taille (n,p) .
- `np.dot(A,B)` prend en argument deux tableaux numpy A et B et renvoie le résultat du produit AB lorsque celui-ci a un sens.
- `la.matrix_rank(A)` prend en argument un tableau numpy A et renvoie le rang de A .
- `rd.random()` renvoie au hasard un flottant de l'intervalle $[0, 1[$.