COMPLEX - Partie 2
2020-2021

Énoncé

Numéro d'anonymat :

Notes manuscrites et documents de cours autorisés
L'utilisation de tout matériel électronique (en dehors d'une montre non connectée) est interdite

Les exercices sont indépendants.
Une rédaction claire et concise sera appréciée. Toute affirmation devra être justifiée.
Une question non résolue n'empêche pas de faire les suivantes
(dans ce cas indiquez clairement que vous admettez le(s) résultat(s) de la question non faite).

Exercice 1 : Tests groupés probabilistes

Les tests moléculaires par RT-PCR (*reverse transcriptase-polymerase chain reaction*) sont à ce jour les tests de référence pour poser le diagnostic de Covid-19. L'idée des *tests groupés* est qu'ils permettent aux responsables de la santé publique de tester de petits groupes de personnes en n'utilisant qu'un seul test. L'approche consiste à regrouper plusieurs échantillons individuels avec un seul test utilisant la RT-PCR. Si aucun individu du groupe n'est infecté, le test de groupe est négatif et si le test est positif alors au moins un individu du groupe est positif.

Cela signifie qu'il est possible de tester un plus grand nombre de personnes plus rapidement, en utilisant moins de tests et donc pour moins d'argent. L'objectif de cet exercice est de proposer une approche probabiliste efficace pour tester une population donnée en faisant le moins de tests possible et en limitant le temps d'attente des résultats.

Nous considérons une population de n personnes (avec $n \geq 1$ entier) parmi lesquelles au plus d personnes sont infectés (avec $0 \leq d \leq n$) et nous supposons d connu (en pratique cette majoration peut-être estimée par l'incidence du virus dans la population générale). Pour indiquer le statut de chaque individu nous associons un vecteur binaire $X = (x_1, \ldots, x_n) \in \{0,1\}^n$ (où, pour $i \in \{1, \ldots, n\}$, $x_i = 1$ signifie que l'individu d'indice i est malade et $x_i = 0$ qu'il ne l'est pas). Ce vecteur n'est pas connu *a priori* et les algorithmes que nous allons considérer doivent le retrouve en ayant en accès *en oracle* à une fonction de test T qui prend en entrée un sous-ensemble S de $\{1, \ldots, n\}$ et retourne Vrai si il existe un indice $i \in S$ pour lequel $x_i = 1$ et Faux sinon (c.-à-d. si $x_i = 0$ pour tout $i \in S$). La complexité de l'algorithme est estimé en fonction du nombre d'appels à T. Nous supposons que le test T est toujours correct, que le cardinal de S peut être quelconque entre 1 et n et que le matériel biologique pour le test est en quantité suffisamment grande pour permettre de tester autant de fois qu'on le souhaite un individu (en l'incluant dans autant de sous-ensembles S que nécessaire).

1.a] Nous supposons que $d = 1$ (c.-à-d. que la population contient au plus un individu malade). Proposer un algorithme déterministe qui permet d'identifier cet individu (s'il existe) en $\log n + O(1)$ appels à T.

1.b] Nous supposons désormais que d est un entier quelconque de $\{1, \ldots, n\}$ Déduire de la question précédente un algorithme déterministe qui permet d'identifier tous les individus malades en $d \log n + O(d)$ appels à T.

L'algorithme précédent est essentiellement optimal en nombre d'appels à T mais les tests sont effectués séquentiellement et l'algorithme est adaptatif (c.-à-d. les ensembles S soumis à T dépendent des tests précédents). Il est donc difficile à mettre en œuvre et le temps d'attente des résultats est allongé. Nous allons maintenant proposer une famille d'algorithmes qui fonctionnent en une seule phase (c.-à-d. les tests peuvent être parallélisés) et ne sont pas adaptatifs.

Chaque algorithme est défini par une matrice $M = (m_{i,j})_{1\leq i\leq t, 1\leq j\leq n}$ binaire à t lignes et n colonnes où t est le nombre de tests effectués. Chaque ligne décrit les individus qui sont inclus dans le test correspondant de la façon suivante :

$$S_i = \{j \in \{1,\ldots,n\} | m_{i,j} = 1\}$$

pour $i \in \{1,\ldots,t\}$. L'algorithme appelle ensuite $T(S_1),\ldots,T(S_t)$ et obtient un vecteur $Y = (y_1,\ldots,y_t)$ avec $y_i =$ VRAI si S_i contient un individu malade (c.-à-d. si il existe un $j \in \{1,\ldots,n\}$ tel que $x_j = 1$ et $m_{i,j} = 1$) et $y_i =$ FAUX sinon). L'algorithme retourne ensuite le vecteur $Z = (z_1,\ldots,z_n)$ (comme approximation du vecteur X) défini de la façon suivante (pour $j \in \{1,\ldots,n\}$) :

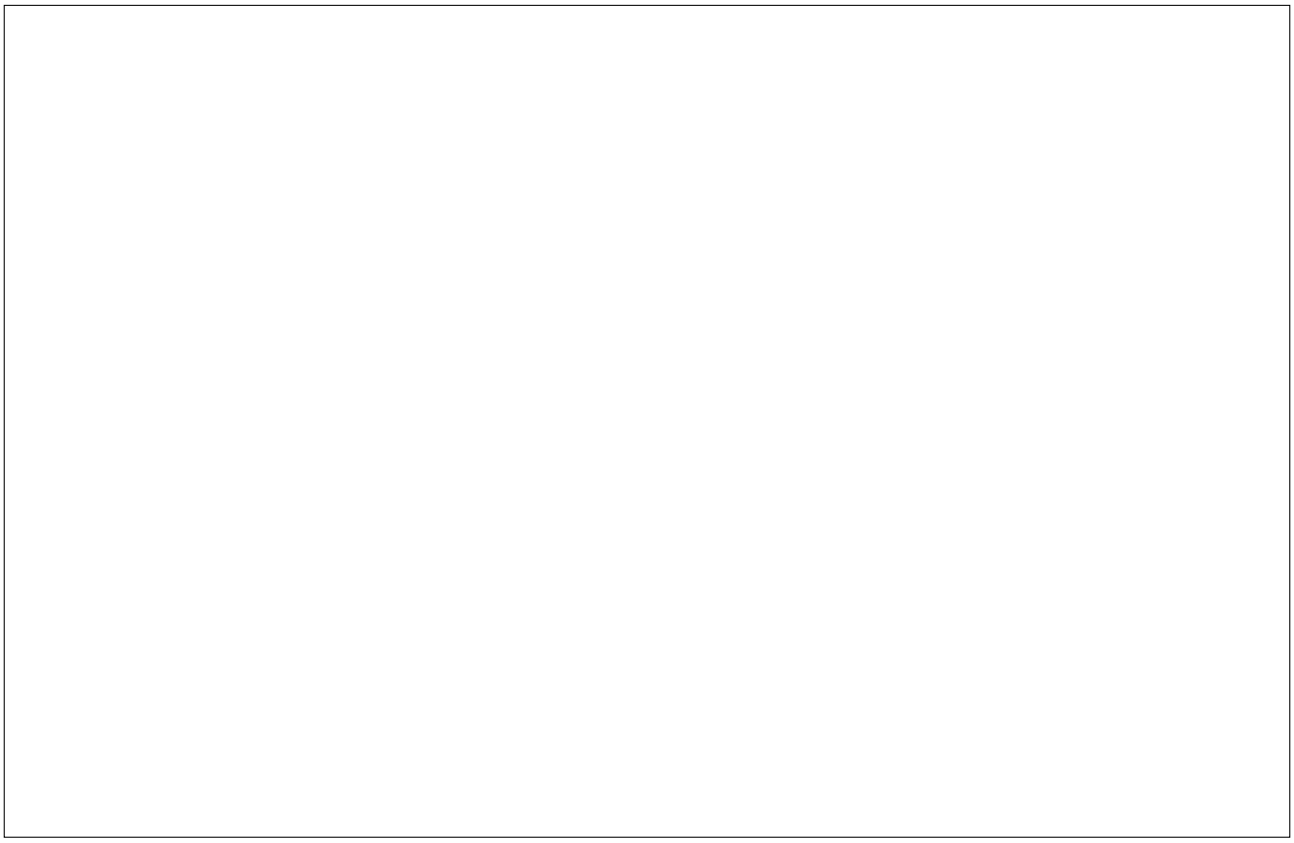
$$z_j = \begin{cases} 1 & \text{si } y_i = \text{VRAI pour tout } i \text{ tel que } m_{i,j} = 1 \\ 0 & \text{sinon} \end{cases}$$

Autrement dit, un individu d'indice i est déclaré positif si et seulement si *tous* les tests sur des sous-ensembles le contenant ont retourné VRAI (en particulier, si un individu n'apparaît dans aucun test, il est déclaré positif).

1.c] Considérons le vecteur $X = (0\ 0\ 1\ 1\ 0\ 0\ 0\ 0\ 1)$ avec $n = 9$, $d = 3$ et les matrices

$$M_1 = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \text{ et } M_2 = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \end{pmatrix}$$

qui correspondent à deux algorithmes effectuant 4 et 6 tests (respectivement). Donner les vecteurs Y_1 et Y_2 obtenus pour ces deux matrices de test puis les vecteurs Z_1 et Z_2 retournés par les deux algorithmes.



1.d] Montrer que les algorithmes de cette famille peuvent retourner des *faux positifs* mais aucun *faux négatif* (c.-à-d. que si $z_j = 0$, alors $x_j = 0$ mais il est possible d'avoir $x_j = 0$ et $z_j = 1$ pour $j \in \{1, \ldots, n\}$).

Soient $p \in [0, 1]$ et $t \geq 1$ un entier (correspondant au nombre de tests). Nous considérons une matrice $M = (m_{i,j})_{1 \leq i \leq t, 1 \leq j \leq n}$ binaire à t lignes et n colonnes tirée aléatoirement de sorte que chaque $m_{i,j} = 1$ avec probabilité p et $m_{i,j} = 0$ avec probabilité $1 - p$ (indépendemment) pour $i \in \{1, \ldots, t\}$ et $j \in \{1, \ldots, n\}$. Nous allons estimer en fonction de p et de t la probabilité que l'algorithme associé à M retourne un faux positif.

1.e] Considérons un individu $j \in \{1, \ldots, n\}$ tel que $x_j = 0$ et une ligne de la matrice d'indicce $i \in \{1, \ldots, t\}$. Montrer que l'événement « le test associé à l'ensemble S_i est négatif **et** $j \in S_i$ » se produit avec une probabilité supérieure ou égale à $p(1 - p)^d$.

1.f] En déduire que la probabilité que l'algorithme associé à une matrice M retourne un vecteur Z différent de X est inférieure ou égale à $n(1-p(1-p)^d)^t$.

1.g] Montrer que la fonction $p \longmapsto (1-p(1-p)^d)$ prend sa valeur minimale en $p = 1/(d+1)$ sur $[0,1]$. Montrer que, pour cette valeur de p, la valeur $(1-p(1-p)^d)$ est inférieure ou égale à $1-1/4d$.

Indication : Pour cette question et la suivante, on pourra utiliser l'inégalité valable pour $x \geq 2$:

$$\frac{1}{4} \leq \left(1-\frac{1}{x}\right)^x \leq \frac{1}{e}$$

1.h] Montrer que pour cette valeur de p et $t = 4d\ln(n)[(1+\delta)]$ avec $\delta > 0$, la probabilité que l'algorithme retourne un faux positif est inférieure ou égale à $n^{-\delta}$.

Exercice 2 : Ensemble indépendant

Soit $G = (V, E)$ un graphe avec n sommets et m arêtes. Un ensemble indépendant dans G est un sous-ensemble $S \subseteq V$ tels que aucune paire de sommets de S ne sont reliés par une arête. Nous considérons l'algorithme probabiliste suivant pour calculer un ensemble indépendant S. Nous posons

$$d = \frac{1}{n} \sum_{v \in V} \deg(v) = \frac{2m}{n}$$

le degré moyen d'un sommet de G. Nous supposons que $d \geq 1$.

Phase 1 Commencer par un ensemble S vide. Ajouter ensuite chaque sommet à S avec probabilité $1/d$.

Phase 2 Le sous-graphe induit par S peut contenir des arêtes et il faut donc au moins retirer un sommet de chacune des arêtes restantes. Pour cela, nous appliquons la stratégie suivante : tant que S n'est pas un ensemble indépendant, nous tirons aléatoirement un sommet u de S qui possède un voisin v dans S et nous supprimons u de S

Par construction, l'algorithme précédent retourne bien un ensemble indépendant de G. Le but de cet exercice est d'estimer la taille de S en fonction de n et de d.

2.a] Donner la complexité de cet algorithme

2.b] Notons X la variable aléatoire correspondant au nombre de sommets de S après la première phase de l'algorithme. Calculer $\mathbb{E}(X)$.

2.c] Notons Y la variable aléatoire correspondant au nombre d'arêtes du sous-graphe induit par S après la première phase de l'algorithme. Montrer que $\mathbb{E}(Y) = n/(2d)$.

2.d] Notons Z la variable aléatoire correspondant au nombre de sommets de S à la fin de l'algorithme. Donner une borne inférieure (en fonction de n et de d) sur $\mathbb{E}(Z)$.

2.e] En considérant la variable aléatoire $T = n - Z$ et en lui appliquant l'inégalité de Markov, montrer que

$$\Pr\left[T \geq n - \frac{n}{4d}\right] \leq 1 - \frac{1}{4d-1}$$

2.f] En déduire que l'algorithme précédent retourne un ensemble indépendant de cardinal inférieur à $n/4d$ avec probabilité au plus $1 - 1/(4d-1)$.

2.g] Montrer comment calculer un ensemble indépendant de taille $n/4d$ avec probabilité supérieure ou égale à $1 - \frac{1}{n}$. Quel est la complexité en temps de votre algorithme probabiliste ?