Examen

Cours de Programmation, R

Master 1 Mathématiques et Applications, specialité IM

8 Janvier 2019

Consignes

— Les élèves ont le droit de se rendre sur internet pour trouver des documentations sur les fonctions à utiliser, mais il leur est strictement interdit d'utiliser la moindre messagerie (mails, Facebook,...) et le moindre service de stockage et de partage de fichiers (Dropbox, etc...). Toute communication avec quiconque sera bien entendue considérée comme une fraude.
— Les élèves demandent l'autorisation d'envoyer le fichier avec leur codes à l'adresse vittorio.perduca@parisdescartes.fr .
— Ils ne quittent pas la salle d'examen avant d'avoir eu confirmation de réception de leur mail
— Les élèves commentent leur code pour en faciliter la compréhension.
— Les exercices sont indépendants et sont en ordre croissant de difficulté.

Exercice 1.

1. Ecrire une boucle calculant les valeurs des vingt premiers termes de la suite de Fibonacci définie par

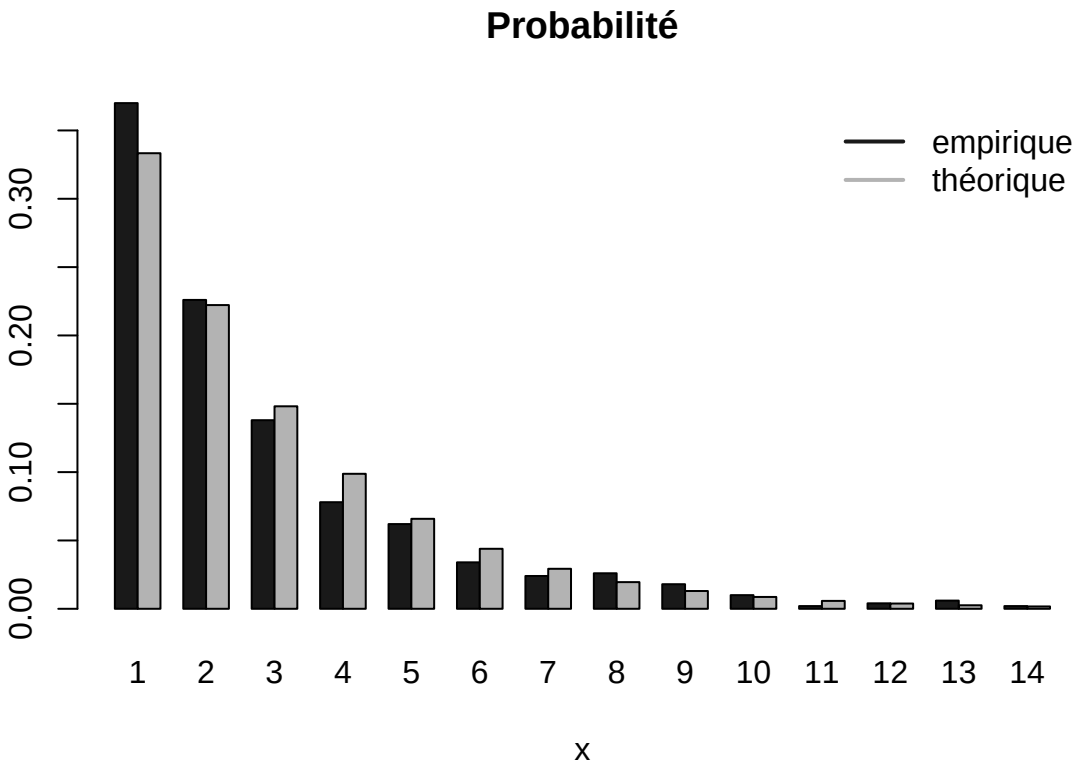
$$u_1 = 0,\ u_2 = 1 \text{ et } u_k = u_{k-1} + u_{k-2},\ \forall k \geq 3 \in \mathbb{N},$$

2. Ecrire une fonction `fibonacci()` qui prend en entrée un nombre naturel n et rend en sortie les termes $u_1, \ldots, u_n$ de la suite de Fibonacci.
3. A l'aide d'un graphique approprié, montrer que le ratio $\frac{u_k}{u_{k-1}}$ converge vers le *nombre d'or* $\varphi = \frac{1+\sqrt{5}}{2}$ quand $k \to +\infty$.

Exercice 2.

1. On considère une pièce de monnaie biaisée avec la probabilité d'obtenir `pile` égale à 1/3. On réalise l'expérience qui consiste à tirer la pièce jusqu'à obtenir pour la première fois `pile`. A l'aide d'une boucle `while`, écrire une fonction `n_essais()` qui réalise l'expérience et compte le nombre n de tirages dans celle-ci. Par exemple si dans une réalisation de l'expérience on a obtenu `face`, `face`, `pile`, la fonction rendra 3 en sortie.
2. Réaliser l'expérience $m = 500$ fois **sans** utiliser de boucle `for`.

3. On sait que n suit une loi géométrique de paramètre 1/3, notée $\mathcal{G}(1/3)$: on a $\mathbb{P}(n = x) = (1-p)^{x-1}p$, pour $x = 1, 2, 3 \ldots$. Coder une fonction `dgeometrique()` qui prend en entrée les paramètres x et p et rend en sortie $\mathbb{P}(n = x)$.
4. Reproduire le barplot groupé suivant, représentant la distribution de l'échantillon $n_1, \ldots, n_m$ obtenu au point 2 ainsi que les valeurs des probabilités théoriques calculées avec `dgeometrique()`. Vous pouvez remplacer les tonalités de gris par les deux couleurs de votre choix.



Exercice 3.

Une méthode pour calculer l'aire A d'un ensemble $S \subset \mathbb{R}^2$ consiste à tirer aléatoirement n points dans le plan et compter la fréquence à laquelle ces points tombent dans S. En particulier, on peut procéder de la façon suivante :

a. On tire n points indépendants (x_i, y_i) selon deux lois uniformes indépendantes $X \sim \mathcal{U}(-w/2, w/2)$ et $Y \sim \mathcal{U}(-w/2, w/2)$, où $w \in \mathbb{R}$ est un paramètre à fixer. On prendra n grand, par exemple $n = 500$.
b. On compte le nombre de points m tombés dans S.
c. On considère l'estimateur

$$\hat{A}_w = \frac{m}{n} w^2.$$

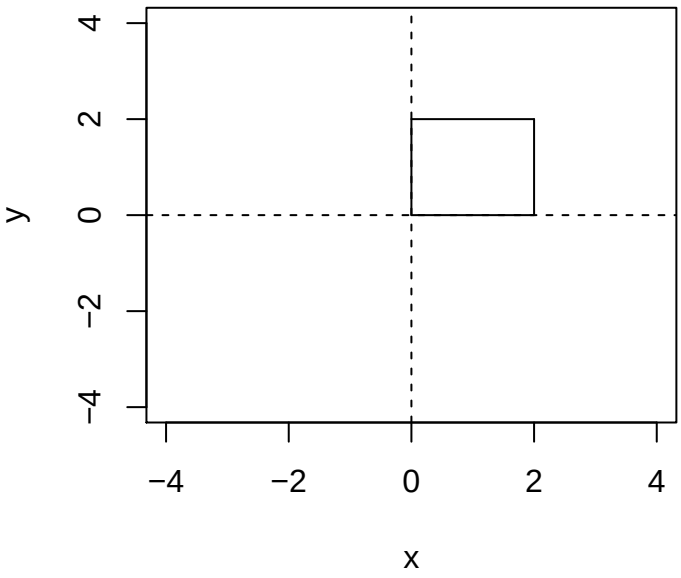
On se propose de calculer par cette méthode l'aire du carrée

$$S = \{(x, y) \,|\, 0 \leq x \leq 2, 0 \leq y \leq 2\}.$$

1. Ecrire une fonction `aire()` qui prend en argument les paramètres w et n et rend en sortie l'estimation $\hat{A}_w$.
2. Estimer l'espérance de $\hat{A}_{w=2}$. Pour cela, on calculera p fois $\hat{A}_{w=2}$ et on prendra la moyenne des valeurs ainsi obtenues. Prendre par exemple $p = 10^3$. Dire si $\hat{A}_{w=2}$ est un estimateur biaisé.
3. Dans un plot avec les axes x, y allant de -4 à 4 et représentant le carré S comme dans la figure ci-dessous, afficher les points $(x_1, y_1), \ldots, (x_n, y_n)$ tirés pour estimer A avec $w = 2$. A l'aide de cette représentation graphique, interpréter la réponse donnée à la question 2 sur le biais de $\hat{A}_{w=2}$. Indications : pour afficher les quatre segments donnant le périmètre de S on utilisera la fonction `segments()`.

4. On considère $w = 4$. Comme dans la question 2, estimer l'espérance de $\hat{A}_{w=4}$. Estimer son écart type.

5. On considère $w = 6$. Comme dans la question précédente, estimer l'espérance et l'écart type de $\hat{A}_{w=6}$. Parmi les trois estimateurs considérés, quel est le "meilleur" en terme de biais et d'écart type ?



Exercice 4.

Le but de cet exercice est de coder une implémentation simple de l'algorithme des k-moyennes pour diviser un ensemble de points de $\mathbb{R}$ en K groupes disjoints afin que chaque point appartienne au groupe dont la moyenne est la plus proche de lui. Plus précisément, on souhaite partitionner les observations $x_1, \ldots, x_n \in \mathbb{R}$ en K groupes $G_1, \ldots, G_K$ de manière à minimiser la quantité

$$\sum_{k=1}^{K} \sum_{x_i \in G_k} |x_i - \mu_k|$$

où μ_k est la moyenne du groupe G_k. L'algorithme des k-moyennes permet de trouver une solution approchée de ce problème et fonctionne comme suit :

a. On initialise les moyennes $\mu_1, \ldots, \mu_k$. Par exemple on pourra tirer k nombres aléatoires dans $[\min(x_1, \ldots, x_n), \max(x_1, \ldots, x_n)]$.

b. On itère plusieurs fois les deux étapes suivantes :
   b.1 : On met chaque point x_i dans le groupe G_k dont la moyenne est plus proche de x_i : Groupe$(x_i) = G_k$ avec
   $$k = \arg \min_{j=1,\ldots,K} |x_i - \mu_j|.$$
   b.2 : On calcule les nouvelles moyennes $\mu_1, \ldots, \mu_k$ des groupes $G_1, \ldots, G_k$ crées à l'étape precedante :
   $$\mu_k = \frac{1}{\#G_k} \sum_{x_i \in G_k} x_i.$$

c. On considère que l'algorithme a convergé quand les moyennes des groupes ne changent plus au cours des itérations, ce qui implique que les groupes ne changent plus.

Première partie

On commence à implémenter l'algorithme pour regrouper en deux classes les observations du vecteur x suivant :

1.4 3.9 2.0 4.4 4.7 20.2 2.6 4.5 2.8 2.3 4.8 2.3 3.4 2.9 0.5 4.5 21.2 0.2 1.6 4.8

1. Ecrire un code qui met x sous la forme d'un vecteur colonne noté `out`.
2. Initialiser l'algorithme en créant un vecteur μ de taille $K = 2$ obtenu en échantillonnant la loi uniforme $\mathcal{U}(\min(x), \max(x))$.
3. **Etape b.1 :** Ecrire un code qui met chaque observation dans le groupe G_k minimisant la distance avec μ_k. Ajouter à `out` la colonne avec les indices des groupes ainsi obtenus (`1` pour les observations dans G_1, `2` pour les observations dans G_2, ...).
4. **Etape b.2 :** Ecrire un code qui crée un vecteur μ' avec les moyennes de chaque nouveau groupe.

Deuxième partie

5. Ecrire un code qui itère les deux étapes ci-dessus jusqu'à ce que les vecteurs des moyennes ne varient pas entre deux itérations successives (autrement dit jusqu'à avoir $\mu_k = \mu'_k$ pour tout k). Si on a bien suivi les instructions ci-dessus, à la fin des itérations on obtiendra une matrice dont la dernière colonne donne le regroupement des observations $x_1, \ldots, x_n$ en deux groupes. Indication : au début de chaque itération on réinitialisera en imposant $\mu = \mu'$.
6. Ecrire une fonction `kmoyennes()` qui prend en entrée un vecteur quelconque d'observations $x = (x_1, \ldots, x_n)$ et le nombre de groupes recherchés K et qui exécute l'algorithme ci-dessus. La fonction rendra en sortie une matrice avec les observations initiales sur la première colonne et les indices des groupes correspondants à chaque itération dans le colonnes successives.