TD sur ordinateur, fiche 1

Révisions, histogrammes, inversion de la fonction de répartition, loi des grands nombres

Avertissement : Ceci est un TD de maths. Le but n'est pas d'apprendre Python auprès d'une matheuse ! Vous avez suivi les cours d'informaticiens infiniment plus qualifiés pour l'enseigner. Le but est de mettre en pratique vos connaissances de probabilités et de statistiques, et d'approfondir votre compréhension des lois et des théorèmes limites.

Ex 1. Loi uniforme

1) Lancer Python. Charger la librairie `numpy` en tapant `from numpy.random import *`. Effectuer des tirages avec `random()`. Que fait cette commande ?

2) Effectuer des tirages de `random()<0.3`. Que fait cette commande ?

3) Effectuer des tirages de `int( random()<0.3 )`. Que fait cette commande ? Quel est le nom mathématique de la fonction qu'on a rajoutée ici (transformation par `int` des booléens) ? Quelle est le nom de la loi de probabilité selon laquelle on tire ici ?

4) Effectuer des tirages de `[int(random()<0.5) for i in range(3)]`. on pourra remplacer 3 et 0.5 par les valeurs de son choix. Que fait cette commande ?

5) Effectuer des tirages de `sum([ int(random()<0.5) for i in range(3) ])`. Quelle est le nom de la loi de probabilité selon laquelle on tire ici ?

6) Quelle est la loi de variables aléatoires comme *le nombre de fois où un dé fait* 6 *en* 10 *lancers* ou *le nombre de piles obtenus en* 1000 *tirages d'une pièce équilibrée*. Simuler de telles v.a. Constate-t-on empiriquement que le nombre de piles en 1000 lancers a une forte probabilité d'être autour de 500 mais une faible probabilté d'être égal à 500 ?

Que signifient les mots *empirique* et *empiriquement* ?

Ex 2. Quelques lois préprogrammées

On a vu qu'à partir de tirages uniformes sur l'intervalle $[0; 1[$ finement discrétisé, on peut faire un tirage d'une Bernoulli, d'une binomiale, etc. En fait *toutes* les lois peuvent être simulées à partir de l'uniforme sur $[0, 1[$. A partir de la fonction `random()`, Python peut donc simuler des tirages de différentes lois classiques.

1) Effectuer des tirages de `binomial(3, 0.5)` et `binomial(3, 0.5, 10)` en variant les paramètres. Que représentent les trois paramètres de cette fonction ? Les résultats sont-ils conformes à ce que vous savez de cette loi, notamment la valeur de son espérance ?

2) Même question pour `poisson(5,10)`.

3) Même question pour `geometric(0.5,10)`.

4) Même question pour `hypergeometric(100,200,9,20)` (attention à la signification des paramètres !).

5) Il n'existe pas de commande `bernoulli()`. Pourquoi ? Par quelle commande fait-on un tirage d'une Bernoulli ?

6) Effectuer des tirages de `normal(0,1,5)`. Faire varier les paramètres et comparer les observations à ce que vous savez de cette loi. Le second paramètre est-il la variance ou l'écart-type ? (Pouvez-vous trouver la réponse empiriquement ?)

7) Effectuer des tirages de `exponential(5,10)` et `exponential(1/5,10)`. Comparer à ce que vous savez de la loi exponentielle. En déduire ce que représente le premier paramètre pour Python. Quel lien avec ce que représente habituellement la paramètre d'une exponentielle ?

8) Expérimenter de même avec `chisquare(3,10)` et `gamma(4,0.5,10)` (question à faire plus tard si ces lois n'ont pas encore été vues).

Ex 3. Simulation des lois de Weibull

Les lois de Weibull sont très utilisées en fiabilité. La loi Weibull(a, b, c) de paramètres $a > 0$, $b \geq 0$ et $c > 0$ est caractérisée par sa fonction de répartition donnée par

$$G_{a,b,c}(t) = 1 - \exp\Big(-\Big(\frac{t-b}{c}\Big)^a\Big) \text{ pour } t \geq b.$$

1) La loi Weibull(a) = Weibull$(a, 0, 1)$ a pour fonction de répartition

$$G_a(t) = 1 - \exp(-t^a), \text{ pour } t \geq 0.$$

Expliquer comment simuler une telle loi par la méthode d'inversion de la fonction de répartition. Effectuer des tirages de loi Weibull$(5, 0, 1)$.

2) De façon générale, il est courant qu'on ait un programme informatique permettant de réaliser des tirages d'une variable aléatoire X de fonction de répartition F_X et qu'on veuille faire des tirages d'une autre variable aléatoire, Y, dont la fonction de répartition F_Y s'en déduit par composition. Par exemple :

$$\forall t \in \mathbb{R} \qquad F_Y(t) = F_X\left(\frac{t-5}{30}\right)$$

Quelle petite modification doit-on faire sur le programme dans ce cas ?

En déduire une simulation de la loi Weibull(a, b, c) de paramètres $a > 0$, $b \geq 0$ et $c > 0$. Effectuer des tirages de la loi Weibull$(5, 2, 10)$.

3) On vient de retrouver la technique préprogrammée de simulation d'une loi classique. Le remarquer en indiquant sous quel nom est plus connue la Weibull$(1, 0, c)$.

Ex 4. Histogrammes

L'*histogramme* d'une loi de probabilité est un graphique représentant la loi sous forme de colonnes verticales. Pour dessiner la loi P_X de la v.a. discrète X, on place une colonne de hauteur $P(X = x_k)$ à chaque abcisse $x_k \in X\Omega$) des valeurs que X peut prendre. Pour des lois non-discrètes, ou pour celles chargeant un grand nombre de valeurs, on place une colonne de hauteur $P(X \in [a; b[)$ au-dessus de l'intervalle $[a; b[$.

1) Dessiner sur papier les histogrammes des lois $\mathcal{B}er(\frac{3}{10})$ et $\mathcal{B}in(3, \frac{1}{2})$.

2) Importer les commandes graphiques en tapant `import matplotlib.pyplot as plt`. La commande `plt.bar([0,1,2], [0.7, 0.1, 0.2])` génère un graphique. Le visualiser en tapant `plt.show()`. Faire afficher successivement les histogrammes des lois $\mathcal{B}er(\frac{3}{10})$ et $\mathcal{B}in(3, \frac{1}{2})$. On pourra effacer l'écran graphique en tapant `plt.clf()`, ou varier les écrans en changeant le paramètre de `plt.show()`.

3) Quand on fait n tirages indépendants d'une v.a. X on obtient un *n-échantillon* X_1, X_2,...,X_n de la loi de X. Tirer un 10-échantillon de la loi $\mathcal{B}er(\frac{3}{10})$ qu'on nommera `tirage`. On peut le lire avec `list(tirage)`.

4) L'*histogramme empirique* d'un n-échantillon est la représentation de colonnes de hauteur égale à la proportion de valeurs égales à x_k dans l'échantillon :

$$\text{Pour chaque } x_k \in X(\Omega) \quad \text{colonne de hauteur } \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}_{X_i=x_k} \text{ à l'abcisse } x_k$$

Que fait la commande `plt.hist(tirage)` ? Visualiser ce graphique pour différents tirages.

5) Contrairement à l'histogramme tout court (histogramme théorique) l'histogramme empirique est aléatoire. Faire tourner plusieurs fois le programme pour $n = 10$ et constater les variations. Que se passe-t-il quand $n = 100$? Quand $n = 1\,000$? $n = 10\,000$? $100\,000$? $1\,000\,000$?

6) Ce que Python affiche par défaut comme histogramme correspond-il à la définition mathématique ci-dessus ? Pour obtenir des colonnes de largeur 1 centrées en les x_k et de hauteur normalisée, on pourra faire `import numpy as np` et rajouter `bins=np.linspace(`a`,`b`,`nb`, density=True)` pour fixer la borne min, la borne max et le nombre total de bornes des colonnes :

```
plt.hist(tirage, bins=np.linspace(-0.5,1.5,3), density=True)
plt.hist(tirage, bins=np.linspace(-0.5,3.5,5), density=True)
```

7) Comparer l'histogramme empirique de chaque échantillon à l'histogramme de la loi correspondante, pour différentes tailles d'échantillon. Que constate-t-on ? Il s'agit d'une propriété générale. A quel théorème est-elle due ?

Ex 5. Convergences

On a vu que les histogrammes empiriques d'un grand nombre de tirages indépendants d'une même loi donnent une représentation approximative (et asymptotiquement exacte) de cette loi.

1) Représenter ainsi la loi $\mathcal{P}ois(5)$.

2) Représenter aussi les lois $\mathcal{B}in(10, \frac{1}{2})$, $\mathcal{B}in(100, \frac{1}{20})$, $\mathcal{B}in(1\,000, \frac{1}{200})$ et $\mathcal{B}in(10\,000, \frac{1}{2\,000})$. Quelle propriété explique le phénomène constaté ici ?

3) Même quetion concernant les lois $\mathcal{H}ypergeom(30, 10, 9)$, $\mathcal{H}ypergeom(300, 100, 9)$, $\mathcal{H}ypergeom(3\,000, 1\,000, 9)$ et $\mathcal{H}ypergeom(30\,000, 10\,000, 9)$. Quelle propriété démontrée et quelle loi sous-jacente apparaissent ici ?

Ex 6. X est une variable aléatoire de fonction de répartition F :

$$\forall t \in \mathbb{R} \quad F(t) = 2^t \ \mathbf{1}_{t<0} + \mathbf{1}_{t\geq 0}$$

1) Déterminer sa fonction quantile.

2) Faire des tirages de X en Python.

3) Calculer l'espérance de X (et la variance si on a le temps). Cela correspond-il aux observations sur les tirages ?

Ex 7. Tirages d'une loi ni discrète ni à densité

La fonction de répartition F_X de la variable aléatoire X prend comme valeurs :

$$F_X(t) = \frac{t}{3} \text{ si } 0 \leq t < 1 \text{ ou } 2 \leq t \leq 3 \qquad\qquad F_X(t) = \frac{1}{2} \text{ si } 1 \leq t < 2$$

1) Quelles sont les valeurs de F_X sur $]-\infty; 0[$ et sur $]3; +\infty[$?

2) Tracer le graphe de la fonction F_X.

3) Déterminer le pseudo-inverse F_X^{-1} de F_X (la fonction quantile de la loi de X).

4) Tracer le graphe de F_X^{-1}.

5) Effectuer des tirages aléatoires selon la loi de X.

Ex 8. Comment simule-t-on par inversion de la fonction de répartition la loi binomiale $\mathcal{B}in(2;1/2)$? Pourquoi cette méthode n'est-elle pas pratique pour simuler une loi binomiale quelconque ? Proposer mieux !

Ex 9. Croissance stochastique des binomiales en fonction du deuxième paramètre

On fixe un entier $n \in \mathbb{N}^*$, un paramètre $p \in]0;1[$, et un autre paramètre $p' \in]0;1[$ tel que $p < p'$. On note F_p la fonction de répartition de la loi $\mathcal{B}in(n,p)$ et $F_{p'}$ celle de la loi $\mathcal{B}in(n,p')$.

1) Rappeler par quelle variable aléatoire on simule la binomiale $\mathcal{B}in(n,p)$ à partir de n tirages $U_1,\ldots,U_n$ de la loi uniforme sur $]0;1[$. On note Z_p cette variable aléatoire. Ecrire à partir des mêmes $U_1,\ldots,U_n$ la variable aléatoire $Z_{p'}$ qui simule $\mathcal{B}in(n,p')$.

2) Si deux événements A et B sont tels que $A \subset B$ alors pour tout $\omega \in \Omega$ on a $\mathbf{1}_A(\omega) \leq \mathbf{1}_B(\omega)$. Justifier cette affirmation.

3) En déduire que l'inégalité $Z_p(\omega) \leq Z_{p'}(\omega)$ est également valable pour tout $\omega \in \Omega$.

4) Trouver une inégalité entre $F_p(t)$ et $F_{p'}(t)$ valable pour tout réel t.

5) Illustrer le résultat trouvé en traçant dans un même repère le graphe des fonctions de répartitions de $\mathcal{B}in(2;1/3)$ et $\mathcal{B}in(2;1/2)$ (on ne demande pas l'expression des fonctions de répartition, mais le tracé des graphes devra être lisible et précis).

Ex 10. Moyenne empirique

1) Tracer le graphe de la fonction de répartition :

$$\forall t \in \mathbb{R} \qquad F(t) = \frac{t+1}{4}\mathbf{1}_{-1\leq t<0} + \frac{t+2}{4}\mathbf{1}_{0\leq t<2} + \mathbf{1}_{t\geq 2}$$

La loi correspondante est-elle discrète ? Est-elle à densité ? (justifier)

2) Calculer la fonction quantile F^{-1} correspondante.

3) Combien vaut l'espérance de la variable aléatoire V définie par $V = F^{-1}(U)$ où U suit la loi uniforme sur $]0;1[$?

4) Programmer l'ordinateur pour faire des tirages de la variable V ci-dessus. On effectue des tirages successifs et on génère ainsi des valeurs aléatoires V_1, V_2, V_3,... Combien vaut approximativement la moyenne des milles premiers tirages ? Comparer la valeur théorique et des valeurs observées.

5) *(question à ne traiter qu'après avoit vu le théorème central limite et Berry-Esséen)* Vous avez donné une valeur proche de la moyenne des milles premiers tirages. Cette valeur est-elle très approximative ou très précise ? Evaluer dans quelle "fourchette" se trouve vraisemblablement la différence entre la moyenne des milles tirages et la valeur donnée précédemment. Que signifie ici "vraisemblablement" ? La "fourchette" calculée est-elle très précise ?

Ex 11. Moyenne empirique d'un min et d'un max de géométriques

La *moyenne empirique* d'un n-échantillon de la loi P_X d'une v.a. X est la moyenne des valeurs aléatoires obtenues par n tirages répétés indépendants de cette loi :

Moyenne empirique $\quad \overline{X}_n = \dfrac{1}{n}\displaystyle\sum_{i=1}^{n} X_i \quad$ avec $X_1, X_2, \ldots, X_n$ indépendantes toutes de même loi que X

1) Rappeler ce qui a été vu en cours sur la moyenne empirique.

2) La commande `mean` de la librairie `numpy` calcule la moyenne empirique d'un échantillon. Calculer plusieurs fois la moyenne empirique d'un 10-échantillon de la binomiale de votre choix. Que constate-t-on ? Faire de même pour des échantillons de plus grande taille. Le phénomène constaté est-il conforme à ce qu'on attend ?

3) Même question pour des échantillons de taille croissante de la loi de Poisson de votre choix, puis de la loi géométrique de votre choix.

4) On a deux v.a. indépendantes $X \sim \mathcal{G}eom(\frac{1}{5})$ et $Y \sim \mathcal{G}eom(\frac{1}{4})$. Trouver informatiquement une valeur approximative de l'espérance des v.a. $\min(X, Y)$ et $\max(X, Y)$.

Indication : utiliser les commandes `min` et `max` de `numpy` sur deux échantillons de X et Y. Que se passe-t-il si on passe en second paramètre 0 ou 1 à ces commandes ?

Ex 12. Générateur de nombres aléatoires (?)

1) Charger `numpy`, taper `seed(0)` puis tirer quelques nombres aléatoires en Python, avec la loi de votre choix. La commande `seed(0)` fixe la "graine" (valeur d'initialisation) du générateur de nombres aléatoires. Essayer avec d'autres valeurs que 0. Cela change-t-il le comportement des tirages ?

2) Pour une même valeur de `seed` et une même loi de probabilité, comparer vos tirages avec ceux d'un/une camarade. Que constate-t-on ? Tirer un échantillon de la loi de votre choix, en tapant `seed(0)` avant chaque tirage. Que se passe-t-il ? Quelle est la raison de ce phénomène ? Quels en sont les inconvénients et les avantages ?

3) Tacer l'histogramme de la fonction `random` pour différentes valeurs de `seed`. Le comportement du générateur aléatoire uniforme sur $[0; 1[$ (qui alimente toutes les autres commandes aléatoires) semble-t-il acceptable ?