Informatique
Préparation à l'oral.

Analyse et probabilités - Sujets

2017

1. D'après Centrale PC ♠

(Planche Officiel de la Taupe Gyroscope 2017 N°147)

On donne $u_0 = 1$ et pour tout entier naturel $n \geq 1$, $u_n = \prod_{k=1}^{n} \frac{2k-1}{2k}$. On définit alors, pour tout entier naturel n : $v_n = u_n \sqrt{n}$.

1. Ecrire une fonction Python `V` qui reçoit en argument un entier naturel `n` et renvoie une valeur approchée de v_n.
2. Ecrire un script Python permettant de représenter graphiquement les v_n pour $0 \leq n \leq 30$.
3. Conjecturer, d'après l'affichage précédent, la monotonie et la convergence de (v_n).
4. Démontrer que la suite (v_n) est monotone.
5. Montrer que la série $\sum \ln \frac{v_{n+1}}{v_n}$ converge et en déduire la convergence de la suite (v_n).
6. A l'aide de la formule de Stirling, calculer $\lim_{n \to +\infty} v_n$.
7. Montrer que la dérivée n-ième de la fonction f définie sur $[0\,;1]$ par $f(x) = \sqrt{1-x}$ est de la forme $\lambda_n (1-x)^{\frac{1}{2}-n}$ où λ_n est un réel que l'on exprimera en fonction de u_{n-1}.

2. D'après Centrale PSI ♠♠

(Planche Officiel de la Taupe Gyroscope 2017 N°166)

Un plateau de type Monopoly comporte 12 cases numérotées de 0 à 11. Le joueur commence par la case 0.
On note Y_n la variable aléatoire associé au numéro de la case sur laquelle se trouve le joueur après le n-ième lancer d'un dé équilibré à 6 faces.
On considère les lancers du dé indépendants.

1. Montrer que pour tout entier n supérieur ou égal à 3, on a : $Y_n(\Omega) = [\![0\,;11]\!]$.
   Donner la loi de Y_0.
2. Ecrire une fonction Python permettant de calculer Y_n (cette fonction recevra comme seul argument l'entier naturel n).
3. Ecrire un script Python affichant les fréquences de $Y_n = k$ avec $n \in \{50, 100, 200, 500\}$ pour 5000 simulations.
4. Exprimer $P(Y_{n+1} = k)$ en fonction des $P(Y_n = i)$ pour $i \in [\![0\,;11]\!]$.
5. Soit $U_n = \big(P(Y_n = 0), P(Y_n = 1), ..., P(Y_n = 11)\big)^T \in \mathscr{M}_{12,1}(\mathbb{R})$.
   Déterminer T tel que $U_{n+1} = T.U_n$ puis exprimer U_n en fonction de U_0.
6. La matrice T est-elle diagonalisable dans $\mathbb{C}$?
   (Indication : on considèrera $J = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & 0 \\ 0 & & \ddots & \ddots & 1 \\ 1 & 0 & \cdots & 0 & 0 \end{pmatrix}$)
7. Ecrire un script Python permettant d'afficher U_n pour $n \in \{5, 10, 20, 50, 100\}$.
   Exécuter le script et en déduire une conjecture. Interpréter.

3. Centrale PSI ♠♠

(Planche Officiel de la Taupe Gyroscope 2017 N°168)

On considère la fonction f définie par : $f(x) = \dfrac{1}{2 - e^x}$ et on note : $a_n = \dfrac{f^{(n)}(0)}{n!}$.

1. Montrer que pour tout entier naturel n non nul, on a : $a_n = \displaystyle\sum_{k=1}^{n} \frac{a_{n-k}}{k!}$.
   Indication : on pourra utiliser $(2 - e^x)\, f(x) = 1$.
2. Ecrire un script Python qui calcule et affiche les a_n pour $n \in [\![0\,;10]\!]$.

3. Tracer les courbes (n, a_n), $\left(n, \dfrac{1}{(\ln 2)^n}\right)$ et $\left(n, \dfrac{1}{2(\ln 2)^n}\right)$.

   Formuler une conjecture. En admettant que cette conjecture soit vraie, donner le rayon de convergence R de la série entière $\sum a_n x^n$. On note $S(x)$ sa somme.

4. On note $S_n(x) = \sum_{k=1}^{n} a_k x^k$. Tracer les courbes représentatives de la fonction f et des fonctions S_n pour $n \in [\![0\,;6]\!]$ sur l'intervalle $[0\,; R[$.

   Formuler une conjecture et la démontrer.

4. ENSAM PSI ♠

(Planche Officiel de la Taupe Gyroscope 2017 N°246 – Exercice II)

On veut tracer la courbe C d'équation $f(x, y) = 0$ où f est la fonction réelle de deux variables réelles définie par :

$$\forall (x, y) \in \mathbb{R}^2,\ f(x, y) = x^4 + 2y^2 + 2xy$$

Les éléments de programme demandés ci-après seront écrits en Python.
Le fichier comporte les deux lignes suivantes :

```
import numpy as np
import matplotlib.pyplot as plt
```

1. Ecrire une fonction `f` recevant en argument un `array` `u` correspondant au couple (x, y) (`u=np.array([x,y])`) et renvoyant un flottant correspondant à $f(x, y)$.
2. Ecrire une fonction `nabla` recevant en argument un `array` `u` et renvoyant un `array` correspondant au gradient de f en (x, y).
3. Ecrire une fonction `points` recevant en arguments :
   1. `A` : `array` correspondant à un couple (x, y) (`A=np.array([x,y])`) tel que $f(x, y) = 0$.
   2. `f` : correspondant à la fonction f.
   3. `pas` : flottant strictement positif correspondant à un déplacement « petit » sur la tangente à C au point de coordonnées (x, y).
   4. n : correspondant à un entier supérieur ou égal à 2

   La fonction `points` devra vérifier que le tableau `A` correspond bien aux coordonnées d'un point de la courbe C.
   La fonction `points` renverra une liste de `array` correspondants aux coordonnées de n points A_k approchant la courbe C et tels que : $\forall k \in [\![0\,; n-1]\!]$, $\overrightarrow{A_k A_{k+1}} = \text{pas} \times \overrightarrow{t_k}$ où le vecteur $\overrightarrow{t_k}$ est un vecteur unitaire tangent à C au point A_k.
4. Obtenir un tracer approximatif de C à l'aide des fonctions précédentes (on choisira $\text{pas} = 10^{-5}$ et $n = 400000$).

5. D'après CCP MP ♠

(Planche Officiel de la Taupe Gyroscope 2017 N°187)

Soit $X_{1,1}$, $X_{1,2}$, $X_{2,1}$ et $X_{2,2}$ quatre variables aléatoires réelles indépendantes et identiquement distribuées qui vérifient : $P\left(X_{i,j}=-1\right)=P\left(X_{i,j}=1\right)=\dfrac{1}{2}$.

On note : $M=\dfrac{1}{\sqrt{2}}\begin{pmatrix} X_{1,1} & X_{1,2} \\ X_{2,1} & X_{2,2} \end{pmatrix}$.

1. Ecrire une fonction Python `Mgen` qui ne reçoit pas d'argument et renvoie une réalisation de la matrice *M*.

2. Déterminer $\mathrm{E}\left(\det\left(M\right)\right)$ et $\mathrm{V}\left(\det\left(M\right)\right)$.

3. Déterminer la probabilité que *M* soit :
   - orthogonale
   - inversible
   - diagonalisable

4. Ecrire un script Python qui effectue 50 000 fois la simulation suivante :
   a. Détermination d'une réalisation de *M*.
   b. Calcul du déterminant de *M* (on ajoutera chaque nouvelle valeur calculée à une liste `LDET` initialement vide).
   c. Mise à jour de compteurs suivant que *M* est, ou non, inversible, orthogonale ou diagonalisable.

   A l'issue de ces simulations, le script affichera la valeur moyenne et la variance des éléments de `LDET`, ainsi que les fréquences des matrices respectivement inversibles, orthogonales et diagonalisables.

2016

1. Centrale PSI ♠♠

(Planche RMS 2016-2017 N°1045)

Pour $n \geq 1$, on pose $I_n = \int_0^{\pi/2} \frac{\sin^2(nx)}{\sin^2(x)} \mathrm{d}x$ et $J_n = \int_0^{\pi/2} \frac{\sin^2(nx)}{nx^2} \mathrm{d}x$.

a) i) Justifier l'existence de I_n.

ii) Ecrire une fonction Python qui calcule I_n. Conjecturer, à l'aide de l'ordinateur, la valeur de I_n (on ne demande pas de preuve).

b) i) Justifier l'existence de J_n.

ii) Ecrire une fonction Python qui calcule J_n.

iii) A l'aide de l'ordinateur, conjecturer la convergence de la suite (J_n) puis la prouver en utilisant I_n.

iv) Justifier l'existence de l'intégrale $\int_0^{+\infty} \frac{\sin^2(x)}{x^2} \mathrm{d}x$ et calculer sa valeur.

2. D'après Centrale PSI ♠♠

(Planche RMS 2016-2017 N°1079)

Soit $x \in \mathbb{R}_+^*$. On considère la suite (u_n) définie par $u_0 = x$ et $\forall n \geq 0, u_{n+1} = \frac{u_n^2}{n+1}$.

a) Ecrire la fonction `Suite(n,x)` en Python qui renvoie $u_n(x)$. Tester pour plusieurs valeurs. Tracer $u(1.6616)$ pour $n \in [\![0\,;30]\!]$ puis $u(1.6617)$ pour $n \in [\![0\,;17]\!]$.

b) Montrer l'équivalence entre les trois propriétés suivantes :

(i) il existe $l \geq 1$ tel que $u_l < 1$ (ii) $(u_n) \to 0$ (iii) (u_n) converge

c) On pose $w_n = \frac{\ln(n+1)}{2^{n+1}}$ et $S_n = \sum_{k=0}^{n} w_k$. Montrer que (S_n) converge. On note δ sa limite. Donner une approximation de δ et de e^{δ}.

d) On pose $v_n(x) = \dfrac{\ln(u_n(x))}{2^n}$. Calculer $v_{n+1}(x) - v_n(x)$. En déduire l'expression de $v_{n+1}(x)$ et $u_{n+1}(x)$ en fonction de S_n et $\ln(x)$.

e) A l'aide des résultats de la question précédente, expliquer les résultats obtenus à la question a).

3. D'après Centrale PSI ♠♠

(Planche RMS 2016-2017 N°1046)

a) Montrer que l'équation $\mathrm{sh}(x) = 1$ admet une unique solution α. Ecrire une fonction Python donnant un encadrement de α d'amplitude inférieure ou égale à $\varepsilon > 0$ donné. Tester votre fonction avec $\varepsilon = 10^{-5}$.

b) Soit (I_n) la suite définie par $I_n = \int_0^{\alpha} \mathrm{sh}(t)^n \,\mathrm{d}t$. Ecrire une fonction Python `suite(n)` donnant des valeurs approchées de $I_1, I_2, ..., I_n$.

c) Trouver une relation de récurrence entre I_n et I_{n-2}.

d) Montrer que la suite (I_n) est convergente et déterminer sa limite.

e) Donner un équivalent de I_n au voisinage de $+\infty$.

f) La série de terme général $(-1)^n I_n$ est-elle convergente ? Si oui, en donner, à l'aide d'un script Python et en justifiant la démarche mathématique sous-jacente, une valeur approchée à 10^{-2} près de la somme.

g) La série de terme général I_n est-elle convergente ?

4. Centrale PSI ♠♠

(Planche RMS 2016-2017 N°1034)

On considère $A = \displaystyle\int_0^1 \frac{x}{\ln(1-x)} \mathrm{d}x$

1. Montrer que cette intégrale converge.
2. Donner une valeur approchée de A à l'aide d'un programme Python.
3. Montrer que $A = \displaystyle\int_0^{+\infty} \frac{e^{-2x} - e^{-x}}{x} \mathrm{d}x$.
4. On note $\forall x > 0,\ F(x) = \displaystyle\int_0^x \frac{e^{-2t} - e^{-t}}{t} \mathrm{d}t$. Montrer que $\forall x > 0,\ F(x) = \displaystyle\int_x^{2x} \frac{e^{-t} - 1}{t} \mathrm{d}t$ et en déduire la valeur de A.

5. Centrale PSI ♠♠

(Planche RMS 2016-2017 N°1079)

On définit une suite de variables aléatoires $(X_i)_{i\in\mathbb{N}^*}$ indépendantes et suivant la même loi : $P(X_i=1)=p$ et $P(X_i=2)=1-p$. On pose, pour $n,\ k\in\mathbb{N}^*$, $S_n=\sum_{i=1}^{n}X_i$ et $Y_k=\inf\left\{n\in\mathbb{N}^*, S_n\geq k\right\}$.

a) Montrer l'existence de Y_k. Ecrire une fonction Python d'arguments *k* et *p* renvoyant Y_k.

b) Ecrire une suite d'instructions Python permettant de trouver une valeur approchée de $m_k=\mathrm{E}(Y_k)$ (on suppose *p* connu). Tracer la courbe définie par les points (k, m_k) pour *k* allant de 1 à 100 et $p\in\{0.1, 0.3, 0.5, 0.7, 0.9\}$.

c) Pour $k\geq 3, n\geq 1$, montrer : $P(Y_k=n)=p\times P(Y_{k-1}=n-1)+(1-p)\times P(Y_{k-2}=n-1)$.

d) Montrer que $\mathrm{E}(Y_k)=p\times\mathrm{E}(Y_{k-1})+(1-p)\times\mathrm{E}(Y_{k-2})+1$.

e) Montrer que $\mathrm{E}(Y_k)\sim kC_p$ quand *k* tend vers l'infini, avec C_p qui ne dépend que de *p*.

6. Centrale PSI ♠♠

(Planche RMS 2016-2017 N°1080)

Deux amis se sont donné rendez-vous à 18 heures. Ils arrivent en retard de *X* et *Y* minutes respectivement. On suppose que *X* et *Y* sont des variables aléatoires suivant la loi uniforme sur $[\![0\,;59]\!]$.

a) A quoi correspond la variable $T=|X-Y|$?

b) Donner la loi de *T*.

c) Ecrire une fonction Python rdv(n) qui renvoie les résultats de n simulations de *T*.

d) *i*) Calculer l'espérance exacte de *T*.

   ii) Donner une approximation de l'espérance avec Python. Commenter l'écart.

e) On pose $n=10^5$. Ecrire un programme qui renvoie approximativement la loi de T à l'aide du programme rdv(n). Commenter les écarts.

7. Centrale PSI ♠♠

(Planche RMS 2016-2017 N°1083)

Pour $t \neq -1$, on pose : $\varphi(t) = \left(f(t), g(t)\right)$ avec $f(t) = \dfrac{t}{1+t^3}$ et $g(t) = \dfrac{t^2}{1+t^3}$.

a) Avec Python, tracer le support de cet arc.
Etudier les symétries, réduire l'intervalle d'étude et déterminer le comportement de φ lorsque t tend vers -1.

b) On remarque la présence d'une boucle. Estimer numériquement la longueur de cette boucle.

c) Trouver une équation cartésienne de la courbe de la forme $F(x, y) = 0$.

d) Montrer que $\varphi(t_1)$, $\varphi(t_2)$ et $\varphi(t_3)$ sont alignés si, et seulement si , $t_1 t_2 t_3 = -1$.

2015

1. Exercice type N°1

1. Rappeler le théorème donnant l'approximation de la loi binomiale par la loi de Poisson.

2. Ecrire une fonction Python `Binom2Poisson` mettant en œuvre l'approximation précédente.

   La fonction `Binom2Poisson` recevra en argument :

   - Les paramètres n et p de la loi binomiale considérée.
   - Le nombre de succès k souhaité.

   Elle affichera :

   - La valeur exacte de la probabilité $\mathrm{P}(\mathrm{X}=k)$.
   - La valeur approchée de cette probabilité.
   - L'erreur relative commise.

   On utilisera les fonctions suivantes du module `math` de Python :

   - `factorial`. Cette fonction reçoit en argument un entier naturel et renvoie la factorielle de cet entier.
   - `exp`. Cette fonction reçoit en argument un nombre et en renvoie l'exponentielle.
   - `pow`. Cette fonction reçoit en argument deux nombres x et y et renvoie x^y.

2. Exercice type N°2

Pour $n \in \mathbb{N}^*$, on pose : $S_{u,N} = \sum_{n=1}^{N} u_n$ où la suite u est définie par : $u_n = \frac{\sin n}{n}$ pour tout entier naturel n non nul.

1. Ecrire une fonction Python `Somme` qui calcule $S_{u,N}$. Cette fonction recevra comme argument un entier naturel non nul `N`. L'exécuter avec $N = 100$, $N = 500$ et $N = 1000$. Que peut-on conjecturer ?

2. On admet que l'on a : $\sum_{n=1}^{N} u_n = \frac{1}{N}\sum_{k=1}^{N}\sin k + \sum_{n=1}^{N-1}\left[\left(\frac{1}{n}-\frac{1}{n+1}\right)\sum_{k=1}^{n}\sin k\right]$.
   Démontrer que $\sum u_n$ converge.

Pour tout entier n supérieur ou égal à 2 et tout réel α strictement positif, on pose
$v_n = \frac{(-1)^n}{n^\alpha + (-1)^{n-1}}$ et $S_{v,N} = \sum_{n=2}^{N} v_n$.

3. Ecrire une fonction Python `Somme2` qui calcule $S_{v,N}$. Cette fonction recevra comme argument un entier naturel non nul `N` et un réel `alpha`. La tester avec `alpha` dans $\left\{\frac{1}{4}\,;\frac{1}{2}\,;1\,;2\right\}$ et $N = 100\,000$ puis $N = 1\,000\,000$.
   Que peut-on conjecturer quant à la nature de la série $\sum v_n$ suivant le réel α ?

4. Calculer la valeur de la somme pour $\alpha = 1$.

3. Exercice type N°3

Deux personnes, A et B, ont rendez-vous en un lieu donné entre 20h et 21h. Chacune décide aléatoirement et indépendamment de l'autre de son instant d'arrivée et aucune n'attendra l'autre plus d'un quart d'heure. On note T_A et T_B les variables aléatoires correspondant aux instants d'arrivée des personnes A et B. On suppose que T_A et T_B suivent des lois uniformes sur l'intervalle $[20\,;21]$.

On s'intéresse à la probabilité que les deux personnes se rencontrent. On note *p* cette probabilité.

a. Ecrire une fonction Python permettant de simuler des séries d'arrivées des deux personnes A et B et d'estimer la probabilité *p*.

b. Calculer la valeur exacte de *p* à l'aide d'une approche graphique.

4. Exercice type N°4

On considère la suite *u* définie par :

$$\forall n \in \mathbb{N} \setminus \{0\,;1\},\, u_n = \prod_{k=2}^{n}\left(1-\frac{1}{\sqrt{k}}\right)$$

1. Ecrire une fonction Python recevant en argument un entier naturel *N* supérieur ou égal à 2 et renvoyant u_N.

2. A l'aide de votre programme, conjecturer la valeur de $\lim\limits_{n\to+\infty} u_n$ puis prouver votre conjecture.

3. Etudier la nature de la série $\sum u_n$.

5. Exercice type N°5

Ce sujet comporte deux parties indépendantes.

Partie N°1

On considère la fonction *f* définie par :

$$f(x) = \int_0^{\frac{\pi}{2}} \ln\left(\cos^2(t) + x^2\sin^2(t)\right)dt$$

Préciser le domaine de définition de *f*.

Partie N°2

Ecrire une fonction Python `fCalc` recevant en arguments un réel *x* et un entier naturel non nul *N* et renvoyant une valeur approchée de $f(x)$ obtenue grâce à l'approximation de l'intégrale par une somme de Riemann.

On testera la fonction `fCalc` avec $x=1$ et $x=10^{-5}$ (pour ce deuxième test, on prendra $N=10^6$).

6. Centrale 2015 Maths2 (PC)

Posons :

$$f(x) = \int_0^{+\infty} \frac{t^x e^{-t}}{1-e^{-t}} dt \qquad \text{et} \qquad g(x) = \sum_{n=1}^{+\infty} \frac{1}{n^{x+1}}$$

1. Etude de *f*
   a) Déterminer le domaine de définition *I* de *f*.
   b) Calculer des valeurs approchées de $f(x)$ pour $x \in [\![1;10]\!]$.
   c) Tracer le graphe de *f* sur *I*.
   d) La fonction *f* est-elle continue sur I ? De classe $\mathscr{C}^2$?

2. Etude de *g*
   a) Déterminer le domaine de définition *J* de *g*.
   b) Calculer des valeurs approchées de $\dfrac{f(k)}{g(k)}$ pour $k \in [\![1;10]\!]$. Emettre une conjecture.
   c) Démontrer la conjecture émise à la question précédente.