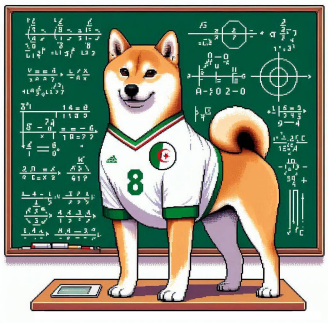
Mathématiques PT. 2025 - 2026

Mathématiques & Informatique - F. Gaunard
http://frederic.gaunard.com
Lycée Voltaire, Paris 11e.



Préparation à l'oral

Math-Info : Math 2
Semaine du 8 Juin

Sujet OB-MATH2-4

Exercice de mathématiques

1. En développant $(1+x)^n$ et $(1+x)^{2n}$, montrer que $\binom{2n}{n} = \sum_{i=0}^{n} \binom{n}{i}\binom{n}{n-i}$.

2. On considère deux variables aléatoires A et B indépendantes et suivant toutes deux une loi binomiale $\mathcal{B}\left(n, \frac{1}{2}\right)$. Calculer $P(A = B)$.

3. On définit la matrice aléatoire M par $M = \begin{pmatrix} 1 & A \\ 1 & B \end{pmatrix}$.
Calculer la probabilité p_n que M soit inversible.

4. On **admet** la formule de Stirling : $n! \underset{n\to+\infty}{\sim} \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$.
Donner un équivalent, puis la limite, de p_n lorsque n tend vers $+\infty$.

Exercice d'informatique

1. Faire tracer les courbes des fonctions $x \mapsto x\log(x) - 2$, où « log » désigne le logarithme népérien, $x \mapsto x^3 - 7x^2 + 11x - 1$ et $x \mapsto 0$ sur l'intervalle $]0, 5]$.

2. La **méthode de Newton** permet de trouver numériquement un zéro d'une fonction g en calculant les premiers termes d'une suite $(x_n)_{n\in\mathbb{N}}$ définie par $x_0 = d$ et $x_{n+1} = \varphi(x_n)$. Rappeler l'expression de la fonction φ en fonction de la variable, de g et de sa dérivée g'.

3. Déterminer la suite de valeurs $[x_0, x_1, \cdots, x_{20}]$ lorsque $x_0 = 3$ et $g(x) = x\log(x) - 2$. À partir de quel indice cette suite est-elle numériquement stationnaire?

4. Recommencer pour $x_0 = 4$ et $g(x) = x^3 - 7x^2 + 11x - 1$.

5. Écrire une fonction `indice` de deux arguments `phi` et `x0` qui renvoie l'indice à partir duquel la suite $(x_n)_{n\in\mathbb{N}}$ est numériquement stationnaire.

6. On va maintenant utiliser la fonction `fsolve` du module `scipy.optimize` de `Python`. Lire l'aide sur cette fonction et expliquer à quoi correspondent les deux premiers arguments de cette fonction.
Utiliser `fsolve` pour résoudre à nouveau les deux équations précédentes.

Solution du sujet OB-MATH2-4

Math (Math II 2019)

1. On peut commencer par voir que

$$(1+x)^n = \sum_{k=0}^{n} \binom{n}{k} x^k$$

et

$$(1+x)^{2n} = \sum_{k=0}^{2n} \binom{2n}{k} x^k.$$

On a aussi

$$(1+x)^{2n} = \left((1+x)^n\right)^2 = \sum_{k=0}^{2n} \left[\sum_{i=0}^{n} \binom{n}{i}\binom{n}{k-i}\right] x^k,$$

par produit de Cauchy.

Par identification des coefficients du terme de degré n, on obtient $\binom{2n}{n} = \sum_{i=0}^{n} \binom{n}{i}\binom{n}{n-i}$.

2. On applique déjà la formule des probabilités totales avec le système complet $\{[B=k] : k \in \mathbb{N}\}$:

$$\begin{aligned}\mathbf{P}(A=B) &= \sum_{k=0}^{n} \mathbf{P}(A=B, B=k) = \sum_{k=0}^{n} \mathbf{P}(A=k, B=k)\\ &= \sum_{k=0}^{n} \mathbf{P}(A=k)\,\mathbf{P}(B=k) \text{ par indépendance}\\ &= \frac{1}{2^{2n}} \sum_{k=0}^{n} \binom{n}{k}\binom{n}{n-k} = \frac{1}{4^n}\binom{2n}{n}\end{aligned}$$

3. M est inversible si et seulement si $\det M \neq 0 \iff = B - A \neq 0 \iff A \neq B$.

On a donc $p_n = \mathbf{P}(A \neq B) = 1 - \mathbf{P}(A=B) = 1 - \frac{1}{4^n}\binom{2n}{n}$.

4. La formule de Stirling donne

$$\frac{1}{4^n}\binom{2n}{n} = \frac{(2n)!}{(n!)^2 4^n} \underset{n\to+\infty}{\sim} \frac{1}{\sqrt{\pi n}} \underset{n\to+\infty}{\longrightarrow} 0.$$

On en déduit que $p_n \underset{n\to+\infty}{\to} 1$.

Info (Math II 2021)

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

1.
```
def f1(x):
    return x*(np.log(x))-2
def f2(x):
    return x**3-7*x**2+11*x-1
def f3(x):
    return 0.
x = np.linspace(0.00001,5,1000)
zero = np.zeros(len(x))
plt.plot(x,f1(x))
plt.plot(x,f2(x))
plt.plot(x,zero)
plt.show()
```

2. D'après la **Méthode de Newton** présentée en classe, $\varphi : x \mapsto x - \dfrac{g(x)}{g'(x)}$.

3.
```
def f1prime(x):
    return 1+np.log(x)
def phi1(x):
```

```
    return x-f1(x)/f1prime(x)
s = [ ]
x = 3
for k in range(0,21):
    s.append(x)
    x = phi1(x)
for e in s:
    print(e) # La suite est numériquement stationnaire à partir du rang 4
```

4.
```
def f2prime(x):
    return 3*x**2-14*x+11
def phi2(x):
    return x-f2(x)/f2prime(x)
t = []
y = 4
for k in range(0,21):
    t.append(y)
    y = phi2(y)
for e in t:
    print(e) # La suite est numériquement stationnaire à partir du rang 8
```

5.
```
def indice(phi,x0):
    r = 0 # indice minimal
    x = x0
    while phi(x) != x:
        r += 1
        x = phi(x)
    return r
```

6.
```
from scipy.optimize import fsolve
help(fsolve)
print(fsolve(f1,3)) # une solution approchée de f1(x)=0 en partant de x0=3
print(fsolve(f2,4)) # une solution approchée de f2(x)=0 en partant de x0=4
```