Banque « Agro-Véto »
A BCPST - 0223

MODÉLISATION MATHÉMATIQUE ET INFORMATIQUE

Durée : 3 heures

L'usage d'une calculatrice est autorisé pour cette épreuve.

Chaque candidate ou candidat est responsable de la vérification de son sujet d'épreuve : pagination et impression de chaque page. Ce contrôle doit être fait en début d'épreuve. En cas de doute, il convient d'alerter au plus tôt l'équipe de surveillance qui vérifiera et, éventuellement, remplacera le sujet.

Ce sujet comporte 9 pages numérotées de 1 à 9.

Si, au cours de l'épreuve, une candidate ou un candidat repère ce qui lui semble être une erreur d'énoncé, elle ou il le signale sur sa copie et poursuit sa composition en expliquant les raisons des initiatives prises.

Ce sujet se compose d'un problème en 3 parties portant sur l'utilisation de marches aléatoires en modélisation.
La partie 1 traite d'une application de la marche aléatoire classique (additive) en physique, dans le cas d'un déplacement libre (sous-partie 1.1) ou contraint dans une boîte (sous-partie 1.2). Cette partie est indépendante des deux suivantes.
Les parties 2 et 3 portent sur l'utilisation d'une marche aléatoire multiplicative (ou modèle binomial recombinant) pour modéliser le cours d'une action économique et estimer le prix d'un contrat sur cette action appelé call. La partie 2 présente l'objet économique et le modèle binomial recombinant de Cox-Ross-Rubinstein (CRR). La partie 3 donne deux façons d'estimer un paramètre de ce modèle appelé volatilité. Les sous-parties 2.3 et 3.2 contiennent des questions algorithmiques.

Un rappel de certaines fonctions Python est donné en annexe.

On pourra admettre le résultat d'une question pour répondre à une question postérieure à condition de le mentionner explicitement.

MARCHES ALÉATOIRES ET APPLICATIONS

Notations

Le coefficient binomial k parmi n désignant le nombre de façons de choisir k éléments parmi n sera noté $\begin{pmatrix} n \\ k \end{pmatrix}$.

La partie entière d'un réel x est notée $\lfloor x \rfloor$.

Le vecteur transposé d'un vecteur $x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$ sera noté x^T.

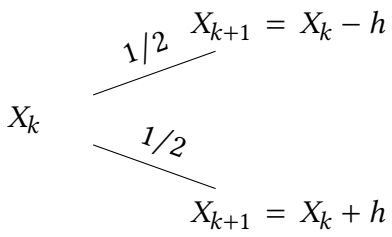
Le produit matriciel entre A et B sera noté $A \cdot B$.

Le produit scalaire usuel de deux vecteurs $\vec{u}$ et $\vec{v}$ de $\mathbb{R}^n$ sera noté $\vec{u} \cdot \vec{v}$.

Partie 1 : Marche aléatoire additive

Sous-Partie 1.1 : Déplacement libre

On considère une particule astreinte à se déplacer sur la droite réelle en effectuant une marche aléatoire de pas $\pm h$, avec $h > 0$ tous les Δt instants. Cette particule peut faire indifféremment un saut à gauche ou à droite avec une même probabilité $\frac{1}{2}$. On suppose qu'à l'instant initial la particule est en $x = 0$. On note X_k la position de la particule au temps $k\Delta t$ c'est-à-dire après k sauts.



1. Donner l'ensemble des valeurs pouvant être prises par X_3, ainsi que sa loi de probabilité. On pourra s'aider d'un arbre de probabilités.
2. Justifier que l'espérance de X_k, pour $k \geq 1$, est nulle.
3. Soit Y_k la variable qui compte le nombre de saut à droite de la particule au temps $k\Delta t$. Quelle est la loi de Y_k ? Justifier.
4. Exprimer X_k en fonction de Y_k.
5. En déduire :
   (a) L'ensemble des valeurs pouvant être prises par X_k pour tout $k \geq 1$, ainsi que sa loi.
   (b) La variance de X_k, pour $k \geq 1$.

Sous-partie 1.2 : Déplacement contraint

On suppose maintenant que cette particule est enfermée dans une boite de n cases de largeur h numérotées de 1 à n. Quand la particule est à gauche, sur la case 1 elle a une probabilité $\frac{1}{2}$ de rester sur place et une probabilité $\frac{1}{2}$ de rebondir à droite et de se retrouver dans la case 2. De manière symétrique si elle est dans la dernière case n elle a une probabilité $\frac{1}{2}$ de rester dans la case n et une probabilité $\frac{1}{2}$ de se retrouver dans la case $n-1$. Dans les autres cas (quand elle est située entre les cases 2 et $n-1$), la particule se comporte comme dans la sous-partie précédente.

Soit X une variable aléatoire prenant ses valeurs dans $[\![1, n]\!]$. On dira qu'un vecteur $\vec{p} = \begin{pmatrix} p_1 \\ \vdots \\ p_n \end{pmatrix}$ est le **vecteur de probabilité** de X si pour tout $i \in [\![1, n]\!]$, $\mathbb{P}(X = i) = p_i$.

Cela nécessite en particulier que pour tout $i \in [\![1, n]\!]$, $p_i \geq 0$ et que $\sum_{i=1}^{n} p_i = 1$.

Soit la matrice

$$M = \begin{pmatrix} 1/2 & 1/2 & 0 & & \cdots & 0 \\ 1/2 & 0 & 1/2 & 0 & & \vdots \\ 0 & 1/2 & 0 & 1/2 & & \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ & & 0 & 1/2 & 0 & 1/2 \\ 0 & \cdots & & 0 & 1/2 & 1/2 \end{pmatrix} \tag{1}$$

6. On note $\vec{p}^{(k)}$ le vecteur de probabilité de X_k pour $k \in \mathbb{N}$. Justifier que ces vecteurs vérifient la relation de récurrence suivante

$$\forall k \in \mathbb{N},\ \vec{p}^{(k+1)} = M \cdot \vec{p}^{(k)} \tag{2}$$

En déduire l'expression de $\vec{p}^{(k)}$ en fonction de M et de $\vec{p}^{(0)}$ pour tout entier $k \geq 0$.

7. Justifier qu'il existe une matrice P inversible et une matrice D diagonale telles que :

$$M = P \cdot D \cdot P^{-1} \quad \text{et} \quad P^{-1} = P^T \tag{3}$$

On notera $\lambda_1, \ldots, \lambda_n$ les coefficients diagonaux de D.

8. On se place, dans cette question uniquement, dans le cas où $n = 3$. Dans ce cas la matrice M vaut

$$M = \begin{pmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 1/2 \\ 0 & 1/2 & 1/2 \end{pmatrix} \tag{4}$$

Montrer que 1 est une valeur propre de M, que l'espace propre associé est de dimension 1 et calculer un vecteur propre associé. On admettra que pour $n \geq 4$, 1 est encore valeur propre de M et que l'espace propre associé est également de dimension 1 (**dans la suite on notera $\lambda_1 = 1$**).

9. On rappelle que la norme euclidienne pour un vecteur est définie par :

$$\|\vec{v}\|_2 = \sqrt{\left(\sum_{j=1}^{n} v_j^2\right)} = \sqrt{\vec{v}^T.\vec{v}} \tag{5}$$

Remarque : Cette norme est notée $\|\cdot\|_2$ et non simplement $\|\cdot\|$ afin de ne pas la confondre avec la norme 1 définie à la question **9.(c)**.

On admettra que $\overrightarrow{v_1} = \frac{1}{\sqrt{n}} \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$ est un vecteur propre associé à la valeur propre 1. Pour $i \in [\![2, n]\!]$, on note $\vec{v_i} = \begin{pmatrix} v_{i,1} \\ \vdots \\ v_{i,n} \end{pmatrix}$ un vecteur propre de norme euclidienne associé à la valeur propre λ_i.

(a) Justifier que pour tout $i \in [\![2, n]\!]$, ces vecteurs vérifient $\overrightarrow{v_1}^T.\vec{v_i} = 0$.

(b) On définit la matrice

$$N = M - \frac{\lambda_1}{\|\overrightarrow{v_1}\|_2^2} \overrightarrow{v_1}.\overrightarrow{v_1}^T \tag{6}$$

Montrer que pour tout $i \in [\![1, n]\!]$, $\vec{v_i}$ est un vecteur propre de N associé à la valeur propre λ_i.

(c) Pour tout vecteur $\vec{v}$ de $\mathbb{R}^n$ on définit :

$$\|\vec{v}\|_1 = \sum_{j=1}^{n} |v_j| \tag{7}$$

Cette fonction présente des propriétés similaires à celles de la norme euclidienne, elle est appelée la norme 1. **On admettra que** :

$$\forall i \in [\![2, n]\!] \ : \ \|N \cdot \vec{v_i}\|_1 < \|\vec{v_i}\|_1 \tag{8}$$

Montrer que pour $i \in [\![2, n]\!]$, $|\lambda_i| < 1$.

10. **(a)** Montrer la limite suivante :

$$\lim_{k \to +\infty} D^k = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 0 & & \\ \vdots & & \ddots & \\ 0 & \cdots & & 0 \end{pmatrix} \tag{9}$$

(b) Exprimer M^k en fonction de D et de P.

(c) En déduire que :

$$\lim_{k \to +\infty} M^k = \frac{1}{n} \begin{pmatrix} 1 & \cdots & 1 \\ \vdots & & \vdots \\ 1 & \cdots & 1 \end{pmatrix} \tag{10}$$

11. On rappelle que pour tout $k \in \mathbb{N}$ on note $\vec{p}^{(k)}$ le vecteur de probabilité de X_k défini à la question **6.** Calculer la limite de $\vec{p}^{(k)}$ quand k tend vers $+\infty$.

12. Interpréter le résultat en terme physique.

Partie 2 : Marche aléatoire multiplicative

Dans cette partie nous proposons d'étudier un second modèle qui peut être vu comme une version multiplicative de la marche aléatoire. Les résultats de cette partie sont entièrement indépendants de la partie 1. La sous-partie 2.3 regroupe des questions Python. La sous-partie 2.2 présente le modèle et l'applique au calcul du prix d'un objet économique défini en partie 2.1, le call.

Sous-partie 2.1 : Contrat à terme

Une meunerie fabriquant de la farine de blé souhaite se prémunir contre une hausse trop importante du prix du blé. Ce prix est modélisé par une variable aléatoire S_n dépendant du temps $n \in \mathbb{N}$. On note s_n une réalisation de la variable aléatoire S_n.

Un **call** sur le prix du blé est un contrat d'assurance payant que l'entreprise passe avec une banque au temps initial $n = 0$ donné. Ce call lui est facturé au prix c_T. En contrepartie, la banque garantit à l'entreprise qu'à une date fixée à l'avance et appelée **le terme**, noté $T \in \mathbb{N}^*$, celle-ci payera au maximum son blé au prix $s^* \in \mathbb{R}^+$, appelé **prix d'exercice** et lui aussi convenu à l'avance. La banque couvre donc les éventuels dépassements par rapport à s^* en prenant à sa charge, s'il y a lieu, et dans ce cas seulement, la différence entre le cours du blé s_T au terme T et le prix d'exercice s^*. Ainsi, au temps T :

— Soit le prix du blé s_T n'a pas dépassé s^* et l'entreprise achète son blé au cours normal.

— Soit le prix du blé s_T a dépassé s^* et l'entreprise achète son blé à la banque au prix s^*.

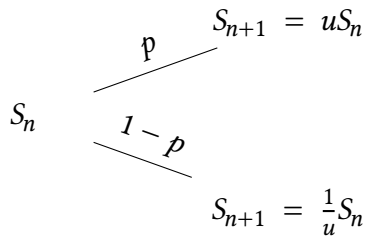
13. Justifier qualitativement que le prix du call c_T dépend de la durée du contrat.

14. Tracer la fonction P_{ent} du prix auquel l'entreprise achète son blé au temps T en fonction du prix du blé s_T

15. Tracer la fonction P_{ban} représentant ce que la banque (qui paie son blé au prix normal) verse à l'entreprise au temps T en fonction du prix du blé s_T

Sous-Partie 2.2 : Le modèle binomial recombinant de Cox-Ross-Rubinstein - application au calcul du prix du call

On suppose qu'il existe $u \in]1, +\infty[$ (pour *"up"*) et $p \in]0, 1[$ deux réels tels que pour tout $n \in \mathbb{N}$ le prix du blé au temps $n+1$ dépend de son cours au temps n (et est indépendant de tous les S_i pour $i \leq n-1$) suivant l'arbre de probabilité suivant :



La question 1 est entièrement indépendante de ce qui précède. Les questions questions suivantes dépendent de la définition du call (sous-partie 2.1).

1. **(a)** Justifier à l'aide d'un arbre que l'ensemble $\mathscr{D}_3$ des valeurs prises par S_3 est égal à :

$$\mathscr{D}_3 = \{u^3 s_0, u s_0, \frac{s_0}{u}, \frac{s_0}{u^3}\} \tag{11}$$

Expliquer l'appellation "modèle recombinant".

(b) Calculer $\mathbb{P}(S_3 = u^3 s_0)$ et $\mathbb{P}(S_3 = u s_0)$.

(c) Donner en justifiant l'ensemble $\mathscr{D}_n$ des valeurs prises par S_n.

2. Soit $n \geq 1$, soit un call de prix d'exercice $s^* \in \mathbb{R}_+^*$ à la date n. Soit G_n la variable aléatoire valant la somme que la banque verse à la meunerie à la date n.

(a) Soit a défini par :

$$a = \left\lfloor 1 + \frac{n}{2} + \frac{\ln\left(\frac{s^*}{s_0}\right)}{2\ln(u)} \right\rfloor \tag{12}$$

On admettra que $a > 0$. Justifier que l'ensemble des valeurs prises par G_n est :

$$\mathscr{D}_{G_n} = \{0\} \cup \{u^{2k-n} s_0 - s^*, k \in \{a, \dots, n\}\} \tag{13}$$

(b) Que représente le coefficient a ?

(c) **Application numérique** : dans le cas où $n = 3$, $u = 2$, $s_0 = 1$, $s^* = 4$ et $p = \frac{1}{3}$ donner les tables de loi de S_3 et G_3.

3. Soit c_n le prix d'un call sur le prix du blé de prix d'exercice s^* et de terme n.

(a) Pour être proposé au prix juste, le prix c_n du call de terme n doit vérifier :

$$\mathbb{E}[G_n] = c_n \tag{14}$$

Interpréter cette égalité.

(b) **Application numérique** : calculer c_3 pour les valeurs de la question **2.(c)**.

(c) Montrer que pour tout $n \geq 1$:

$$c_n = \sum_{k=a}^{n} \left(u^{2k-n} s_0 - s^*\right) \times \left(\begin{array}{c} n \\ k \end{array}\right) p^k (1-p)^{n-k} \tag{15}$$

Sous-partie 2.3 : Calcul effectif du prix du call par Python

La question 1. est indépendante de ce qui précède. Les questions suivantes dépendent des résultats (13) et (15).

4. **(a)** Soit la fonction Python suivante :

```
1  def C(n,k):
2      Num = 1
3      for i in range(2,n+1):
4          Num = Num*i
5      Den = 1
6      for i in range(2,k+1):
7          Den = Den*i
8      for i in range(2,n-k+1):
9          Den = Den*i
10      return (Num/Den)
```

Justifier que cette fonction renvoie le coefficient binomial $\left(\begin{array}{c} n \\ k \end{array}\right)$. Évaluer le nombre d'opérations effectuées par la fonction `C` en fonction de n. Quel autre problème cette fonction pose-t-elle en terme de calcul numérique ?

(b) Compléter en le recopiant le code suivant afin de renvoyer le coefficient binomial $\left(\begin{array}{c} n \\ k \end{array}\right)$ de façon optimisée en nombre d'opérations. Expliquer la démarche suivie.

```
1  def Copt(n,k):
2      Num = ...
3      for i in range(...,...):
4          Num = Num*i
5      Den = ...
6      for i in range(...,...):
7          Den = Den*i
8      return (Num/Den)
```

5. Que renvoie la fonction suivante ? Justifier :

```
1  def Feuilles(n,u,s0,setoile):
2      F = []
3      for k in range(n+1):
4          F.append(max(u**(k*2-n)*s0-setoile,0))
5      return F
```

6. Écrire une fonction `call` qui utilise la fonction `Feuilles` et le résultat (15) pour calculer le prix du call de prix initial $s0$, de prix d'exercice $s*$, de terme n et de paramètres u et p. Justifier.

Partie 3 : Estimation des paramètres du modèle de Cox-Ross-Rubinstein

Dans cette partie nous supposerons que $p = \frac{1}{2}$.

Sous-partie 3.1 : Calcul de la volatilité historique et calibrage du modèle

Dans cette sous-partie nous chercherons à calculer une approximation des valeurs de u ainsi que de la *volatilité* σ, fonction de u définie à la question **8.(a)**. Cette sous-partie utilise le modèle de la partie 2 mais est indépendante des résultats précédents.

7. Pour $n \geq 1$ on pose $Z_n = \ln\left(\frac{S_n}{S_{n-1}}\right)$.

(a) Montrer que les Z_i sont indépendantes et suivent toutes une même loi qu'on précisera.

(b) Calculer l'espérance et la variance de Z_1.

8. Soit $\overline{Z_n} = \frac{1}{n}\sum_{i=1}^{n} Z_i$.

(a) Justifier que lorsque n tend vers $+\infty$ la convergence en loi suivante est vraie :

$$\frac{\overline{Z_n}}{\sqrt{\sigma^2/n}} \xrightarrow{\mathscr{L}} \mathcal{N}(0,1) \tag{16}$$

avec σ un réel strictement positif à exprimer en fonction de u. *Le terme σ s'appelle* ***la volatilité****.*

(b) Que signifie une volatilité élevée dans la modélisation du prix du blé ?

9. **(a)** Soit $n \in \mathbb{N}$. Exprimer la variance de $\overline{Z_n}$ en fonction de la volatilité. En déduire que l'espérance de la variable aléatoire $V_n = \frac{1}{n-1}\sum_{i=1}^{n}(Z_i - \overline{Z_n})^2$ est σ^2. On pourra commencer par montrer que :

$$\mathbb{E}[V_n] = \frac{1}{n-1}\sum_{i=1}^{n}\left(\mathbb{E}[Z_i^2] - 2\mathbb{E}[Z_i\overline{Z_n}] + \mathbb{E}[\overline{Z_n}^2]\right) \tag{17}$$

(b) Que représente V_n ?

(c) Donner une interprétation qualitative de la variable aléatoire $U_n = \exp(\sqrt{V_n})$.

Sous-partie 3.2 : Calcul de la volatilité implicite avec python

Cette sous-partie peut-être vue comme une partie indépendante du reste, les résultats utiles équations (20) et (21) étant rappelés.

Ici nous chercherons à estimer la volatilité non-pas à partir de réalisations du prix du blé, mais à partir de réalisations du call en utilisant le modèle.

Pour trouver un zéro d'une fonction $f(u)$ dérivable (résoudre $f(u) = 0$) on peut utiliser **la méthode de Newton** appelée aussi **méthode de la tangente**. Elle consiste à définir une suite (u_n) par la relation de récurrence :

$$u_{n+1} = u_n - \frac{f(u_n)}{f'(u_n)}. \tag{18}$$

Cette suite converge vers un zéro de f sous de bonnes conditions qu'on ne demande pas de remontrer ici. Numériquement on s'arrête de calculer les termes consécutifs de la suite quand $|u_{n+1} - u_n| < \varepsilon$ ou lorsque le nombre d'itération dépasse un seuil $n_{\max}$. On supposera que les deux fonctions $f(u)$ et $f'(u)$ sont programmées sous Python dans les codes `F` et `Fprime` (on ne demande pas ici la programmation de ces fonctions) :

```
def F(u) :
    f = ...
    return f
def Fprime(u) :
    fp = ...
    return fp
```

10. **(a)** Écrire une fonction `Newton(u0,epsilon,F,Fprime)` renvoyant un zéro de f approché par la méthode de Newton. Cette fonction prendra en entrée u_0, ε, et les codes `F` et `Fprime` de f et de sa dérivée f'. On supposera que $n_{\max}$ est une variable globale.

(b) Choisir en justifiant une valeur de ε cohérente.

(c) Que renvoie cette fonction si $f'(u_n) = 0$? Modifier le code pour renvoyer un message d'erreur dans ce cas.

Étant donnée le prix actuel du marché c^* d'un call donné, la probabilité p étant fixée à $1/2$, on veut estimer la *volatilité* en calculant u vérifiant $c_n(u) = c^*$. Ceci revient à chercher un zéro de la fonction f définie par :

$$f(u) = c_n(u) - c^* \tag{19}$$

On rappelle que :

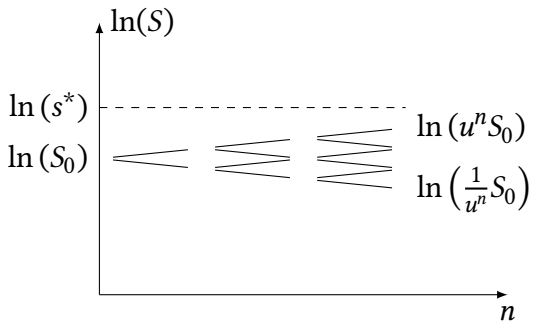
$$c_n(u) = \sum_{k=a}^{n} \left(u^{2k-n} s_0 - s^*\right) \times \binom{n}{k} p^k (1-p)^{n-k} \tag{20}$$

avec :

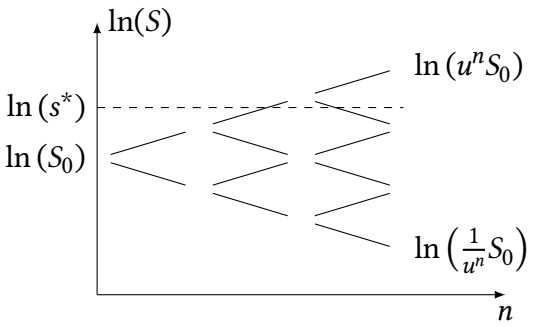
$$a = \left\lfloor 1 + \frac{n}{2} + \frac{\ln\left(\frac{s^*}{s_0}\right)}{2\ln(u)} \right\rfloor \tag{21}$$

On se place dans le cas où $s^* > s_0$.

11. Quelle est la valeur de u minimale (notée u_{min}) pour que $c_n(u)$ soit positif ? On pourra s'aider du graphique suivant :



(a) Pour u petit



(b) Pour u grand

12. Si $u < u_{min}$ la valeur du call est nulle. Donner en une explication qualitative en termes financiers.

13. Exhiber des intervalles pour u sur lesquels a est constant, a variant de n à $\left\lfloor 1 + \frac{n}{2} \right\rfloor$. Quel est l'intérêt de rechercher de tels intervalles ? Expliquer une démarche exploitant l'algorithme de Newton et de tels intervalles pour la recherche d'un zéro de la fonction f.

14. Calculer la dérivée de la fonction f sur ces intervalles.

15. **(a)** Supposant les fonctions `F` et `Fprime` programmées sous Python, que renvoie l'algorithme suivant ? Justifier.

```
1  def mystere(F,Fprime,s0,setoile,epsilon):
2      a = n
3      u = (setoile/s0)**(1/(2*a-n))
4      while F(u)<0 and a>=m.floor(1+n/2):
5        a -= 1
6        u = (setoile/s0)**(1/(2*a-n))
7      a +=1
8      u = (setoile/s0)**(1/(2*a-n))
9      u=Newton(u,epsilon,F,Fprime)
10     return u
```

(b) Dans le cas où $p = 1/2$ on rappelle que :

$$u = \exp\left(\sqrt{\sigma^2}\right) \tag{22}$$

En déduire une fonction **Volatilite** estimant la volatilité σ. Cette volatilité estimée est appelée **la volatilité implicite**.

Annexe - quelques fonctions python

On suppose que le module `math`, qui permet d'utiliser des fonctions mathématiques, est importé avec l'alias `m`. Ce module contient les fonctions

- `floor(x)` qui renvoie la partie entière inférieure de x.
- `exp(x)` qui renvoie la valeur de la fonction exponentielle en x
- `log(x)` qui renvoie la valeur de la fonction logarithme en x

Par ailleurs on rappelle que la valeur absolue de x est donnée par la fonction **`abs`**`(x)`.

FIN DU SUJET