Examen Mag 372 – Modélisation en probabilités 31-03-2026

1. Les exercices sont indépendants. Toutes les réponses doivent être soigneusement justifiées. Les réponses aux questions théoriques doit être préférentiellement sur les copies papier.
2. Vous ne pouvez pas vous connecter à votre session personnelle : à la place, vous vous connecterez à la session `mathens` avec le mot de passe `mathens`.
3. À la fin de l'examen, vous enregistrerez votre feuille de calcul Jupyter dans le dossier `remise_copie` présent sur le bureau. Il faut impérativement inclure votre nom dans le nom des fichiers (donc, sauvegarder un fichier nommé par exemple `NOM.ipynb`).
4. On rappelle les commandes d'import des bibliothèques adéquates :

```
import numpy as np
import scipy.stats as scs
import matplotlib.pyplot as plt
import numpy.random as random
```

Exercice 1. Simulation d'une loi bêta. On s'intéresse à la simulation de variables aléatoires suivant une loi bêta de paramètres $\alpha = 2$ et $\beta = 3$, de densité :

$$f(x) = 12\, x(1-x)^2\, \mathbb{1}_{[0,1]}(x).$$

Partie 1.
On considère une variable aléatoire $X \sim \mathcal{B}(2,3)$ de densité f.

1. Calculer explicitement la fonction de répartition F de X.
2. Quelles hypothèses doit vérifier une fonction $F : \mathbb{R} \to [0,1]$ pour être une fonction de répartition ? Vérifier que c'est bien le cas pour la fonction de la question précédente.
3. Montrer que pour tout $u \in [0,1]$, l'équation $F(x) = u$ revient à résoudre une équation polynômiale de degré 4. Discuter de la possibilité d'obtenir une expression explicite de $F^{-1}(u)$.
4. Écrire sur papier le principe de la méthode par dichotomie pour approximer $F^{-1}(u)$.
5. Écrire en Python une fonction `repartition` qui calcule $F(x)$ pour $x \in \mathbb{R}$.
6. Écrire en Python une fonction `dichotomie` qui approxime $F^{-1}(u)$ par dichotomie. *Indication : on utilisera le critère d'arrêt* $|F(x) - u| < 10^{-6}$. Calculer $F^{-1}(0.8)$.
7. Écrire en Python une fonction `simulation` qui permet de simuler N réalisations de X par inversion. On affichera le résultat pour $N = 100$.
8. Tracer sur un même graphique la fonction de répartition empirique de 1000 valeurs simulées, et la fonction de répartition théorique de X. Commenter.

Partie 2.
On souhaite maintenant simuler $X \sim \mathcal{B}(2,3)$ par une méthode de rejet.

1. Rappeler le principe de la méthode de rejet. On choisira ensuite pour loi d'évaluation la loi uniforme sur $[0,1]$ de densité $g(x) = \mathbb{1}_{[0,1]}(x)$.
2. Déterminer une constante M telle que $f(x) \leq Mg(x)$ pour tout $x \in \mathbb{R}$.
3. Donner la probabilité d'acceptation théorique (c'est-à-dire, la probabilité pour qu'un point simulé (X,Y) soit accepté pour la simulation de la loi).
4. Écrire en Python une fonction `simulation_rejet` qui renvoie N réalisations de X par la méthode de rejet. On affichera le résultat pour $N = 100$.
5. Tracer sur un même graphique la fonction de répartition empirique de 1000 valeurs simulées par la méthode de rejet, et la fonction de répartition théorique de X. Commenter.

Exercice 2. Chaîne de Markov et algorithme PageRank. On considère un petit réseau de pages web constitué de 4 pages. Le PageRank permet de mesurer l'importance relative des pages d'un réseau en modélisant la navigation d'un utilisateur comme une chaîne de Markov. La distribution stationnaire obtenue donne un classement des pages en fonction de leur accessibilité et de la structure des liens. L'ajout de la téléportation garantit la convergence et rend le modèle robuste, même en présence de pages isolées ou de cycles. Les liens entre les pages sont les suivants :

- La page 1 pointe vers les pages 2 et 3.
- La page 2 pointe vers la page 3.
- La page 3 pointe vers les pages 1, 3 et 4.
- La page 4 pointe vers les pages 1 et 3.

On suppose l'équiprobabilité entre les différentes pages pointées à partir d'une page.

1. Construire la matrice de transition P associée à la marche aléatoire décrite ci-dessus.
2. Vérifier que P est une matrice stochastique.
3. Représenter le graphe associé.
4. On note π la distribution stationnaire (ou loi invariante) de la chaîne. Écrire l'équation vérifiée par π.
5. Calculer numériquement une approximation de π.
6. Écrire une fonction Python `trajectoire` simulant une trajectoire de longueur N de la chaîne. On pourra partir de la page 1.
7. Simuler une trajectoire pour $N = 100$, et pour $N = 1000$.
8. Dessiner sur un même graphique l'histogramme empirique des fréquences de visite (pour $N = 1000$ visites), et celui de la distribution stationnaire. Comparer.

Afin d'éviter certains problèmes (pages pièges), on modifie la matrice de transition en introduisant un facteur d'amortissement $\alpha = 0.85$. La nouvelle matrice est alors :

$$P_\alpha = \alpha P + \left(\frac{1-\alpha}{4}\right) Q$$

où Q est la matrice 4×4 dont tous les coefficients sont égaux à 1.

9. Calculer la matrice P_α. Est-ce une matrice stochastique ?
10. Calculer la nouvelle distribution stationnaire.
11. Interpréter le résultat en termes de classement des pages.
12. La distribution stationnaire pour la chaîne associée à P_α est-elle une mesure réversible ?