Feuille de TP n°3 – Vecteurs aléatoires

1 Vecteurs aléatoires avec composantes indépendantes

On veut simuler un vecteur aléatoire $X = (X_1, X_2, X_3)$ de dimension 3, dont les coordonnées sont indépendantes et telles que les lois marginales sont :

— X_1 suit une loi exponentielle de moyenne 2,
— X_2 suit une loi uniforme sur $[0, 2]$,
— X_3 suit une loi de Bernoulli de paramètre $1/4$.

Exercice 1.1. Calculer théoriquement $\mathbb{E}(X)$ et la matrice de covariance K de X.

Exercice 1.2. Faire 10000 simulations de ce vecteur. Par la méthode de Monte Carlo, calculer l'espérance de X (c'est un vecteur de dimension 3) et la matrice de covariance K de X (matrice à 3 lignes et 3 colonnes, on utilisera `cov`). Comparer avec les résultats obtenus dans l'exercice précédent.

Exercice 1.3. Refaire le premier exercice en remplaçant la loi de X_2 par une loi gamma de paramètres 3 et 2 de densité

$$f(x) = 9xe^{3x}\mathbf{1}_{]0,+\infty[}(x),$$

en faisant attention aux paramètres utilisés par `numpy.random.gamma` ou `scipy.stats.gamma`.

2 Vecteurs gaussiens

On se propose ici de générer un vecteur aléatoire gaussien $X = (X_1, X_2)$, de moyenne nulle $m = (0, 0)$ et de matrice de covariance

$$K = \begin{pmatrix} 1 & \rho \\ \rho & 1 \end{pmatrix}$$

où ρ est un nombre compris entre -1 et 1.

On rappelle que la *factorisation de Cholesky* consiste pour une matrice réelle symétrique définie positive M à trouver A matrice triangulaire inférieure telle que $A \times A^T = M$, A^T étant la matrice transposée de A. La commande `numpy.linalg.cholesky` le fait numériquement sous Python.

Exercice 2.1.

1. Calculer une matrice A telle que $A \times A^T = K$ avec Python.
2. Simuler un vecteur gaussien Z de dimension 2, dont les lois sont des gaussiennes centrées et réduites, et dont les composantes sont indépendantes.
3. Poser $X = A \times Z$ (attention c'est la multiplication d'une matrice avec un vecteur).
4. En répétant cette simulation un grand nombre de fois et en utilisant la méthode de Monte Carlo, vérifier que $\mathbb{E}(X) = m$ et que la matrice de covariance de X est bien K.
5. Simuler plusieurs fois le vecteur X et tracer sous forme d'un nuage de points X_2 en fonction de X_1. Observer graphiquement ce qui se produit quand ρ varie.

On se propose maintenant de générer un vecteur aléatoire gaussien $X = (X_1, X_2, X_3)$, dont l'espérance est donnée par $m = (m_1, m_2, m_3)$ et la matrice de covariance par K. On rappelle que la seule contrainte sur Γ est qu'elle soit symétrique et positive. Ici on prendra

$$m = (-1, 0, 2), \quad K = \begin{pmatrix} 14 & 8 & 3 \\ 8 & 6 & -3 \\ 3 & -3 & 26 \end{pmatrix}.$$

Exercice 2.2.

1. En utilisant à nouveau le même algorithme, simuler X.
2. En répétant cette simulation un grand nombre de fois et en utilisant la méthode de Monte Carlo, vérifier que $\mathbb{E}(X) = m$ et que la matrice de covariance de X est bien K.
3. Tracer le nuage de points ainsi créé (en dimension 3), ainsi que les trois nuages de points en dimension 2. En dimension 3, on pourra utiliser le code suivant

```
import matplotlib.pyplot as plt
# Tracé du résultat en 3D
fig = plt.figure()
ax = fig.gca(projection='3d')  # Affichage en 3D
ax.scatter(x, y, z, label='Courbe', marker='d')  # Tracé des points 3D
plt.title("Points 3D")
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
plt.tight_layout()
plt.show()
```