Contrôle de connaissances 2

Régressions linéaires en Python

Dans tout cet exercice, on suppose les importations suivantes :

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

Pour chaque question ci-après, écrire les commandes `python` demandées :

1 Donnez 2 manières de créer `X` le tableau `numpy` comportant les données `[1, 2, 3]`. (3 lignes)

```
X = np.array([1, 2, 3])
# ou
X = [1, 2, 3]
X = np.asarray(X)
```

2 Calculer `a`, `b` le coefficient directeur et l'ordonnée à l'origine de la régression linéaire entre `X` et `Y`. (1 ligne)

```
a, b = np.polyfit(X, Y, 1)
```

3 Créer la **fonction** `yfunc` permettant de tracer les valeurs d'une régression. (2 lignes)

```
def yfunc(x, a, b):
    return a*x+b
```

4 Créer la liste `xliste` découpant l'intervalle entre la plus petite valeur de `X` et la plus grande valeur de `X` en 1000 points. (1 ligne)

```
xliste = np.linspace(min(X), max(X), 1000)
```

5 Créer la liste `yliste` des valeurs de la régression calculées sur `xliste`. (1 ligne)

```
yliste = yfunc(xliste, a, b)
```

6 Tracer le nuage de points de `X` et `Y`. (1 ligne)

```
plt.scatter(X, Y)
```

7 Tracer le nuage de points avec barres d'erreurs, avec `uX` l'incertitude sur `X`, `uY` l'incertitude sur `Y`. (1 ligne ou 2)

```
plt.errorbar(X, Y, xerr=uX, yerr=uY,
             linestyle='None') # bonus
```

8 Tracer la régression linéaire entre `X` et `Y`. (1 ligne)

```
plt.plot(xliste, yliste)
```

9 Comment valider une régression ? (2 éléments)

a – Pour être valide, un régression doit passer par les points et les barres d'erreurs.

b – Pour s'assurer de sa qualité, on peut ensuite regarder la valeur du coefficient de corrélation r^2 : on doit avoir $r^2 > 0{,}99$. $r^2 = 0{,}98$ n'est pas satisfaisant.

10 Citer les 4 étapes pour obtenir la meilleure estimation de a et b ainsi que leurs incertitudes grâce à une méthode Monte-Carlo. Pour obtenir des incertitudes sur a et b, nous allons :

1) faire varier aléatoirement les n couples de valeurs mesurées (x_i, y_i) selon des **lois de probabilité uniformes rectangulaires** de demi-largeur la précision $\Delta(x_i) = \sqrt{3}u(x_i)$, simulée grâce à la fonction `np.random.uniform()` (même chose pour y_i) ;
2) Pour chaque série de mesures simulée, faire la régression linéaire, et obtenir des valeurs de pente a_k et d'ordonnée à l'origine b_k ;

3) La meilleur estimation de a et b sera la moyenne des valeurs calculées ;
4) Les incertitude $u(a)$ et $u(b)$ sur ces moyennes seront l'écart-type expérimental des valeurs calculées.