Annexe TP08 : spécification des bibliothèques utiles

I Bibliothèque numpy

si la bibliothèque numpy est importée sous le nom np on peut utiliser les fonctions suivantes :

écart-type et moyenne d'une liste

soit l une liste

np.std(l) *renvoie l'écart-type de la liste des valeurs de l*

np.average(l) *renvoie la moyenne des valeurs de l*

régression linéaire

soit x et y deux listes de même longueur

np.polyfit(x,y,1)

renvoie une liste de deux valeurs [a,b]

a et b sont les paramètres issus de l'ajustement linéaire de la forme ***y=ax+b*** *sur les données expérimentales x et y*

Rappel utile

soit une liste l , **l[0]** est le premier élément de la liste l

exemple :

```
para= np.polyfit(x,y,1)
a=para[0]
b=para[1]
```

Dans la variable a on stocke le premier élément de la liste para qui est le coefficient directeur de la droite modélisant la relation y=ax+b

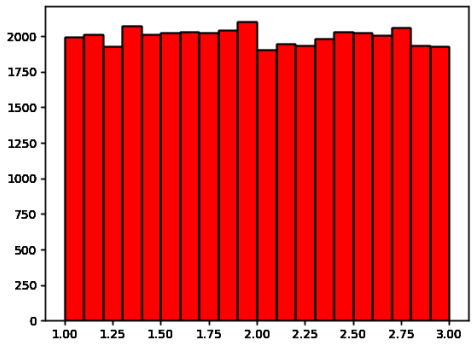
Dans la variable b on stocke le deuxième élément de la liste para qui est l'ordonnée à l'origine de la droite modélisant la relation y=ax+b

Simulation de nombres aléatoires

l=np.random.uniform(1 ,3, 40000)

l est une liste qui contient 40000 valeurs générées aléatoirement et uniformément réparties entre 1 et 3.

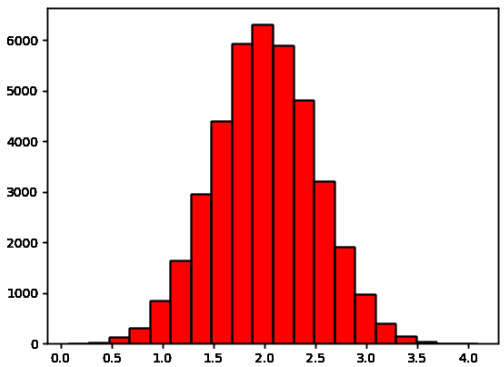
Distribution des valeurs de l



l=np.random.normale(2,0.5, 40000)

l est une liste qui contient 40000 valeurs générées aléatoirement et réparties selon une distribution normale (gaussienne) de valeur moyenne *2 et d'écart-type 0,5*

Distribution des valeurs de l



II Bibliothèque Matplotlib.pyplot

si la bibliothèque matplotlib.pyplot est importée sous le nom plt on peut utiliser les fonctions suivantes :

soit x et y deux listes de même longueur

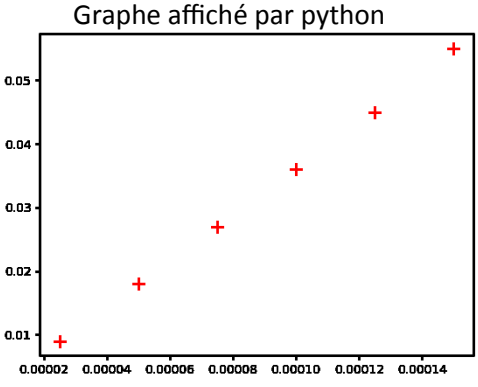
plt.plot(x,y,color='red', marker='+', linestyle='')

plt.show()

permet de tracer le nuage de point représentant y en fonction de x.

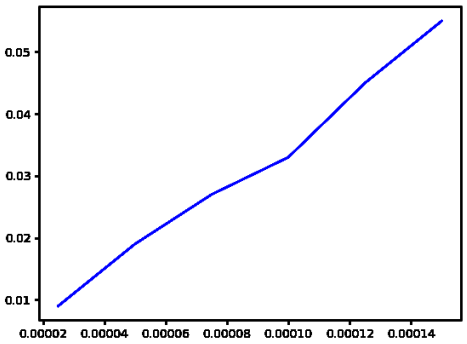
Les points expérimentaux seront représentés par des croix '+' rouges

Graphe affiché par python



plt.plot(x,y,color='blue')
plt.show()
permet de tracer une ligne brisée bleue reliant les points expérimentaux représentant y en fonction de x

Graphe affiché par python



plt.title('bonjour') permet d'ajouter le titre « bonjour » au graphique
plt.xlabel("axe des abscisses") permet d'ajouter un nom (ici « axe des abscisses ») à l'axe des abscisses.
plt.ylabel("axe des ordonnées") permet d'ajouter un nom à l'axe des ordonnées

Principe de la méthode de Monte-Carlo pour estimer l'incertitude-type sur les paramètres issus d'une régression linéaire

On dispose de données expérimentales x et y telles que l'incertitude-type sur les mesures de x est négligeable devant celle sur les mesures de y (notée u(y)).

On souhaite vérifier une relation affine entre y et x et déterminer l'incertitude-type sur les paramètres a et b de l'ajustement affine : **y=ax+b**

- On va simuler de façon aléatoire grâce au programme python un grand nombre (noté N) de données y **qu'on aurait pu obtenir compte tenu de l'incertitude-type sur y.**

- Pour chaque liste des valeurs de y simulées (y_{simu1} , y_{simu2} , ….. $y_{simu\,N}$) on réalise un ajustement linéaire pour trouver les paramètres de la relation affine attendue entre les valeurs de y simulées et les valeurs de x mesurées :

  $\mathbf{y_{simu1} = a_1\,x + b_1}$
  $\mathbf{y_{simu2} = a_2\,x + b_2}$
  .
  .
  .
  $\mathbf{y_{simuN} = a_N\,x + b_{2N}}$

- On obtient deux listes de paramètres a_{simu}= [a_1 , a_2 …., a_N] et b_{simu} = [b_1, b_2 , ….. b_N]

Ce sont les valeurs des paramètres qu'on aurait pu obtenir compte tenu de la variabilité de y

On peut considérer que la valeur de a correcte est la moyenne des valeurs simulées et que l'incertitude-type sur a est égale à l'écart-type des valeurs simulées de a

On peut faire un raisonnement similaire pour b