Thème: statistiques (moyenne, écart type)
Niveau: 2nde

TI-83 Premium CE

python

En attendant 1, 2, 3 sigma ...

F. GIROD

Compétences visées

- **chercher**, expérimenter – en particulier à l'aide d'outils logiciels ;
- **calculer**, appliquer des techniques et mettre en œuvre des algorithmes ;
- **communiquer** un résultat par oral ou par écrit, expliquer une démarche.

Ces compétences sont mises en œuvre au regard de l'extrait du programme de 2nde GT ci-dessous :
« Pour des données réelles ou issues d'une simulation, lire et comprendre une fonction écrite en Python renvoyant la moyenne m, l'écart type s, et la proportion d'éléments appartenant à $[m - 2s , m + 2s]$. »

Situation déclenchante

Pour bien comprendre les formules statistiques et prendre un peu de recul ...

Comment distinguer dans une série numérique de grande taille représentant des données (a priori réelles) les valeurs aberrantes des valeurs extrêmes ?

Problématique

Créer des fonctions en python permettant de :

- calculer la moyenne d'une série de nombres ;
- calculer l'écart type d'une série de nombres ;
- calculer la proportion de valeurs dont la distance à la moyenne est inférieure à deux écarts types.

Thème: statistiques (moyenne, écart type)

Niveau: 2nde

TI-83 Premium CE

python

F. GIROD

Fiche méthode

Proposition de résolution

Pour ce travail, il paraît opportun d'utiliser le type « liste[ ] » de python. Les fonctions utiles seront détaillées au fur et à mesure de la fiche.

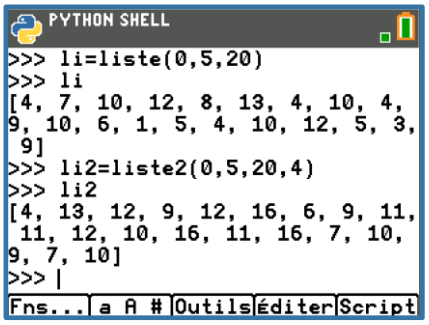
Pour éviter d'avoir à saisir des listes de nombres à la main, on crée au préalable une fonction permettant de générer une liste de n nombres aléatoires entiers à partir de la fonction native **randint**().

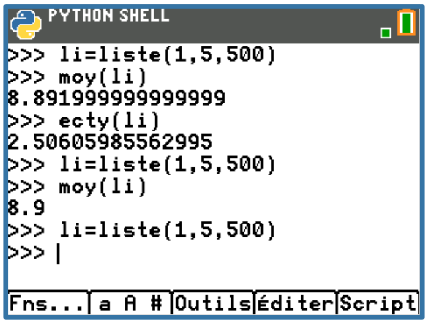
- Sur la copie d'écran ci-contre, la fonction **liste**(0,5,20) renvoie une liste de 20 valeurs de nombres entiers compris entre 0 et 15 ; cette liste est stockée dans la variable li.

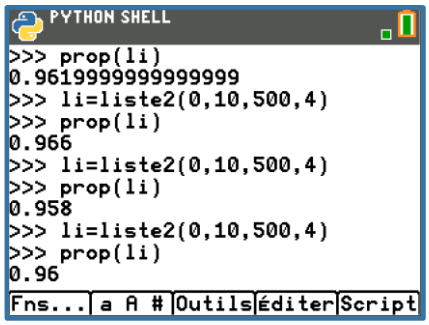
  li est ensuite affichée (c'est à déconseiller par la suite lorsqu'on aura beaucoup de valeurs).

  liste2(0,5,20,4) génère quant à elle une liste de 20 nombres entiers avec un mode plus « élaboré » décrit plus tard.

- On crée la fonction **moyenne**() qui renvoie la moyenne des valeurs d'une liste, et la fonction **ecty**() qui en renvoie l'écart type. Ces deux fonctions ont comme paramètre une liste.
- On teste enfin si une valeur de la liste est à une distance inférieure de deux écarts types par rapport à la moyenne et on donne par la fonction **prop**() la proportion de valeurs de la liste pour lesquelles c'est le cas.
- Pour effectuer plusieurs essais, il est commode d'utiliser la flèche directionnelle (vers le haut) qui rappelle les instructions précédentes.



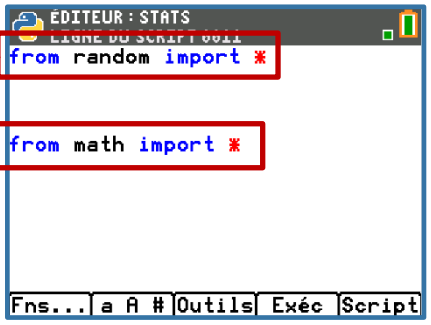




Remarque

importation en préambule du code de la bibliothèque « random » par « **from random import *** » pour pouvoir utiliser la fonction **randint()**

importation de la bibliothèque « math » par « **from math import *** » pour pouvoir utiliser les fonctions **sqrt()** (racine carrée) et **fabs**() (valeur absolue)



Pour profiter de tutoriels vidéos, Flasher le QRCode ou cliquer dessus

Thème: statistiques (moyenne, écart type)

Niveau: 2nde

TI-83 Premium CE

python

F. GIROD

Fiche méthode

Etapes de résolution

fonction **liste**(a,b,n) ; elle génère une liste de n valeurs entières comprises entre 3a et 3b

```
ÉDITEUR : STATS
LIGNE DU SCRIPT 0010
from random import *
from math import *

def liste(a,b,n):
  li=[]
  for i in range(n):
    li.append(randint(a,b)+randi
      nt(a,b)+randint(a,b))
  return li

Fns... | a A # | Outils | Exéc | Script
```

`li=[]` définit une variable de type liste et l'initialise en une liste vide.
`.append()` permet d'ajouter à la liste écrite juste avant les valeurs mises en paramètre : ici, on ajoute « randint(a,b)+randint(a,b)+randint(a,b) » à la liste li.

fonction **moy**() : elle a comme paramètre une liste

`sum()` renvoie la somme des valeurs de la liste entrée en paramètre.
`len()` renvoie le nombre de valeurs de la liste entrée en paramètre.

fonction **ecty**() : elle a comme paramètre une liste

```
EDITEUR : STATS
LIGNE DU SCRIPT 0022
def moy(liste):
  return sum(liste)/len(liste)

def ecty(liste):
  v=0
  n=len(liste)
  m=moy(liste)
  for i in range(n):
    v=v+(liste[i]-m)**2
  v=v/n
  return sqrt(v)

Fns... | a A # | Outils | Exéc | Script
```

`list[i]` représente la (i+1)ème valeur de la liste. En effet, la première valeur de la liste est list[0]

La fonction **ecty**() utilise la fonction **moy**() précédemment définie, ce qui rend son écriture proche de la formule du cours.

Il faut comprendre le rôle joué par la variable **v** (v pour variance) qui ajoute au fur et à mesure que l'on parcourt la liste les valeurs « $(x_i\text{-}m)^2$ ».

fonction **prop**() : elle a comme paramètre une liste

```
ÉDITEUR : STATS
LIGNE DU SCRIPT 0041
def prop(liste):
  c=0
  n=len(liste)
  m=moy(liste)
  s=ecty(liste)
  for i in range(0,n):
    if fabs(liste[i]-m)<2*s:
      c=c+1
  return c/n

Fns... | a A # | Outils | Exéc | Script
```

Pour des raisons de lisibilité, on a choisi de noter : n=len(liste), m=moy(liste), s=ectyp(liste), ce qui permet d'avoir un test écrit de manière proche de l'écriture naturelle :

« si | valeur – *m* | < 2*s*, alors ajouter 1 à *c* »

Pour profiter de tutoriels vidéos, Flasher le QRCode ou cliquer dessus

Thème: statistiques (moyenne, écart type)

Niveau: 2nde

TI-83 Premium CE

python

Fiche méthode

F. GIROD

Retour à la situation déclenchante

Lorsqu'une liste est issue d'une situation réelle (valeurs données par un capteur, valeurs issues d'un sondage, etc.), il se peut que certaines soient erronées.

S'il est possible de remonter à la source, on pourra éventuellement corriger la valeur (erreur de saisie par exemple), sinon, il sera pertinent de l'éliminer.

Comment examiner ces valeurs ?

Il faut les traiter au cas par cas, et le travail précédent va être un atout de poids !

En effet, on examinera les valeurs inférieu res à $m - 2\,s$ et celles supérieures à $m + 2\,s$ et on fera fonctionner son esprit critique ! pour distinguer des valeurs acceptables qui seraient extrêmes de valeurs aberrantes.

Si par exemple un sondage traite de la taille des individus (en cm), la valeur 1.78 aura sans doute été saisie par quelqu'un répondant en m. On pourra la modifier.

liste(50,70,100) génère 100 nombres entiers compris entre 150 et 210 : ils peuvent simuler un échantillons de tailles d'adultes. On observe les valeurs extrêmes, hors de l'intervalle [m – 2s ; m + 2s]

EDITEUR : STATS
LIGNE DU SCRIPT 0061

```
def test(liste):
  examine=[]
  n=len(liste)
  m=moy(liste)
  s=ecty(liste)
  for i in range(n):
    if fabs(liste[i]-m)>2*s:
      examine.append(liste[i])
  return examine
```

Fns... | a A # | Outils | Exéc | Script

PYTHON SHELL

```
>>>
>>> li=liste(50,70,100)
>>> li.append(1.78)
>>> test(li)
[1.78]
>>> li=liste(50,70,100)
>>> test(li)
[158, 157, 202, 203, 158]
>>>
```

Fns... | a A # | Outils | éditer | Script

Remarque

On peut améliorer le processus permettant de générer une liste à partir de la fonction native **randint()** par la fonction ci-contre qui va ajouter r fois des nombres aléatoires entiers compris entre a et b. Elle fournit une liste de n nombres dont la distribution est proche d'une loi « normale ».

fonction **liste2**(a,b,n,r) générant une liste de n valeurs entières comprises entre a.r et b.r en ajoutant r fois un nombre issu de randint(a,b).

ÉDITEUR : STATS
LIGNE DU SCRIPT 0019

```
def liste2(a,b,n,r):
  li=[]
  for i in range(n):
    val=0
    for i in range(r):
      val=val+randint(a,b)
    li.append(val)
  return li
```

Fns... | a A # | Outils | Exéc | Script

Pour profiter de tutoriels vidéos, Flasher le QRCode ou cliquer dessus !

Thème: statistiques (moyenne, écart type)

Niveau: 2nde

TI-83 Premium CE

python

F. GIROD

Fiche méthode

Un peu de théorie

Pourquoi avoir créé des fonctions pour générer des listes de valeurs ?

On aurait pu écrire le code ci-contre, générant une liste directement à partir de **randint**() : on obtient n nombres entiers répartis 'uniformément' entre a et b.
Mais alors, toutes les valeurs appartiennent à l'intervalle [m - 2s , m + 2s] ; on va le montrer ici.

Si X est une variable aléatoire qui suit la loi de probabilité suivante :

a_i	1	2	…	n
$P(X=a_i)$	$\frac{1}{n}$	$\frac{1}{n}$		$\frac{1}{n}$

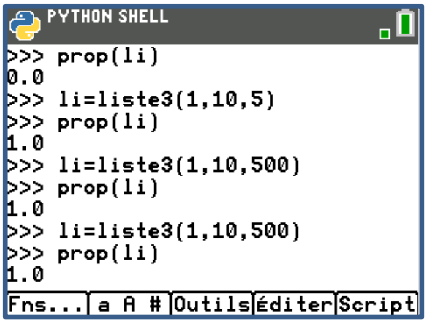
Son espérance mathématique est : $E(X)=\frac{1}{n}\sum_{k=1}^{n}k=\frac{1}{n}\cdot\frac{n(n+1)}{2}=\frac{n+1}{2}$

La variance se calcule par :

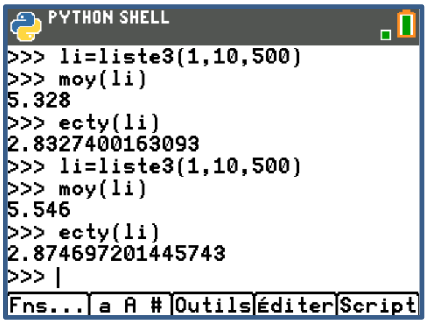
$$V(X)=\frac{1}{n}\sum_{k=1}^{n}\left(k-\frac{n+1}{2}\right)^2=\frac{1}{n}\left(\sum_{k=1}^{n}k^2-(n+1)\sum_{k=1}^{n}k+\sum_{k=1}^{n}\frac{(n+1)^2}{4}\right)$$

$$V(X)=\frac{1}{n}\left(\frac{n(n+1)(2n+1)}{6}-2(n+1)\frac{n(n+1)}{4}+n\frac{(n+1)^2}{4}\right)=\frac{n^2-1}{12}$$

Au final, l'écart type est égal à : $\sigma(X)=\sqrt{V(X)}=\sqrt{\frac{n^2-1}{12}}$



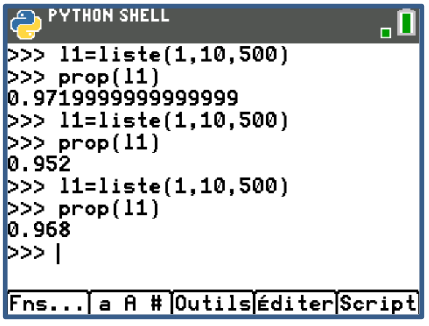
On peut confronter ces résultats à la moyenne et l'écart type obtenu à partir de **liste3**(1,10).
Théoriquement, la moyenne vaut 5,5 et l'écart type $\sqrt{\frac{99}{12}}\approx 2,87$



Ainsi, une telle liste ne présente pas d'intérêt quant à la proportion d'éléments distants de moins de deux écarts-types de la moyenne.
En effet, pour $n \geq 1$,

$$E(X)-2\sigma(X)=\frac{n+1}{2}-2\sqrt{\frac{n^2-1}{12}}\leqslant 1$$

$$E(X)+2\sigma(X)=\frac{n+1}{2}+2\sqrt{\frac{n^2-1}{12}}\geqslant n$$



Ce qui montre que toutes les valeurs comprises entre 1 et n sont dans l'intervalle $[E(X)-2\sigma(X)\,;\,E(X)+2\sigma(X)]$

C'est pourquoi on a créé les fonctions **liste**() et **liste2**() qui génèrent des listes s'approchant de distributions 'normales' présentant plus d'intérêt.

Pour profiter de tutoriels vidéos, Flasher le QRCode ou cliquer dessus !