FICHE TECHNIQUE 16

Utiliser le langage Python pour le traitement statistique de données

Énoncé

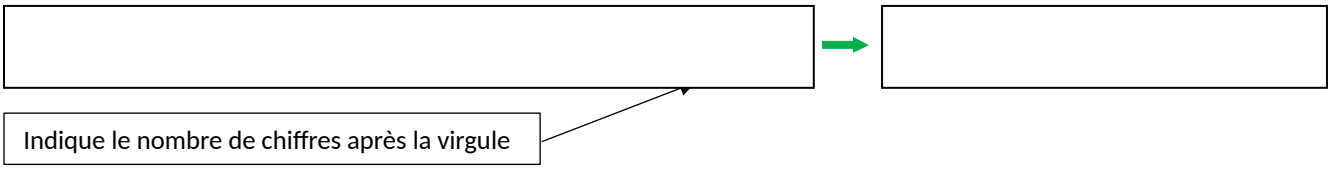
Le traitement statistique élémentaire de données avec le langage de programmation Python peut se réaliser avec la bibliothèque de fonctions mathématiques **Numpy**. À partir de séries de données stockées sous forme de listes, on peut réaliser des calculs statistiques simples comme celui d'une moyenne ou d'un écart type, etc.

Il est également possible de modéliser l'évolution d'un nuage de points par une fonction mathématique.

Technique

Calculer la moyenne et l'écart type d'une série de valeurs

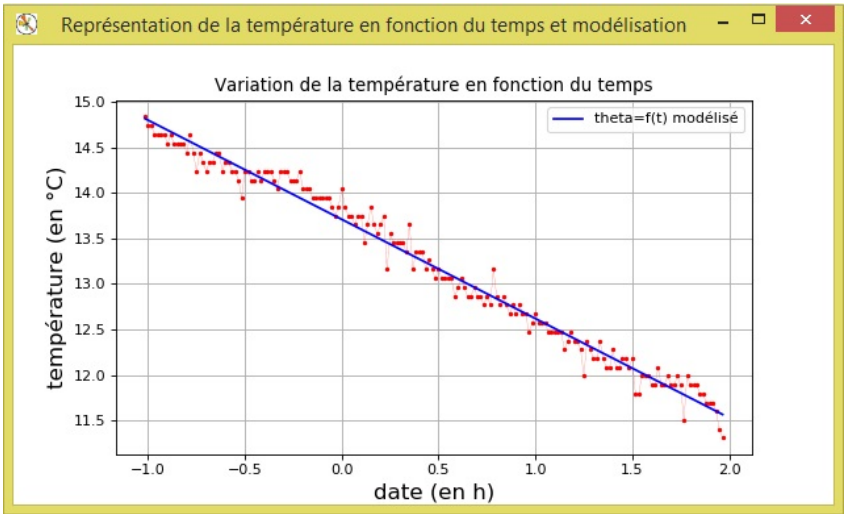
- Calculer la moyenne d'une série de valeurs est une opération qui peut se réaliser en écrivant un programme ou en utilisant la fonction **tableau.mean()** de la bibliothèque **Numpy**, où **tableau** représente un tableau de valeurs que l'on peut obtenir à partir d'une liste avec la conversion suivante : **tableau = numpy.array(liste)**.
- Pour calculer un écart type qui permet de quantifier la dispersion d'une série de mesures, on peut utiliser la fonction **numpy.nanstd()**. Pour utiliser d'autres fonctions statistiques de la bibliothèque **Numpy**, on peut se référer au site : http://www.python-simple.com/python-numpy-scipy/statistics-numpy.php



Modéliser l'évolution d'un nuage de points par une fonction mathématique

- Un nuage de points est caractérisé au minimum par deux listes associées aux deux coordonnées des points. On peut modéliser l'évolution d'une des coordonnées en fonction de l'autre par une fonction mathématique polynomiale y de degré n : $y = a_0 + a_1x + a_2x^2 + ... + a_nx^n$, en utilisant la fonction **numpy.polyfit(x,y,n)** qui renvoie les valeurs des coefficients a_n.

```
# Modélisation du nuage de points par la droite d'équation theta_mod=a*t+b. Calcule les coefficients de la droite
# modélisant le nuage de points et les range dans un tableau nommé Modele
Modele = numpy.polyfit(t,theta,1)
# Affecte les coefficients du modèle aux variables a et b
a,b = [coef for coef in Modele]
# Pour chaque valeur ti du temps, calcule l'ordonnée donnée par la modélisation et les range dans une liste theta_mod
theta_mod = [a*ti+b for ti in t]
# Trace les points de coordonnées t et theta_mod en bleu et reliés
plt.plot(t,theta_mod,'b-',label='theta=f(t) modélisé')
# Affiche l'équation de la droite en arrondissant les coefficients a et b à 1 décimale près
print('Expression du modèle')
if (round(b,1)==0.0):
    print('fonction linéaire : theta(en °C) = ',round(a,1),'x t (en mn)')
else:
    print('fonction affine : theta(en °C)=',round(a,1),'x t (en mn) +',round(b,1))
plt.legend()                        # Affiche la légende
plt.show()                          # Affiche la figure
```



Shells

Python

```
Expression du modèle
fonction affine : theta(en °C)= -1.1 x t (en mn) + 13.7
```

Pour s'entraîner

- À l'aide du fichier texte de données **01_04_19.txt** disponible sur le site compagnon, écrire un programme Python pour déterminer la valeur moyenne de la température dans la nuit du 1er au 2 avril 2019, entre 23 h et 2 h, ce qui correspond aux données situées entre les lignes n° 17 (15 mn + 2 lignes d'en-tête du fichier) et n° 197 (180 + 15 + 2) du fichier texte de données.
- Compléter ensuite ce programme pour modéliser l'évolution de la température en fonction du temps sur la même plage horaire.