éduSCOL
Informer et accompagner les professionnels de l'éducation

Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE

MINISTÈRE DE L'ÉDUCATION NATIONALE ET DE LA JEUNESSE

VOIE GÉNÉRALE ET TECHNOLOGIQUE

2DE | 1RE | TLE

Physique-chimie

ENSEIGNEMENT COMMUN

PROGRAMMER EN PYTHON
FICHE N°1 : PREMIERS GRAPHES

Cette activité a pour but d'exploiter une chronophotographie de la chute d'une bille.

Capacité numérique mise en œuvre : représenter les positions successives d'un système modélisé par un point lors d'une évolution unidimensionnelle ou bidimensionnelle à l'aide d'un langage de programmation.

Premiers tableaux

Sur une chronophotographie (photo imprimée ou vidéo ouverte dans un logiciel de pointage), pointer les positions successives de la balle et relever les positions successives dans un tableau.

Dans l'éditeur (spyder ou jupyter), les positions successives de la bille vont être saisies puis exploitées avec python. Pour ce faire, deux bibliothèques doivent être importées :

- l'une pour faire du calcul numérique appelée numpy ;
- l'autre pour tracer des graphiques, matplotlib, dont nous utiliserons la fonction pyplot.

La pratique répandue est de les renommer à l'aide de noms plus courts.

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

Un tableau des positions mesurées sur la chronophotographie est créé à l'aide de la fonction np.array() en entrant les ordonnées successives de la balle comme suit :

```
ymes=np.array([-0,-0.7,-1.5,-2.3,-3.5,-4.5,-5.9,-7.7,-8.8,-10.6,-12.3,-14.2,-16.4,-18.5,-21,-23.5])
```

Remarque : en langage Python, les virgules sont des séparateurs d'objets et les points sont les séparateurs des décimales. Il est possible de vérifier la construction du tableau via la commande print.

```
print('ymes:',ymes)
```

```
ymes: [  0.   -0.7  -1.5  -2.3  -3.5  -4.5  -5.9  -7.7  -8.8 -10.6 -12.3 -14.2
 -16.4 -18.5 -21.  -23.5]
```

Une base de temps, c'est-à-dire un tableau des valeurs de t, est ensuite créé. Ce tableau peut être construit de manière automatique, à l'aide de la fonction np.linspace. Les trois arguments utilisés sont le début, la fin, le nombre de pas. Ici, l'expérience dure 0,25 s et correspond à 16 points de mesure : il faut donc engendrer 16 instants entre de 0 à 0,25 s (durée de l'expérience). On peut commander une impression dans la même cellule, pour vérifier la bonne saisie de la commande.

```
t=np.linspace(0,1/4,16)
print('t:',t)
```

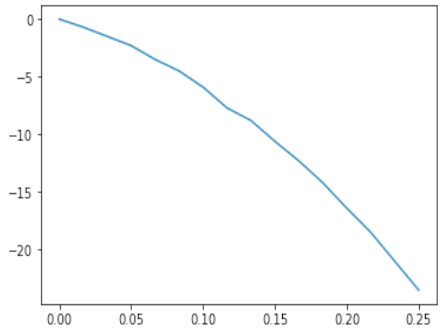
```
t: [0.         0.01666667 0.03333333 0.05       0.06666667 0.08333333
 0.1        0.11666667 0.13333333 0.15       0.16666667 0.18333333
 0.2        0.21666667 0.23333333 0.25      ]
```

Premier graphe

Le tracé des positions mesurées en fonction du temps est généré grâce à matplotlib.pyplot.

```
plt.plot(t,ymes)
```

```
[<matplotlib.lines.Line2D at 0x119b4f470>]
```

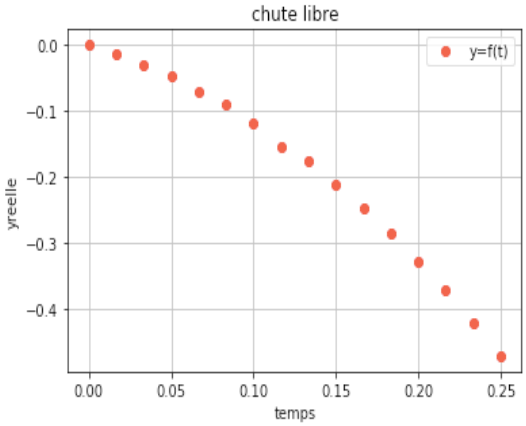


Python gère automatiquement les échelles, les couleurs, les axes.

Manipulation de tableau et tracé de la position calculée

L'ordonnée réelle est enfin calculée à partir de l'ordonnée mesurée sur la chronophotographie. L'échelle de la photo étant de 2/100, numpy permet de multiplier directement toutes valeurs du tableau ymes par l'échelle et d'automatiser la conversion. Les nouvelles valeurs sont rassemblées dans le tableau yreelle. Les fonctions utilisées ensuite servent à « personnaliser » le graphe : étiquettes pour l'abscisse et pour l'ordonnée, grille, légende et titre.

```
yreelle=ymes*2/100
plt.plot(t,yreelle,'ro',label="y=f(t)")
plt.xlabel("temps")
plt.ylabel("yreelle")
plt.grid()
plt.legend()
plt.title("chute libre")
plt.show()
```



Remarque : la commande plt.plot peut être enrichie de divers arguments (comme ici avec ro = cercle rouge) :

Couleur : « r » (red), « k » (black), « b » (blue), « y » (yellow), « g » (green)

Marqueur : « o » (gros point), « - » (ligne), « . » (pointillé), « x » (croix), « s » (square), « v » (triangle)