Utilisation de python en physique

Savoir faire 1 : utilisation de fonctions

Exemple:

def g(t):

——return 3*np.sin(t)

$g(t) = 3\sin(t)$

Exemple:

def f(x,y):

——return x**2+4*y

$f(x,y) = x^2 + 4y$

f(2,3) renvoie $2^2 + 4\times 3 = 16$

f(x,1) renvoie $x^2 + 4$

f(2,y) renvoie $4 + 4y$

Savoir faire 2 : utilisation de listes et des vecteurs

Syntaxe concernant les listes	*Illustration*
Une liste l s'écrit de la forme l=[...,...,...]	l=[1,3,5,7]
len(l) : désigne le nombre de termes dans la liste	len(l)= 4
l[0] : désigne le 1er terme de la liste	l[0]= 1
l[i] : désigne le i+1 terme de la liste	l[2]= 5
l.append(i) : sert à ajouter un terme dans la liste, ce terme a pour valeur i	l.append(9) renvoie l= [1,3,5,7,9]

Syntaxe concernant les vecteurs	*Illustration*
Un vecteur s'écrit v=np.array([...,...,...])	v=array([2,4,6,8])
len(v) : désigne le nombre de termes dans le vecteur	len(v)= 4
v[0] : désigne le 1er terme du vecteur	v[0]= 2
v[i] : désigne le i+1 terme de la liste	v[2]= 6
v=np.zeros(N) crée un vecteur contenant N zéros	v=np.zeros(2) renvoie array([0,0])

Remarque: intérêt d'une liste: on peut à tout moment ajouter des termes, c'est très utile lorsque l'on ne connaît pas à l'avance le nombre de termes de la liste.

Remarque: intérêt d'un vecteur: on peut appliquer une fonction f aux différents termes d'un vecteur (on ne peut pas le faire avec les termes d'une liste).

v est un vecteur de la forme array([v[0],v[1],...]), f(v) renvoie un vecteur de la forme array([f(v[0]),f(v[1]),...])

Exemple 1:

```
lt=[0] # on crée une liste lt qui possède 1 seul élément: 0
dt=0.1
N=100
for i in range(N): # i = 0, 1, 2, ... 99
——lt.append(lt[i]+dt)  on ajoute N = 100 termes à la liste
```

Exemple 2:

```
lt=[0] # on initiale 2 listes
lx=[1]
g,dt,N=10,0.2,500
for i in range(N): # i = 0, 1, ... N-1
——lt.append((i+1)*dt)
——lx.append(lx[i]+g*lt[i]**2)
```

$lt = [0, dt, 2dt, \dots N dt]$

$lx = [1, \dots, lx[i] + g\,lt[i]^2, \dots]$

$i=0$: lt.append(lt[0]+dt) $lt = [0, dt]$

$i=1$: lt.append(lt[1]+dt) $lt = [0, dt, 2dt]$

$lt = [0, dt, 2dt, \dots, Ndt]$

Exemple 3:

```
v=np.zeros(4)
for i in range(len(v)):
——v[i]=3*i
```

Exemple 4:

```
v=np.zeros(3)
v[0]=2
for i in range(len(v)-1):
——v[i+1]=v[i]+2*i+3
```

Exemple 5:

```
v=np.array([1,2,3,4])
def f(x):
——return x**2
v1=f(v)
```

Savoir faire 3 : tracer une courbe

La syntaxe à utiliser, donnée dans l'énoncé d'un sujet de concours, est:

plt.plot(x,y) : x et y sont deux listes (ou deux vecteurs) contenant le même nombre de termes. Python trace les points de coordonnées $[x_i, y_i]$ et les relie entre eux par des segments

plt.xlabel('variable en x'): permet d'écrire le nom de la variable sur l'axe des abscisses

plt.ylabel('variable en y'): permet d'écrire le nom de la variable sur l'axe des ordonnées

plt.title('titre') : permet d'écrire le titre du graphe

plt.grid() : permet d'ajouter un quadrillage pratique pour la lecture des coordonnées des points

plt.show() : permet d'afficher le courbe

Cas 1: tracé de points de mesures expérimentaux

x=[x_1, x_2, ..., x_N] : liste de valeurs de la variable x

y=[y_1, y_2, ..., y_N] : liste de valeurs de la variable y

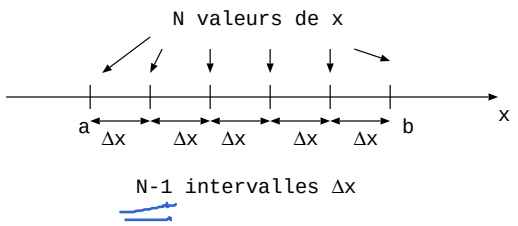
plt.plot(x,y,'o') : trace les points expérimentaux représentés par des •

plt.plot(x,y,'*') : trace les points expérimentaux représentés par des $*$

Cas 2: tracé d'une fonction $f(x)$ sur un intervalle $[a, b]$

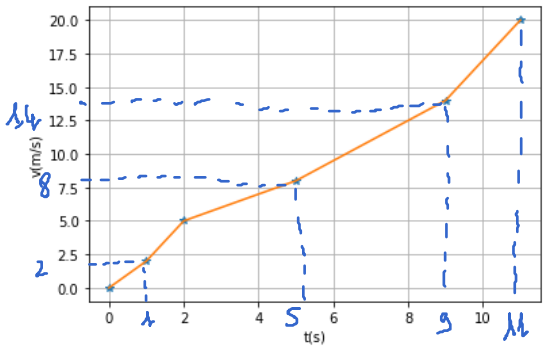
Pour créer le vecteur avec les valeurs en x, les deux possibilités les plus fréquentes reposent sur l'utilisation de np.linspace ou np.arange. Ensuite on obtient le vecteur contenant les valeurs de y en appliquant la fonction f au vecteur x.

x=np.linspace(a,b,N) : crée un vecteur contenant N valeurs régulièrement espacées entre a et b. L'intervalle Δx entre deux valeurs consécutives est $\Delta x = \dfrac{b-a}{N-1}$: on parle d'échantillonnage.



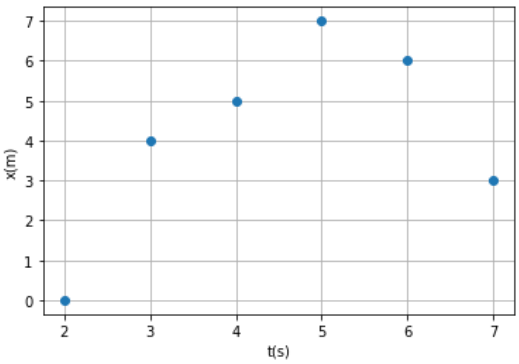
x=np.arange(a,b,p) : crée un vecteur dont le premier terme est a, le dernier terme est inférieur ou égal à b et où p désigne le pas. Le vecteur est donc de la forme array([a,a+p,a+2p,...,a+Np $\leq b$])

Exemple 1:



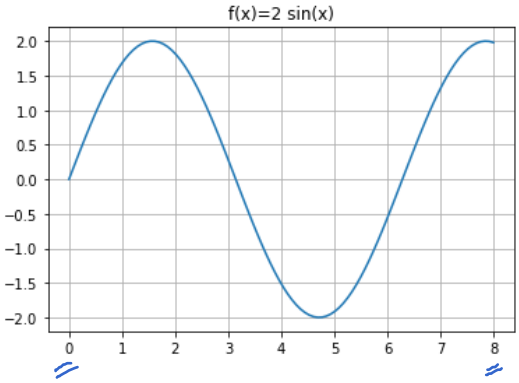

```
lt = [0, 1, 2, 5, 9, 11]
lv = [0, 2, 5, 8, 14, 20]
plt.plot(lt, lv)
plt.plot(lt, lv, '*')
plt.xlabel('t(s)')
plt.ylabel('v(m/s)')
plt.grid()
plt.show()
```

Exemple 2:



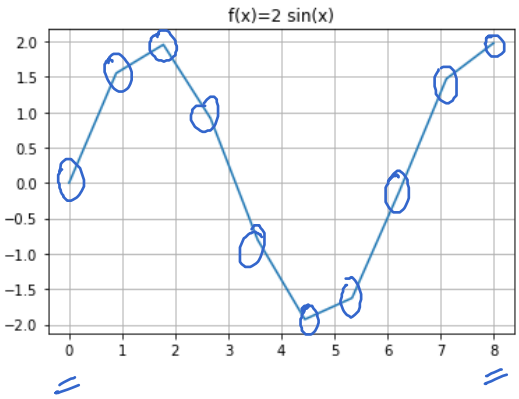

```
lt = [2, 3, 4, 5, 6, 7]
lx = [0, 4, 5, 7, 6, 3]
plt.plot(lt, lx, 'o')
plt.xlabel('t(s)')
plt.ylabel('x(m)')
plt.grid()
plt.show()
```

Exemple 3:




```
x = linspace(0, 8, 1000)
y = 2 * np.sin(x)
plt.plot(x, y)
plt.grid()
plt.show()
```

Exemple 4:




```
x = np.linspace(0, 8, 10)
y = 2 * np.sin(x)
plt.plot(x, y)
plt.grid()
plt.show()
```

Savoir faire 4: méthode d'Euler pour une résolution numérique d'équations différentielles

La méthode d'Euler repose sur le DL à l'ordre 1 en dt: $f(t+dt) = f(t) + f'(t)dt = f(t) + \frac{df}{dt}dt$ où dt est le pas de temps (petit).

Dans le cas où $f(t)$ est la fonction vitesse: $v(t+dt) =$ $v(t) + \frac{dv}{dt}(t)\,dt = v(t) + a(t)\,dt$

d'où $v(t_{i+1}) =$ $v(t_i + dt) = v(t_i) + a(t_i)\,dt$

Dans le cas où $f(t)$ est la fonction position: $x(t+dt) =$ $x(t) + \frac{dx}{dt}(t)\,dt = x(t) + v(t)\,dt$

d'où $x(t_{i+1}) =$ $x(t_i + dt) = x(t_i) + v(t_i)\,dt$

Exemple 1:

```
 9 import matplotlib.pyplot as plt
10 import numpy as np
11
12 lt,lv=[0],[0]
13 g,k=9.8,2
14 tau,N=0.01,100
15 for i in range(N):
16     lt.append(lt[i]+tau)
17     a=-k*lv[i]**2+g
18     lv.append(lv[i]+tau*a)
19
20 plt.plot(lt,lv)
21 plt.grid()
22 plt.xlabel('t(s)')
23 plt.ylabel('v(m/s)')
24 plt.show()
```

on crée deux listes avec les CI. $t=0$ et $v(t=0)=0$

$i = 0, 1, \dots N-1 = 99$: on ajoute $N = 100$ termes

$lt = [0, tau, 2tau, \dots N \times tau] = [0, 0.01, 0.02, \dots, 0.01 \times 100 = 1]$

$lv[i+1] = lv[i] + a \times tau$ avec $a = -kv^2 + g$

$v(t=0) = 0$ et $a(t=0) = g$

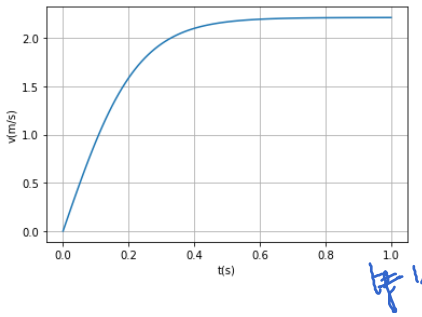
le mvt est accéléré $v \uparrow$ et a un moment $-kv^2 + g = 0$,

le mvt devient uniforme : $v = cte = \sqrt{\frac{g}{k}} = \sqrt{\frac{9,8}{2}} \approx 2,2\ m\,s^{-1}$

L'exécution du code donne les courbes suivantes en modifiant la ligne 14:

tau,N=0.01,100

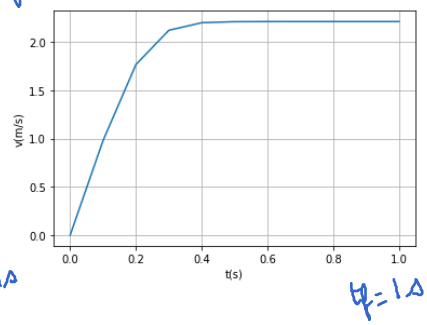
$t_f = N \times tau = 1\ s$



$t_f = 1\ s$

tau,N=0.1,10

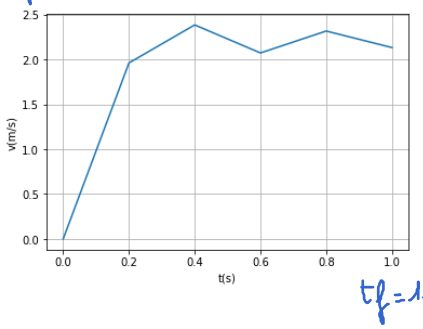
$t_f = N\,tau = 1\ s$



$t_f = 1\ s$

tau,N=0.2,5

$t_f = N \times tau = 1\ s$



$t_f = 1\ s$

Commenter le code et les courbes obtenues.

Les simulations se font toutes entre $t=0\ s$ et $t_f = N\,tau = 1\ s$.

On applique la méthode d'Euler avec $N+1$ points

Pour $N = 100$ la courbe est propre et continue

$N = 5$ et 10 : la courbe n'est pas continue, la méthode d'Euler repose sur du DL qui ne sont justifiés que pour tau petit, dans les cas 2 et 3, tau est mal choisi

Exemple 2:

```
27 N,tau=500,0.02
28 lt=np.zeros(N)
29 lv=np.zeros(N)
30 lx=np.zeros(N)
31 lt[0],lv[0],lx[0]=0,10,0  # CI : t=0  v(t=0)=10 m.s^-1  x(t=0)=0
32 for i in range(len(lt)-1): # i=0,1,2,... N-2 ou i+1=1,2,... N-1
33     lt[i+1]=lt[i]+tau
34     a=-lv[i]**3
35     lv[i+1]=lv[i]+a*tau
36     lx[i+1]=lx[i]+lv[i]*tau
```

on crée 3 vecteurs avec N termes nuls pour le moment

lt = array([0, tau, 2 tau, ... (N-1) tau])

$t_f = 499 \times 0{,}02 \simeq 10\ s$

car $v(t_{i+1}) = v(t_i + tau)$
$= v(t_i) + a * tau$

$x(t_{i+1}) = x(t_i + tau)$
$= x(t_i) + v(t_i) * tau$

Compléter le code et l'expliquer.