```
import matplotlib.pyplot as plt
import numpy as np
from scipy.optimize import curve_fit


# Import des donnees du fichier txt
t, T = np.loadtxt('newtonlaw2.txt', unpack = True , usecols=(0, 1), delimiter = ';', skiprows = 0)


# Creation du graphique avec les points de mesure
plt.title('Temperature en fonction du temps')
plt.xlabel('temps t (s)')
plt.ylabel('température T(°C) ')
plt.scatter(t, T, color = 'blue', label = 'Points expérimentaux', marker = '+')
plt.legend()

#Définition du modèle que l'on souhaite vérifier
def FonctionModele(t,Ta,Ti,tau):
        return Ta + (Ti-Ta)*np.exp(-t/tau)

#Calcul du modèle
parametre,covariance= curve_fit(FonctionModele,t,T,p0=(30,60,2000))
Ta=parametre[0]
Ti=parametre[1]
tau=parametre[2]

#Tracé du modèle
tth=np.linspace(0,7000,5000)
Tth=Ta + (Ti - Ta)*np.exp(-tth/tau)
plt.plot(tth, Tth, color = 'green', label = 'Modele', marker = '')
plt.legend()

#affichage de l'équation
plt.text(1500,50,'T(t) = {: .1f} + ({: .1f} - {: .1f}) . exp(-t/{: .1f})'.format(Ta,Ti,Ta,tau))

plt.show()
```

Import des bibliothèques python à utiliser

Import des données dans t et T → t,T=np.loadtxt(
''newtonlaw2.txt', → fichier de données dans le même répertoire
unpack = True , → permet d'importer directement sous la forme t,T=
Usecols=(0, 1), → on utilise les colonnes 0 et 1 du fichier
Delimiter = ';', → on spécifie le délimiteur utilisé dans ce fichier
Skiprows = 0 → nombre de lignes à sauter (en cas d'en-tête)

Tracé des points expérimentaux M(t,T) . **plt.scatter** permet d'avoir des points non reliés à la différence de **plt.plot**

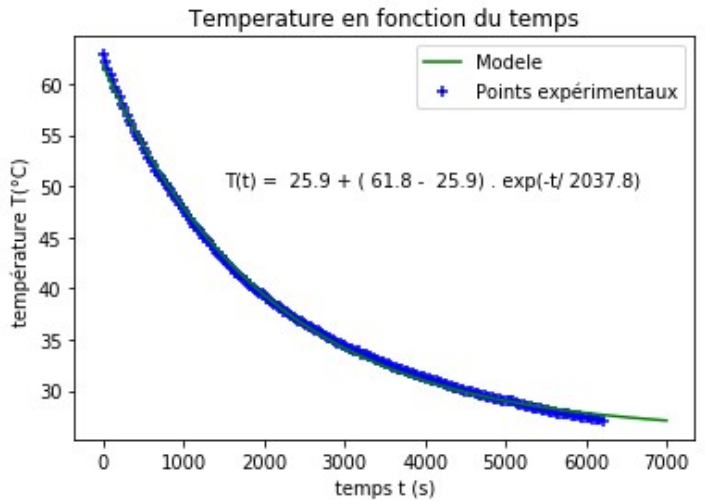
On donne la loi à vérifier sous la forme
loi(variable, paramètre1, paramètre2, etc …)

curve_fit calcule les paramètres du modèle et la covariance. Ici on rentre un jeu initial de paramètres p0=(….) mais pour des fonctions simples ce n'est pas obligé (voir exemple2)

Tracé des points théoriques M(tth,Tth) .
On commence par prendre 5000 valeurs de tth entre 0 et 7000 grâce à np.linspace et ensuite on calcule Tth pour ce points avec le modèle calculé.

{: .1f} affiche les variables dans l'ordre définit dans format(…) avec 1 décimale (.2f pour 2, etc ...)

```
import matplotlib.pyplot as plt
import numpy as np
from scipy.optimize import curve_fit

# Lecture des donnees du fichier txt
t, x, y = np.loadtxt('tirbasket.txt', unpack = True , usecols=(0, 1, 2), delimiter = '\t', skiprows = 2)

# Creation du graphique avec les points de mesure
plt.title('Trajectoire du ballon')
plt.xlabel('Abscisse x (m)')
plt.ylabel('Altitude y (m)')
plt.scatter(x, y, color = 'blue', label = 'Points experimentaux, marker = '+')
plt.legend()

#Définition du modèle que l'on souhaite vérifier
def FonctionModele(x,a,b,c):
    return a*x**2 + b*x + c

#Calcul du modèle
parametre,covariance= curve_fit(FonctionModele,x,y)
a=parametre[0]
b=parametre[1]
c=parametre[2]


#Tracé du modèle
xth=np.linspace(1,3.5,200)
yth=a*xth**2 + b*xth+c
plt.plot(xth, yth, color = 'green', label = 'Modele', marker = '')
plt.legend()

#affichage de l'équation
plt.text(2,3,'y= {:.3f} $x^2$+{:.3f} x + {:.3f}'.format(a,b,c))

plt.show()
```



Dans cet exemple on affiche les points obtenus par pointage vidéo d'un tir au basket et on modélise par une fonction de degré 2