Correction Exercice 5.b : filtrage vitesse tête imprimante

1. Correction code python

```
import numpy as np; import matplotlib.pyplot as plt ; from scipy import
signal

# importation des mesures
mesures = np.loadtxt('BO200imprimante.csv', delimiter=';', skiprows=1,
dtype=str)
mesures = np.char.replace(mesures, ',', '.');
mesures = mesures.astype(float)
temps = mesures[:,0]; vitesse = mesures[:,2]

# filtre du premier ordre : Te période d'échantillonnage en (s) et fc de
coupure en Hz
Te = 0.002 ; fc = 40 ; tau = 1/(2*np.pi*fc) ; f_f1=[vitesse[0]] #
Initialisation
for i in range(1,len(vitesse)):
     f_f1.append(f_f1[i-1]+Te/tau*(vitesse[i-1]-f_f1[i-1]))

# filtre de butterworth en utilisant le module scipy
Te = 0.002 ; fc = 40 ; fnyq = 1/2/Te ; ordre = 1 # inititalisation et choix
de l'ordre
b, a = signal.butter(ordre, fc/fnyq, 'low', analog=False) # calcul du filtre
f_fb1 = signal.filtfilt(b, a, vitesse) # Application du filtre

# Affichage des 3 courbes
plt.plot(temps, vitesse, label=" vitesse mesurée (mm/s) ")
plt.plot(temps, f_f1, label=" filtrage 1er ordre ")
plt.plot(temps, f_fb1, label=" filtrage Butter 1 ")
plt.xlabel("Temps (s)")
plt.title("Vitesse tête Imprimante") ; plt.legend() ; plt.grid(True) ;
plt.show()
```

Question 3 : Le résultat avec Butterworth est meilleur qu'avec un filtre du 1er ordre simple. Si on baisse la période d'échantillonnage, les deux filtres sont proches (après essai, le fait d'augmenter l'ordre du filtre de Butterworth ne change pas sensiblement le résultat avec ce jeu de mesure).