TP - Filtrage numérique

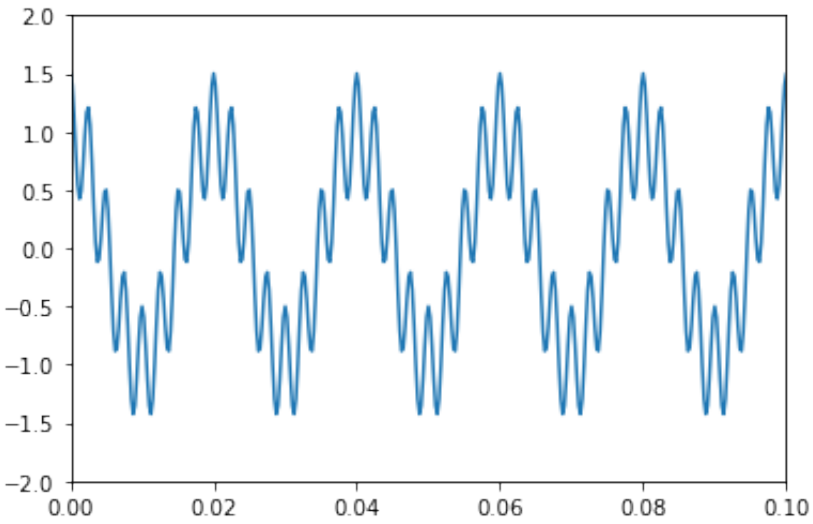
On teste les parties compétences numériques pour la FFT et le filtrage numérique.

Dans le programme on doit utiliser la fonction **rfft** de la bibliothèque **numpy.fft**.

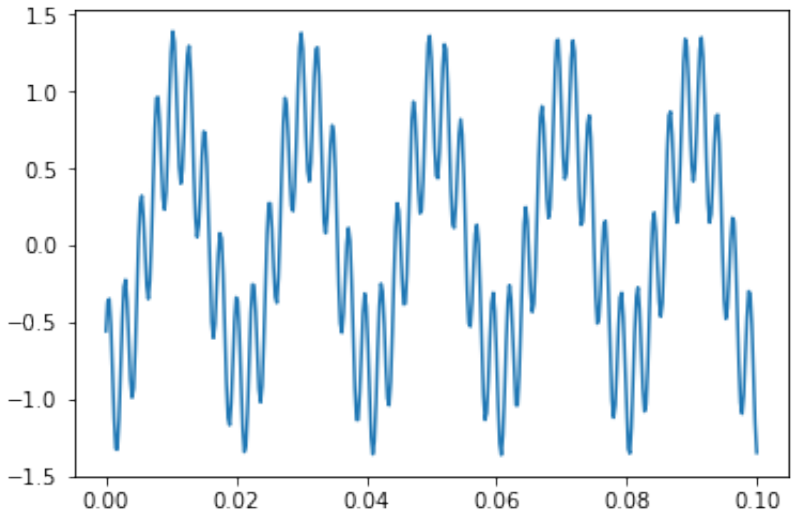
Le signal à étudier peut être obtenu en sommant des fonctions (avec numpy) ou en récupérant des données expérimentales (fichier csv).

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt

fe=4e3
f1=50
f2=400

Tacq=1


Te=1/(fe)



# Création du signal à filtrer
t=np.arange(0,Tacq,Te)
N=len(t)
signal=np.cos(2*np.pi*f1*t)+0.5*np.cos(2*np.pi *f2*t)

print(N)
plt.axis([0,0.1,-2,2])
plt.plot(t,signal)
plt.show()
```

```
4000
```

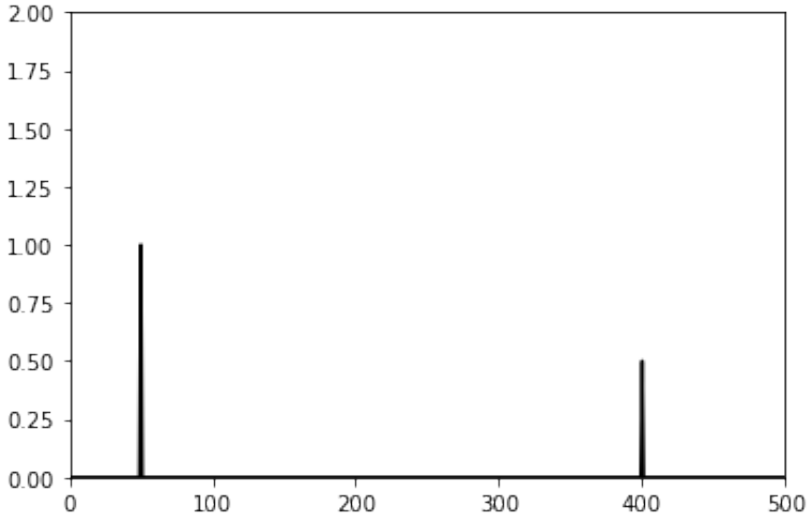


In [2]:
```
f = open("signal2.csv") # on oouvre le fichier avec les résultats
sep=";" # données séparées par un ; dans tableau1
data=f.readlines() #on lit toutes les lignes
f.close() #on referme le fichier


t1=[]
s1=[]

for ligne in data:
    ligne=ligne.strip().split(sep) #on sépare les différents élémen
ts
    ligne=list(map(float,ligne)) #on convertit chaque élément en fl
ottant
    t1.append(ligne[0]) #on rentre les valeurs dans les lites adapt
ées
    s1.append(ligne[1])
plt.plot(t1,s1)
plt.show()

N1=len(t1)
```



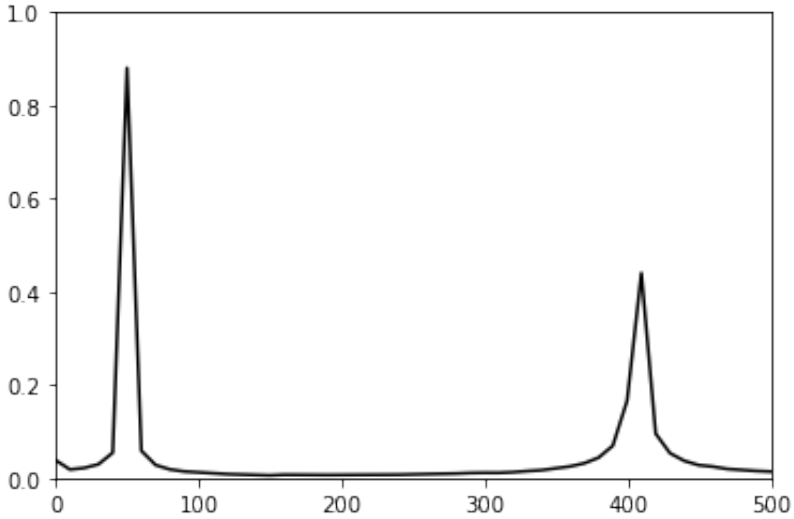
On peut alors calculer le spectre du signal.

In [3]:
```
fourier = np.fft.rfft(signal)

freq = np.fft.rfftfreq(N, d=Te)

plt.plot(freq,np.abs(fourier[0:len(freq)])*2/N,'k-')
plt.axis([0,500,0,2])
plt.show()
```



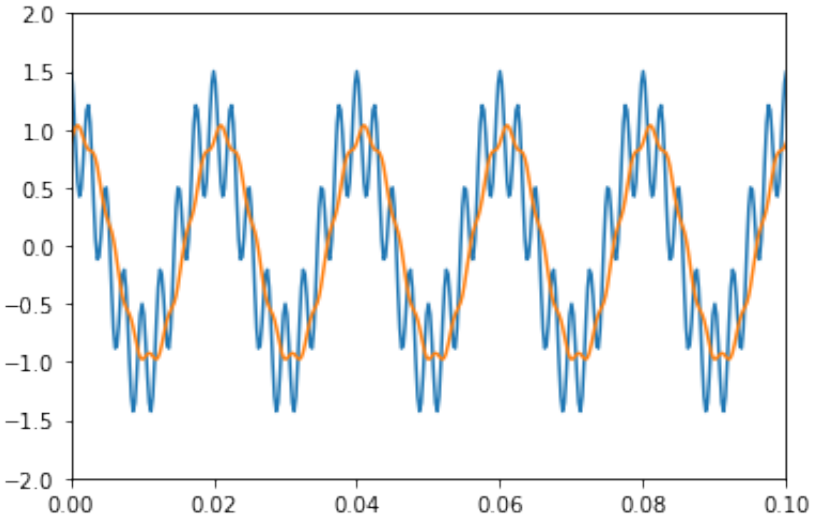
Même chose pour les données expérimentales :

In [4]:
```
fourier_s1 = np.fft.rfft(s1)

freq_s1 = np.fft.rfftfreq(N1, t1[1]-t1[0])

plt.plot(freq_s1,np.abs(fourier_s1[0:len(freq_s1)])*2/N1,'k-')
plt.axis([0,500,0,1])
plt.show()
```

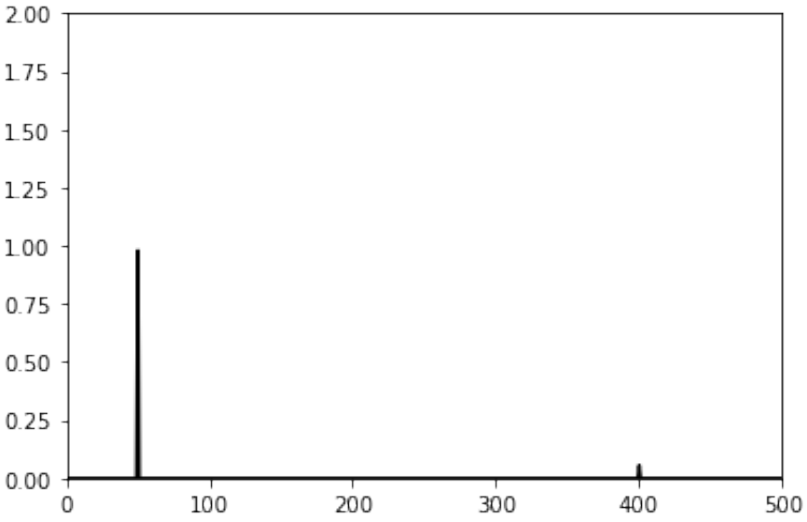


On peut ensuite réaliser les différents filtrage et observer leur action sur le spectre du signal.

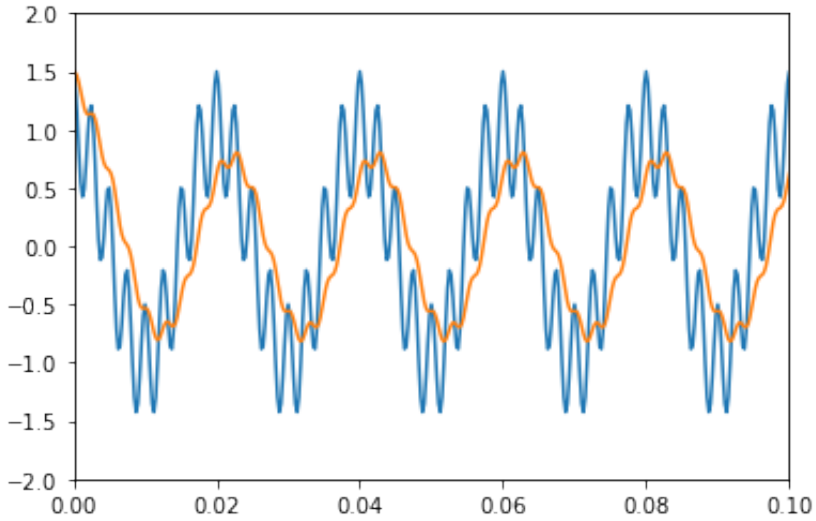
In [5]:
```
#Filtre à moyenne glissante
filtre1=np.zeros(N)
nfiltre=9
for i in range(N):
    for k in range(nfiltre):
        filtre1[i]+=signal[i-k]/nfiltre

plt.axis([0,0.1,-2,2])
plt.plot(t,signal)
plt.plot(t,filtre1)
plt.show()
```



In [6]:
```
fourierFiltre1=np.fft.rfft(filtre1)
plt.plot(freq,np.abs(fourierFiltre1[0:len(freq)])*2/N,'k-')
plt.axis([0,500,0,2])
plt.show()
```



In [7]:
```
#Filtre passe-bas : méthode d'Euler
fc=60 #fréquence de coupure du filtre
wc=2*np.pi*fc
filtre2=np.zeros(N)
filtre2[0]=signal[0]
for i in range (1,N):
    filtre2[i]=(filtre2[i-1]+Te*wc*signal[i])/(1+wc*Te)

plt.axis([0,0.1,-2,2])
plt.plot(t,signal)
plt.plot(t,filtre2)
plt.show()
```



In [ ]:

In [ ]:

In [ ]: