# ESSAI05 FILTRAGE NUMERIQUE ET FFT V. HAUTIERE

L'objectif est de réaliser un filtrage numérique passe-bas (**circuit** $RC$) en mettant en entrée des signaux générés par le $GBF$ et enregistrés via **l'oscilloscope numérique**. J'ai à ma dispositions quatres signaux que j'ai fabriqué avec le gbf AFG1022 etenregistré via le port USB de l'oscilloscope : un signal sinusoïdal à $1kHz$ (essai02_sinus.csv), un signal triangle à $1kHz$ (essai02_triangle.csv), un signal carré à $1kHz$ (essai02_carre.csv) et un signal sinusoïdal bruité avec un sinus $5V, 1kHz +$ un sinus $1V, 100kHz$ (essai05_signalbruite.csv).

Sur ces signaux, on peut appliquer la méthode d'Euler explicite, la fft et les représenter.

**J'ai créé les fonctions suivantes :**

- `readCSV (file,sep,n)` fonction de lecture des fichiers csv.
    - Cette fonction a comme paramètres `file` le nom du fichier csv, `sep` le separateur employé dans le csv et `n` le numéro de colonne. Cette fonction retourne une liste `col` contenant les valeurs numériques du numéro de colonne choisie du fichier csv.
- `eulerPB (te,ue,f_coupure)` fonction qui met em oeuvre la méthode d'Euler explicite avec l'équation différentielle du passe-bas passif du premier ordre RC.
    - Cette fonction a comme paramètres `te` la liste des dates en entrée, `ue` la liste des valeurs de la tension d'entrée $u_e(t_e)$, `f_coupure` la fréquence de coupure pour le filtre passe-bas. Cette fonction retourne deux listes `ts` , `us` qui correspondent à la tension de sortie $u_s(t_s)$
- `fourier (t,u,z)` fonction qui met em oeuvre la FFT sur le signal u(t).
    - Cette fonction retourne deux listes `freqs` (la liste des fréquences en abscisse) et `FFT_u` (la liste des amplitudes associées). `z` est un paramètre de "zoom" de la FFT que j'ai ajouté. ($z \in \mathbb{N}$ et $z \geq 2$). $z = 2$ est la valeur à utiliser par défaut. Si on souhaite zoomer le spectre, on augmente la valeur de z avec parcimonie!!!
- `graphique(t,u,freqs,FFT,titre)` fonction qui trace la tension et sa FFT.

## Importations des modules

In [1]:
```
#IMPORTATION DES MODULES
import csv
import math
import matplotlib.pyplot as plt
import numpy as np
import scipy as sp
import scipy.fftpack #Pour récupérer les fréquences de la FFT
```

## `readCSV (file,sep,n)` *fonction de lecture des fichiers csv*

In [2]:
```
#`readCSV (file,sep,n)` fonction de lecture des fichiers csv.

def readCSV(file,sep,n):
    with open(file,"r") as f:
        read=csv.reader(f,delimiter=sep)
        col=[]
        for row in read :
            try:
                col.append(float(row[n].replace(",","")))
            except:
                pass
    return col
```

## `eulerPB (te,ue,f_coupure)` *fonction qui met em oeuvre la méthode d'Euler explicite avec l'équation différentielle du passe-bas passif du premier ordre RC*

In [3]:
```
#`eulerPB (te,ue,f_coupure)` fonction qui met em oeuvre la méthode d'Euler
#avec l'équation différentielle du passe-bas passif du premier ordre RC.

def eulerPB (te,ue,f_coupure):
#initialisation des variables
    N=len(te) #nombre d'iterations
    a=te[0] #borne inferieure
    b=te[N-1] #borne superieure
    h=te[1]-te[0] #pas de discretisation

    tn=te[0] #valeur initiale de l'abscisse
    un=0 #valeur initiale de l'ordonnee
    ts=[tn] #initialisation de la liste des abscisses
    us=[un] #initialisation de la liste des ordonnees

    #algorithme d'Euler
    for i in range(1,N):
        tn=tn+h
        un=un*(1-h*2*math.pi*f_coupure)+h*2*math.pi*f_coupure*ue[i]
    #enregistrement des valeurs de tn et fn dans les listes t et f
        ts.append(tn)
        us.append(un)

    return ts, us
```

## `fourier (t,u,z)` *fonction qui met en oeuvre la FFT sur le signal u(t)*

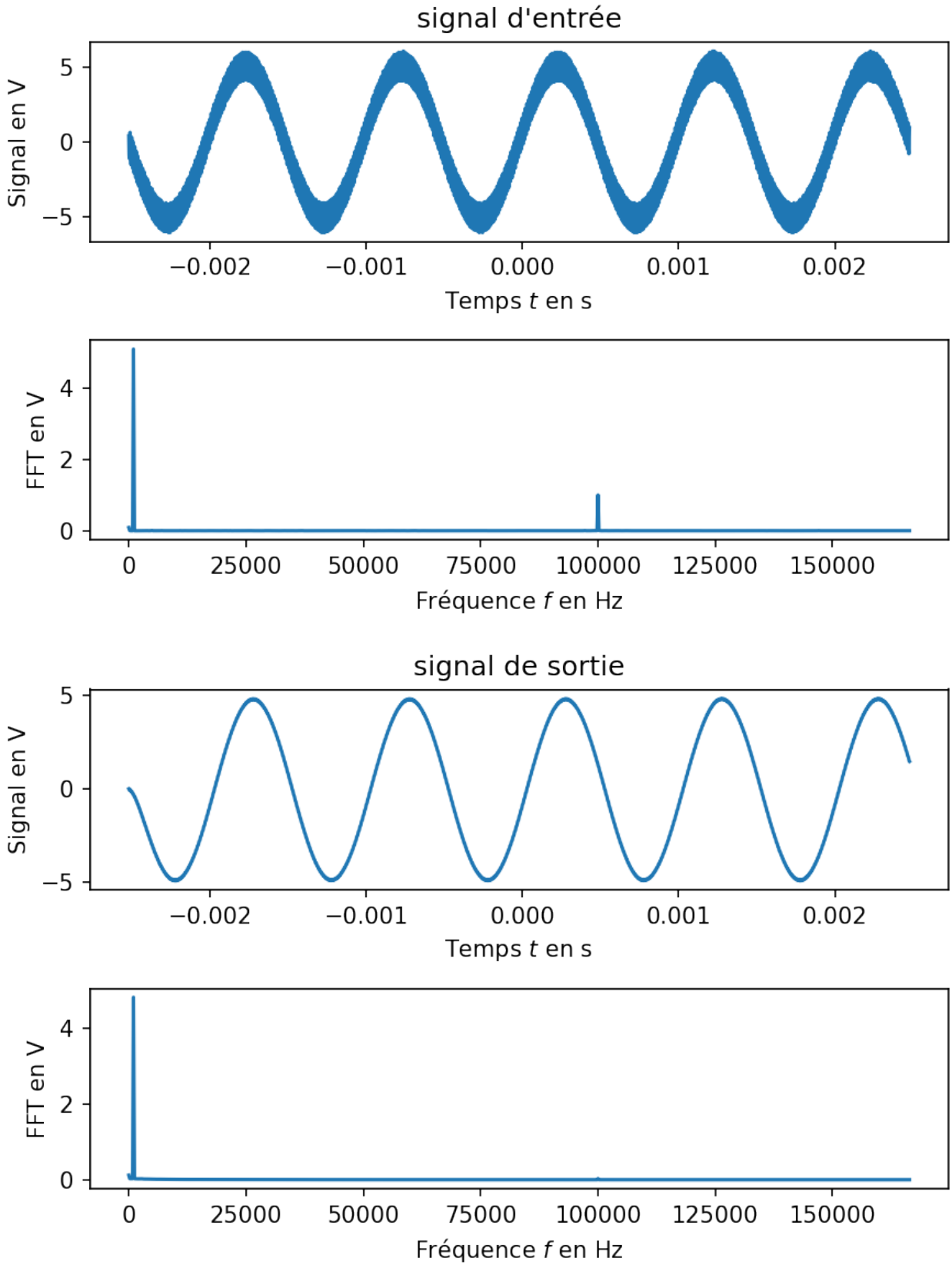
In [4]:
```
#`fourier (t,u)` fonction qui met em oeuvre la FFT sur le signal u(t).

def fourier (t,u,z):

    signal=np.array(u)
    #On fait les FFT sur le signal u en prenant le module.

    #Pour avoir les amplitudes, on normalise en divisant par la moitié du

    FFT_u=abs(sp.fft.fft(signal))/(signal.size/2)

    freqs=sp.fftpack.fftfreq(signal.size,t[1]-t[0])

    #On ne prend pas en compte les résultats pour les fréquences négatives
    FFT_u=FFT_u[0:len(FFT_u)//z]
    freqs=freqs[0:len(freqs)//z]
    return freqs,FFT_u
```

## `graphique(t,u,freqs,FFT,titre)` *fonction qui trace la tension et sa FFT*

In [5]:
```
#graphique(t,u,freqs,FFT,titre)` fonction qui trace la tension et sa FFT.
def graphique(t,u,freqs,FFT,titre):
    plt.figure(dpi=150)
    plt.subplot(211) #Fenêtre en 2 lignes, 1 colonne, première figure
    plt.title(titre)
    plt.plot(t,u) #Le signal en fonction du temps
    plt.xlabel('Temps $t$ en s')
    plt.ylabel('Signal en V')
    plt.subplot(212) #Fenêtre en 2 lignes, 1 colonne, deuxième figure
    plt.plot(freqs,FFT) #La FFT en fonction de la fréquence
    plt.xlabel('Fréquence $f$ en Hz')
    plt.ylabel('FFT en V')
    plt.tight_layout() #Pour mettre de l'espace entre les sous-figures
    #plt.savefig(fichier) Sauvegarde
    #plt.clf() #Nettoyage
    plt.show()
```

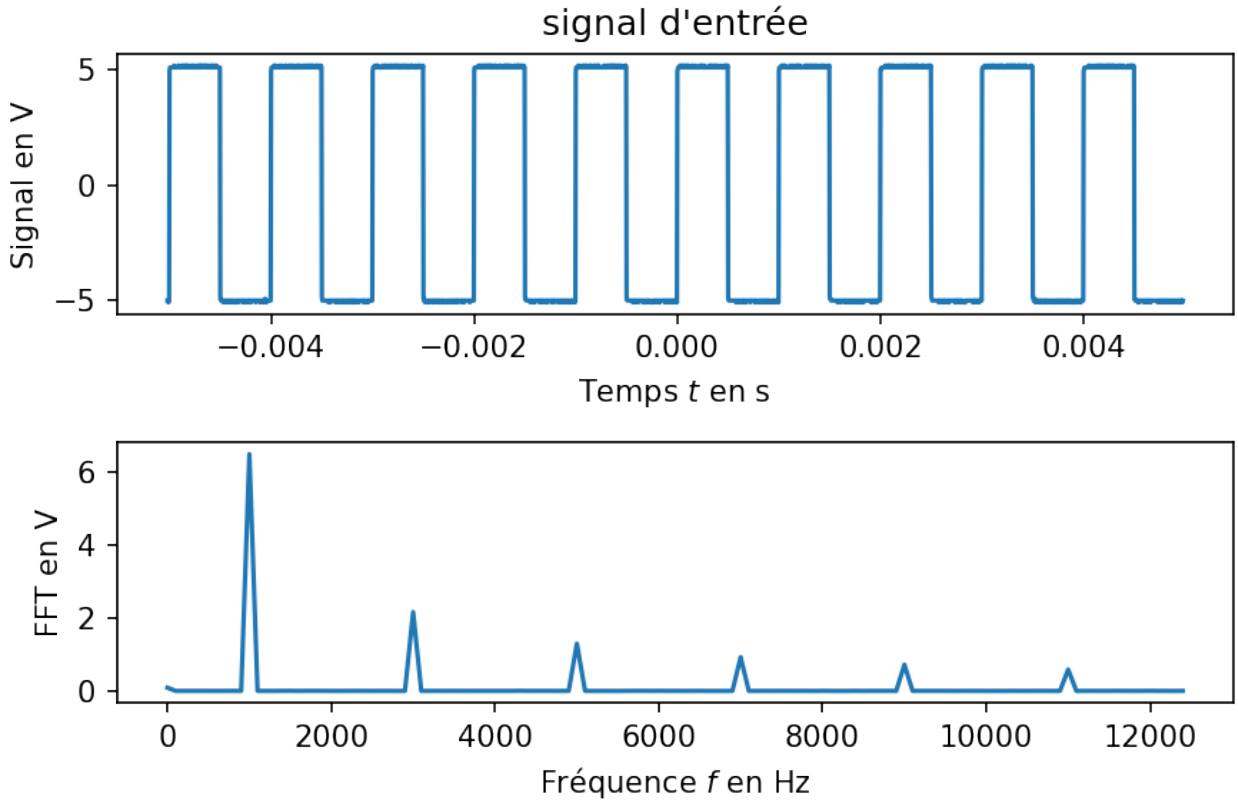
## *Exemple01 d'utilisation sur `essai05_signalbruite.CSV` avec une frequence de coupure* $f_c = 3000Hz$

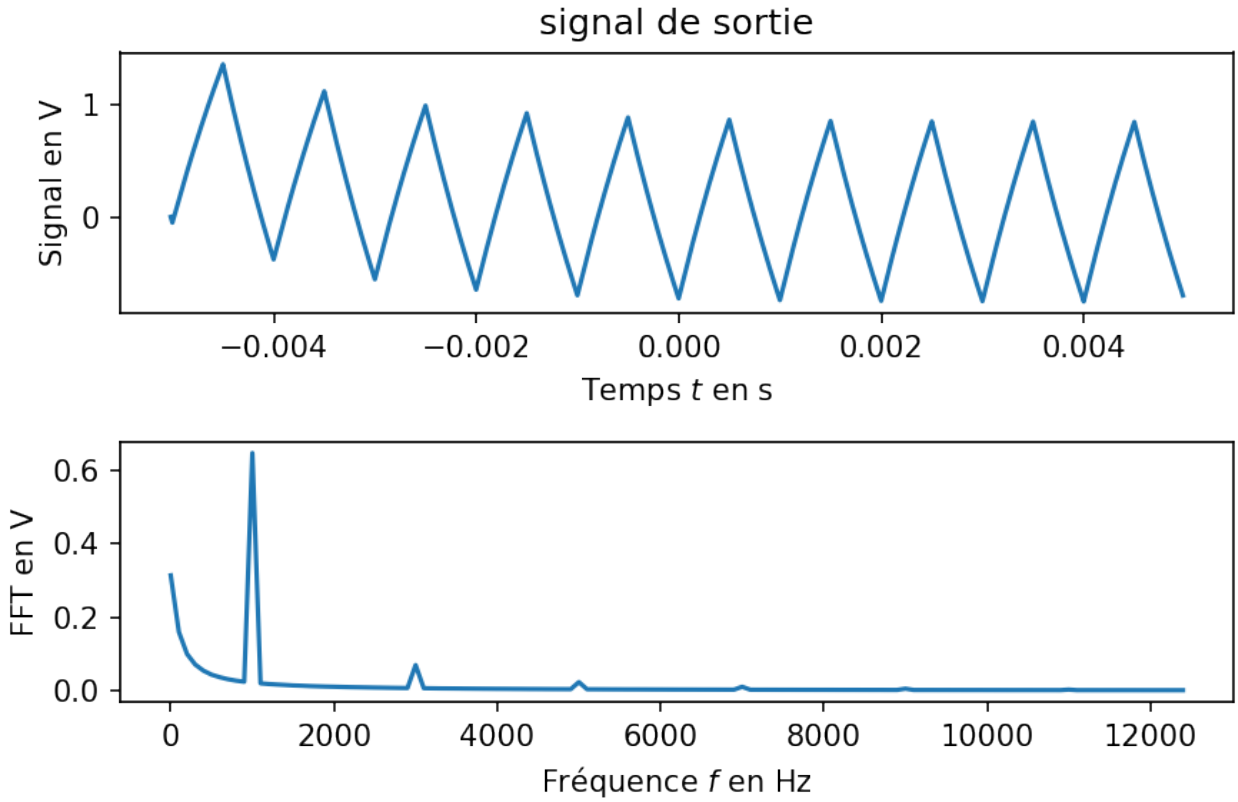
In [6]:
```
te=readCSV("essai05_signalbruite.CSV",",",3)

ue=readCSV("essai05_signalbruite.CSV",",",4)

ts,us=eulerPB(te,ue,3000)

fe,fft_ue=fourier(te,ue,3)

fs,fft_us=fourier(ts,us,3)


graphique(te,ue,fe,fft_ue,"signal d'entrée")

graphique(ts,us,fs,fft_us,"signal de sortie")
```




***Exemple02 d'utilisation sur `essai05_carre.CSV` avec une frequence de coupure $f_c = 100Hz$***

In [7]:
```
te=readCSV("essai05_carre.CSV",",",3)

ue=readCSV("essai05_carre.CSV",",",4)

ts,us=eulerPB(te,ue,100)

fe,fft_ue=fourier(te,ue,20)

fs,fft_us=fourier(ts,us,20)


graphique(te,ue,fe,fft_ue,"signal d'entrée")

graphique(ts,us,fs,fft_us,"signal de sortie")
```




signal de sortie
Signal en V
1
0
−0.004
−0.002
0.000
0.002
0.004
Temps $t$ en s


FFT en V
0.6
0.4
0.2
0.0
0
2000
4000
6000
8000
10000
12000
Fréquence $f$ en Hz