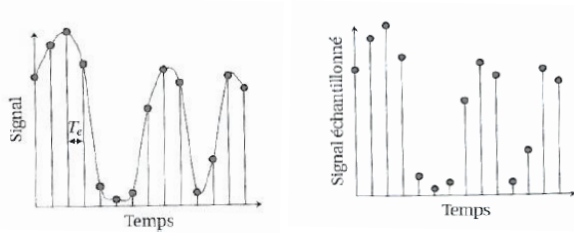


Compétences : liste, slicing (extraction de tranche), maximums et minimum d'une liste

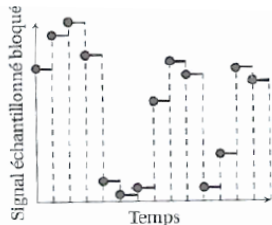
Lien capitale : 5661-3797620

Fabriquer un effet audio : lire en sens inverse une piste audio

Le but de cet exercice est d'illustrer la numérisation d'un signal en utilisant comme exemple le son. L'échantillonnage d'un signal consiste à prendre la valeur du signal à différents instants et de manière périodique (avec une période T_e et donc une fréquence d'échantillonnage $f_e = \frac{1}{T_e}$) :



La seconde opération d'une conversion consiste à bloquer la valeur de l'échantillon pendant T_e :



On obtient une série de valeurs s_i du signal échantillonné. La dernière étape de numérisation consiste à convertir ces tensions sur n bits. Par exemple la conversion d'un signal variant entre $-5V$ et $5V$ sur 16 bits consiste à convertir ces 10 V sur 65536 niveaux (de -32768 à 32767).

Nous allons utiliser la librairie `scipy.io.wavfile` qui permet de récupérer le niveau de chaque échantillon ainsi que les librairies `numpy` et `matplotlib`:

```
import scipy.io.wavfile
import matplotlib.pyplot as plt
import numpy as np
```

Pour ouvrir avec python le fichier audio « projet2i.wav » fourni, obtenir la fréquence f_e d'échantillonnage et une liste `data` de données audios échantillonnées (au format `int16`), on utilise les commandes ci-dessous :

```
"""lecture"""
fe,data =scipy.io.wavfile.read('projet2i.wav')
data=data.tolist()
```

La commande ci-dessous permet de créer un fichier audio `projet2i_reponse.wav` à partir de son fichier `data` et de sa fréquence d'échantillonnage :

```
scipy.io.wavfile.write('projet2i_reponse.wav',fe,np.array(data,dtype="int16"))
```

- 1) Ecrire une fonction `extreme(L)` qui prend en argument une liste `L` et qui renvoie la valeur minimale et son index, la valeur maximale et son index. Tester cette fonction avec la liste `data` et vérifier que ces valeurs extrêmes ne dépassent les valeurs extrêmes imposées par le codage sur 16bits signés.
- 2) Le fichier audio « projet2i.wav » contient un message qu'il convient de déchiffrer en inversant le sens de lecture. Définir une fonction `inversion` qui prend comme argument la liste `data`. Cette fonction ne renvoie rien mais génère un nouveau `projet2i_reposne.wav` qui reprend le fichier initial mais lu à l'envers.