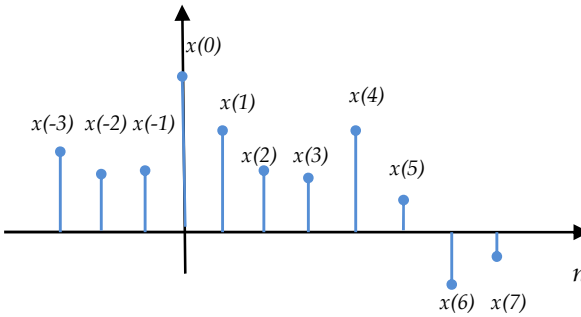
TP n°1 : Analyse temporelle des SLID (sous Python)
Convolution, Energie, Puissance et Corrélation

I. Rappels



Un signal discret s(n) est une suite de N échantillons régulièrement espacés de Te secondes: x(0),x(Te),x(2Te),……, x((N-1)Te) où Fe=1/Te est la fréquence d'échantillonnage.

Au début du programme python, importer les bibliothèques nécessaires. Pour alléger l'écriture leur donner un nom plus court:

```
import numpy as np
import matplotlib.pyplot as plt
```

NumPy est une bibliothèque destinée à manipuler des matrices ou tableaux multidimensionnels ainsi que des fonctions mathématiques opérant sur ces tableaux.
SciPy regroupe un ensemble de bibliothèques Python à usage scientifique avec un environnement très similaire à Matlab. Elle comporte des modules pour l'optimisation, l'algèbre linéaire, les statistiques, le traitement du signal ou encore le traitement d'images. **Scipy** utilise les tableaux et matrices du module **NumPy**.

II. Exemples à tester avant le TP

1. Le programme suivant permet de générer un Dirac en 0 : $\delta(n) = 1$ pour n=0 et vaut 0 ailleurs

```
import numpy as np
import matplotlib.pyplot as plt
N=32;
x = np.zeros(N); x[0]=1;
plt.figure(1)
plt.stem(x)
plt.title('Un Dirac.'); plt.grid(True); plt.xlabel('n'); plt.ylabel('Amplitude'); plt.show()
```

2. Rajouter les lignes suivante pour générer un échelon U(n)=1 pour n≥0 et 0 pour n<0

```
x=np.zeros(N); y = np.ones(N); z=np.concatenate((x,y));
plt.figure(2); plt.stem(z)
plt.title('Un Echelon'); plt.grid(True); plt.xlabel('n'); plt.ylabel('Amplitude'); plt.show()
```

3. Générer une sinusoïde de fréquence f_0=1000 et une fréquence d'échantillonnage : Fe = 1200 (Te=1/Fe)

```
import numpy as np
import matplotlib.pyplot as plt
N = 128; f0=1000; fe=12000.; Te=1/fe
t = np.linspace(0.0, (N-1)*Te, N) ; x = np.cos(2.0*np.pi*f0*t)
plt.figure(1)
plt.subplot(211); plt.plot(t,x); plt.grid(True); plt.xlabel('Seconde(s)');
plt.ylabel('Amplitude')
plt.subplot(212); plt.stem(t, x); plt.grid(True); plt.xlabel('Seconde(s)');
plt.ylabel('Amplitude')
plt.show()
```

III. Programmes à réaliser

1. Le programme suivant permet de créer une porte de largeur 1s, centrée en 2,5 s, d'amplitude 4, échantillonnée avec Te=0.1s avec N=50 et de calculer son auto-corrélation et son énergie.

```
import numpy as np
import matplotlib.pyplot as plt
N = 50; A=4; Te=0.1; t = np.linspace(0, N*Te, N)
x = A*np.concatenate((np.zeros(20),np.ones(10),np.zeros(20)))
plt.figure(1)
plt.subplot(221); plt.plot(x); plt.grid(True); plt.xlabel('S'); plt.ylabel('Amp')
plt.subplot(222); plt.plot(t, x); plt.grid(True); plt.xlabel('S'); plt.ylabel('Amp')
plt.subplot(223); plt.stem(t, x); plt.grid(True); plt.xlabel('S'); plt.ylabel('Amp')
plt.show()
tt=np.linspace((1-N)*Te, (N-1)*Te, 2*N-1); Rx=np.correlate(x,x,mode='full');
plt.subplot(224);  plt.title('Autocorr');plt.plot(tt,  Rx);  plt.grid(True);  plt.xlabel('S');
plt.ylabel('Amp')
Energie=sum(abs(x)**2)
```

- Quelle est la différence entre plt.plot(x) et plt.plot(t,x)? et la différence entre plt.plot et plt.stem?
- Utiliser l'explorateur de variables pour visualiser la taille et le contenu des vecteurs t, tt, x et Rx.
- Calculer l'auto-corrélation théorique et comparer avec Rx.
- Retrouve-t-on les propriétés de l'auto-corrélation?
- Calculer l'énergie théorique et comparer avec Energie.

2. Commenter le programme suivant :

```
import numpy as np
import matplotlib.pyplot as plt
N = 500; sigma = 0.5 ; mu = 0;
x = np.zeros(N); x[0:9]=1;   y = np.roll(x,50)+ sigma * np.random.randn(N) + mu
Ryx = np.correlate(y,x,mode='full');Ryx=Ryx[N-1:2*N-1]
plt.figure(1); plt.subplot(131); plt.plot(x); plt.title('signal émis')
plt.subplot(132); plt.plot(y); plt.title('signal reçu')
plt.subplot(133); plt.plot(Ryx); plt.title('Intercorrélation entre signal émis et signal reçu');
plt.show()
```

- Induire un autre retard et observer.
- Changer la puissance du bruit et commenter.
- Donner un exemple d'application de ce programme.

3. Soit le programme ci-dessous

```
import numpy as np
import matplotlib.pyplot as plt
N = 128; f0=500; fe=10000.; Te=1/fe; T=3; sigma = 0.3 ;
t = np.linspace(0.0, (N-1)*Te, N); x = np.cos(2.0*np.pi*f0*t)+ sigma * np.random.randn(N) ;
t_h = np.linspace(0.0, (T-1)*Te, T); h = 1.0/T*np.ones(T)
t_y = np.linspace(0.0, (N+T-2)*Te, N+T-1); y = np.convolve (x, h, mode='full')
plt.figure(1);
plt.subplot(311); plt.plot(t,x); plt.grid(True); plt.title('Signal d entrée x(n)')
plt.subplot(312); plt.stem(t_h, h); plt.grid(True); plt.title('Le filtre h(n)')
plt.subplot(313); plt.plot(t_y,y); plt.grid(True); plt.title('Signal de sortie y(n)')
plt.show()
```

- Commenter le programme. Changer la valeur de T et observer.
- Comment nomme-t-on le signal h? Quel est son rôle?

4.Générer et visualiser le signal *x* composé de 64 échantillons d'une sinusoïde de fréquence f_0 =0.1 avec fe=20.f_0

- Calculer et afficher son auto-corrélation et comparer avec l'auto-corrélation théorique.
- Retrouver les caractéristiques du signal (puissance et fréquence) à partir de l'auto-corrélation.
- Générer le signal z=x+b ou b est un bruit avec (m=0 , s=0.5) . Calculer l'auto-corrélation du bruit et commenter. Calculer et visualiser l'auto-corrélation de z pour s=0.5, s=1 puis s=2 en commentant.
- Retrouver dans chacun des cas précédents la fréquence du signal à partir de l'auto-corrélation de *z*.