TP n° 5 : Notions d'estimation et Filtrage de Wiener (sous Python)

Buts : -Aborder les notions d'estimation ainsi que les propriétés des estimateurs (biais, variance) à travers un exemple

- étudier les estimateurs de la moyenne, de la variance, de la corrélation et de la DSP.
- Application du filtrage de Wiener

Exercice 1: On reprend les 3 estimateurs de la valeur d'une résistance R donnés comme suit :

$$\hat{R}_1 = \frac{\sum_{k=1}^{N} U(k)I(k)}{\sum_{k=1}^{N} I^2(k)} \qquad \hat{R}_2 = \frac{1}{N}\sum_{k=1}^{N} \frac{U(k)}{I(k)} \qquad \hat{R}_3 = \frac{\sum_{k=1}^{N} U(k)}{\sum_{k=1}^{N} I(k)}$$

Le programme suivant permet de simuler des courants et tensions bruitées et d'étudier les 3 estimateurs.

```
import numpy as np; import matplotlib.pyplot as plt;
N=1000;R1=[];R2=[];R3=[];R=1;
for k in range(2,N):
    U=1+0.22*np.random.randn(k);I=1+0.22*np.random.randn(k);
    R1=np.append(R1,sum(U*I)/sum(I**2));
    R2=np.append(R2,sum(U/I)/k);
    R3=np.append(R3,sum(U)/sum(I));
plt.figure(1);
plt.subplot(311); plt.plot(R1); plt.grid(True); plt.title('R1');
plt.subplot(312); plt.plot(R2); plt.grid(True);plt.title('R2');
plt.subplot(313); plt.plot(R3); plt.grid(True);plt.title('R3');

bR1=[];bR2=[];bR3=[];VarR1=[];VarR2=[];VarR3=[];
for k in range(2,len(R1)):
    bR1=np.append(bR1, np.mean(R1[:k])-R);
    bR2=np.append(bR2, np.mean(R2[:k])-R);
    bR3=np.append(bR3, np.mean(R3[:k])-R);
    VarR1=np.append(VarR1, np.var(R1[:k]));
    VarR2=np.append(VarR2, np.var(R2[:k]));
    VarR3=np.append(VarR3, np.var(R3[:k]));
plt.figure(2);
plt.subplot(321); plt.plot(bR1); plt.grid(True); plt.title('Biais R1');
plt.subplot(323); plt.plot(bR2); plt.grid(True);plt.title('Biais R2');
plt.subplot(325); plt.plot(bR3); plt.grid(True);plt.title('Biais R3');
plt.subplot(322); plt.plot(VarR1); plt.grid(True); plt.title('Variance de R1');
plt.subplot(324); plt.plot(VarR2); plt.grid(True);plt.title('Variance de R2');
plt.subplot(326); plt.plot(VarR3); plt.grid(True);plt.title('Variance de R3');
```

1. Expliquer toutes les boucles.
2. Etudier le biais et la variance de chaque estimateur. Conclure.
3. Lesquels sont consistants? Lequel est le plus efficace?
4. Utiliser le programme ci-dessus pour construire les estimateurs de la moyenne et de la variance

$$\hat{m} = \frac{1}{N}\sum_{k=0}^{N-1} x_k \qquad \hat{\sigma}_1^{\,2} = \frac{1}{N}\sum_{k=0}^{N-1} (x_k - m)^2 \qquad \hat{\sigma}_2^{\,2} = \frac{1}{N-1}\sum_{k=0}^{N-1} (x_k - \hat{m})^2$$

5. Prendre m=1 et σ^2=2. Etudier le biais, la variance et la consistance de l'estimateur de la moyenne
6. Quelle est la différence entre les deux estimateurs de la variance ?
7. Comparer les 2 estimateurs de la variance sur les 100ères valeurs. Quel est le biais de chacun d'eux ?
8. Que remarque-t-on lorsque N augmente ? Les 3 estimateurs sont-ils efficaces?

Exercice 2 : Les deux estimateurs classiques de la fonction d'autocorrélation sont donnés par :

$$\hat{R}_{kk}(k)=\hat{R}_{kk}(-k)=\frac{1}{N}\sum_{i=1}^{N-k}x(n)x(n+k) \qquad et \qquad \hat{R}_{kk}(k)=\hat{R}_{kk}(-k)=\frac{1}{N-k}\sum_{i=1}^{N-k}x(n)x(n+k)$$

```
import numpy as np; import matplotlib.pyplot as plt;
N=500; f0=0.02; Te=1;
t = np.linspace(0, N-1, N)*Te; s=np.cos(2*np.pi*f0*t)
var=4; bb = (var**0.5)*np.random.randn(N);
Cor_S= np.correlate(s,s,mode='full'); Cor_bb=np.correlate(bb,bb,mode='full');
Cor_S_biase=Cor_S/N; Cor_bb_biase=Cor_bb/N;
A=np.linspace(1,N-1,N-1); B=A[::-1]; A=np.append(A,N);A=np.append(A,B);
Cor_S_non_biase=Cor_S/ A;
Cor_bb_non_biase=Cor_bb/ A;
plt.figure(1);
plt.subplot(221); plt.plot(Cor_S_biase); plt.grid(True); plt.title('Cor du cos Biaisé');
plt.subplot(223); plt.plot(Cor_bb_biase); plt.grid(True);plt.title('Cor du bb Biaisé');
plt.subplot(222); plt.plot(Cor_S_non_biase); plt.grid(True);plt.title('Cor du cos Non Biaisé');
plt.subplot(224); plt.plot(Cor_bb_non_biase); plt.grid(True); plt.title('Cor du bb Non Biaisé');
```

1. Calculer le biais des 2 estimateurs.
2. Des 2 estimations classiques de l'autocorrélation pour le bruit laquelle paraît la plus satisfaisante?
3. Même question pour l'autocorrélation de la sinusoïde
4. Observer la DSP dans chaque et commenter en justifiant les différences (amplitude, lobes secondaires, etc.)

Exercice 3 : Il existe différentes méthodes d'estimation de la densité spectrale de puissance. Les méthodes les plus couramment utilisées sont :

- Périodogramme simple**:** $\hat{S}_{xx}(f)=\frac{1}{N}\left|X_T(f)\right|^2$ Corrélogramme: TF de la corrélation $\hat{S}_{xx}(f)=TF\left[\left\{\hat{R}_{xx}(k)\right\}\right]$
- Périodogramme moyenné **et** Périodogramme moyenné avec recouvrement (overlapping)

```
import numpy as np; import matplotlib.pyplot as plt;
N=1024; fe=N; Te=1/fe; f1=100; f2=150; A1=2; A2=1.2; A3=4;
t = np.linspace(0, N-1, N)*Te;
x = A1*np.cos(2*np.pi*f1*t)+ A2*np.cos(2*np.pi*f2*t)+A3*np.random.randn(N);
"""Périodogramme"""
TFx = np.fft.fft(x);TFx = np.fft.fftshift(TFx);
Sbx1=abs(TFx)**2/N; ff1 = np.linspace(-N/2, N/2-1, N)*fe/N;
"""Corrélogramme"""
Cor_x= np.correlate(x,x,mode='full')/N; N2=len(Cor_x);
Sbx2 = np.fft.fft(Cor_x);Sbx2 = np.fft.fftshift(Sbx2); ff2 = np.linspace(-N2/2, N2/2-1,
N2)*fe/N;
plt.figure(1);
plt.subplot(211); plt.plot(ff1,Sbx1/N); plt.grid(True); plt.title('DSP par TF au carré');
plt.subplot(212); plt.plot(ff2,abs(Sbx2)/N); plt.grid(True);plt.title('DSP par TD de la
corrélation');
"""Périodogramme moyenné: méthode de Welch"""
"""Périodogramme moyenné avec Recouvrement"""
```

1. Comparer les 4 techniques en faisant varier A1, A2 et A3 et commenter.
2. Rapprocher les fréquences et observer.
3. Commenter les quatre techniques d'estimation de la DSP en faisant varier N.
4. Quelle est l'avantage et l'inconvénient du moyennage?

Exercice 4 : L'extraction d'ECG du fœtus se fait à partir de plusieurs électrodes placées en différents endroits du ventre de la mère ; mais les enregistrements obtenus sont des mélanges de l'ECG du fœtus noté (FECG) et celui de la mère noté (MECG) auquel se rajoute le bruit thermique des électrodes et des équipements électroniques d'où l'emploi du filtre de Wiener.

```
import numpy as np; import matplotlib.pyplot as plt;
from scipy.io import loadmat;
import scipy.linalg as la; import scipy.signal as sp;
FF = loadmat('mecg1.mat')
mecg1 = [[element for element in upperElement] for upperElement in FF['mecg1']]
mecg1 = np.array(mecg1);N=len(mecg1); mecg1= np.resize(mecg1,new_shape=N)
FF = loadmat('fecg1.mat')
fecg1 = [[element for element in upperElement] for upperElement in FF['fecg1']]
fecg1 = np.array(fecg1);  fecg1= np.resize(fecg1,new_shape=N)
var = 0.01; bb=(var**0.5)*np.random.randn(N); x=fecg1+mecg1+bb;
plt.figure(1);
plt.subplot(411); plt.plot(fecg1); plt.grid(True); plt.title('ecg bébé');
plt.subplot(412); plt.plot(mecg1); plt.grid(True);plt.title('ecg mère');
plt.subplot(413); plt.plot(bb); plt.grid(True);plt.title('Bruit');
plt.subplot(414); plt.plot(x); plt.grid(True); plt.title('Observation');
"""Détermination des coefficients du Filtre de Wiener"""
P=20;
x=x-np.mean(x); mecg1=mecg1-np.mean(mecg1);
Rxx = np.correlate(x,x,mode='full')[len(x)-1:];
Rmecg = np.correlate(mecg1,mecg1,mode='full')[len(x)-1:];
Ryx = Rxx-Rmecg;
C = la.toeplitz(Rxx[0:P]);
B = Ryx[0:P];
b = la.inv(C).dot(B)
"""Filtrage"""
a = np.array([1,0]);
y = sp.lfilter(b,a,x) ;
plt.figure(2);
plt.subplot(311); plt.plot(fecg1); plt.grid(True); plt.title('ecg bébé');
plt.subplot(312); plt.plot(x); plt.grid(True);plt.title('Observation');
plt.subplot(313); plt.plot(y); plt.grid(True);plt.title('ecg bébé estimé');
```

1. Pourquoi a-t-on utilisé l'instruction `Ryx = Rxx-Rmecg;` ?
2. Quelles lignes du programme faut-il changer pour retrouver l'ECG de la mère.