TP TSA – Détection quadratique

2018-2019

Nom, Prénom :

Groupe :

Date :

**Contexte et objectifs**

On souhaite étudier expérimentalement la chaine de détection quadratique ci-dessous



Chaque élément de cette chaine est implémenté par une fonction Matlab à récupérer sur la plateforme CPe-campus sous l’archive Fichiers\_TP\_DQ.zip (cf. Section 4.2 pour les descriptions de ces fonctions).

Le signal  est un signal sinusoïdal de fréquence , d’amplitude , à phase équipartie sur  et modulé par un signal binaire  ou 1 :



Ce signal est bruité lors de la transmission par un bruit  gaussien, centré, stationnaire d’ordre 2 et de largeur de bande *B* centrée sur  (on supposera que le bruit est blanc sur le support fréquentiel du filtre F1).



L’objectif de la chaine de détection quadratique est de détecter dans l’observation reçue , la présence () ou l’absence () du signal utile .

# Manipulation

Dans l’ensemble du TP :

– tous les signaux seront échantillonnés à la fréquence  = 500 Hz.

– la bande passante du bruit  est fixée à *B* = 160 Hz.

– la fréquence du signal sinusoïdal  est fixée à  = 100 Hz.

– l'ordre du filtre passe-bande F1 (butterworth) est fixé à *ordre* = 6.

**Vous veillerez à mettre sur vos Figures des légendes et des labels explicites et informatifs.**

# Etude du bruit seul

Dans cette partie, , de sorte que le signal est toujours absent.

## Synthèse du bruit *B*(*t*)

On considère un bruit  centré, de puissance moyenne .

Avec les paramètres déterminés en préparation, reproduire dans le cadre ci-dessous, le code Matlab permettant :

* de générer une réalisation du bruit  sur une durée *T* = 100 s. Afficher dans la Figure 1, la sortie de CGN.m (**veillez à ajouter des légendes pertinentes**);
* De mesurer sur la trace obtenue les paramètres demandés en Table 1.

|  |
| --- |
| CODE 1 |

|  |
| --- |
| Figure 1 – |

|  |  |
| --- | --- |
| Moyenne de |  |
| Variance de |  |

Table 1 – Mesures de la moyenne et de la variance de

A partir de la Figure 1 et en expliquant la démarche suivie, retrouver (approximativement) la valeur de . Comparer à la valeur théorique trouvée en préparation.

## Etude du filtre passe-bande F1

On filtre le bruit  par un filtre passe-bande, de bande passante  centrée sur la fréquence

On choisit  = 16 Hz et la valeur de  identifiée dans la préparation.

Donner dans le cadre ci-dessous, le code permettant de :

* synthétiser le filtre F1 correspondant
* filtrer le bruit  par le filtre F1 (afficher avec des légendes pertinentes, la sortie de BPF.m dans la Figure 2)
* de mesurer sur la trace en sortie du filtre F1 les valeurs des paramètres demandés au Tableau 2.

|  |
| --- |
| CODE 2 |

|  |
| --- |
| Figure 2 – |

|  |  |
| --- | --- |
| Moyenne de |  |
| Variance de |  |

Table 2 – Mesures de la moyenne et de la variance de 

Estimer la valeur de . Comparer les mesures ( et ) aux valeurs théoriques obtenues en préparation. Comment peut on expliquer les éventuelles différences ?

En pratique, qu’est ce qui limite le choix d’une bande passante  trop étroite ?

## Elévation au carré et Filtrage RC passe-bas

Comme précédemment, on choisit  = 16Hz. En faisant varier le produit  dans une boucle (du type for ... end), donner dans le cadre ci-dessous, le code qui :

* génère le signal 
* calcule la valeur de la constante  correspondant au produit  choisi
* filtre le signal  par le filtre ***H*I** de constante de temps 
* mesure sur la sortie  (portez la valeurs mesurées dans la Table 3a):

- la valeur moyenne

- la variance

- le Kurtosis

|  |
| --- |
| CODE 3 |

|  |  |  |  |
| --- | --- | --- | --- |
|  | **2** | **20** | **100** |
|  |  |  |  |
| Moyenne de |  |  |  |
| Variance de |  |  |  |
| Kurtosis de |  |  |  |

Table 3a – Sortie Filtre  - Cas du bruit seul.

Le processus  est-il gaussien ? Pourquoi ?

Pour les 2 valeurs extrêmes de  proposées dans la Table 3a, afficher dans la Figure 3 ci-dessous, les sorties de RCF.m.

|  |
| --- |
| Figure 3 – |

Comparer pour chacune de ces valeurs de la constante , la valeur moyenne mesurée à la valeur théorique déterminée dans la préparation. Qu’est ce qui peut expliquer ces différences ? Comment corriger cet effet ?

Donner dans l'encadré ci-dessous, les 2 lignes de code qui implémentent cette solution.

|  |
| --- |
| CODE 4 |

Appliquer cette correction et porter les nouvelles mesures dans la Table 3b.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **2** | **20** | **100** |
|  |  |  |  |
| **Moyenne de**  (après correction) |  |  |  |
| **Variance de**  (après correction) |  |  |  |
| **Kurtosis de**  (après correction) |  |  |  |

Table 3b – Sortie Filtre  **après correction**- Cas du bruit seul.

Lorsque le Kurtosis est proche de 3, que peut on dire de la statistique du processus  ? Quel théorème ce résultat illustre-t-il ? Pour quelles(s) valeur(s) de  a-t-on une *intégration forte* ?

Comparer les variances de  pour les deux valeurs extrêmes de .

**Dans la suite du TP, il faudra systématiquement appliquer cette correction aux mesures effectuées en sortie du filtre RC.**

# Mélange Signal + Bruit

On étudie à présent le signal  en sortie du filtre *RC* passe-bas, lorsque le mélange  est reçu en entrée du détecteur.

## Sortie du filtre passe-bande F1

En utilisant les paramètres déterminés en préparation, générer une réalisation du signal  sur la même durée *T* = 100 s et la même fréquence d’échantillonnage  = 500 Hz.

Reporter le code correspondant ci-dessous.

|  |
| --- |
| CODE 5  Fs=500;  B=160;  nu\_0=100;  T=100;  Pb=5;  sigma=sqrt(Pb);  A=1;    Sp = struct('Fs',Fs,'A',A,'Fc',nu\_0,'FM',0,'Phi',2\*pi\*rand(),'T',100,'W',[]);  [S,Sp,M] = OOK(Sp); |

Vérifier que le filtre passe-bande, s’il est accordé sur la fréquence  n’altère pas le signal , en mesurant en sortie de F1 (dans le cas où  se présente seul en entrée) les paramètres demandés à la Table 4. En reprenant les mesures effectuées au paragraphe 1.2, déterminer le rapport signal sur bruit en sortie du sortie du filtre F1 ainsi que le gain  introduit par F1.

|  |  |
| --- | --- |
| Fréquence de | **100 Hz =** |
| Amplitude de | **1 =** |
| Puissance de | **0.4999** |
| Puissance de  (recopie Table 2) | **Valeur que tu dois mettre** |
| SNR | **1.0355** |
| Gain | **9.7256** |

Table 4 – Mesures des SNR et gains en sortie deF1

Comparer aux valeurs théoriques.

Les valeurs obtenues expérimentalement semblent cohérentes puisque se rapprochent des valeurs théoriques.

## Sortie du filtre RC passe-bas

Dans les mêmes conditions expérimentales (, = 16Hz, ), effectuer les différentes mesures demandées dans le tableau 5.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **2** | **20** | **100** |
| *RC* | 0.1250 | 1.25 | 6.25 |
| Moyenne de  (recopie Table 3) |  |  |  |
| Moyenne de | 0.9797 | 0.9785 | 0.9712 |
|  | 0.5012 | 0.4985 | 0.4956 |
|  | 0.4021 | 0.1225 | 0.0502 |
| SNR | 1.2630 | 4.0700 | 9.9234 |
| Gain | 1.1962 | 3.8862 | 9.4951 |
| Gain | 12.5040 | 40.6541 | 99.2420 |

Table 5 – Sortie Filtre *RC* - Cas du mélange signal + bruit.

Représentez en Figure 4, la sortie de RCF.m correspondant au cas ** =** 20.

|  |
| --- |
| Figure 4 – |

# Transmission d'un message binaire

## Modulation binaire périodique

On souhaite à présent transmettre et détecter une séquence périodique binaire.

Avec les paramètres suivant :

– Puissance du bruit *B(t)*, 

– Rapport signal sur bruit en entrée de la chaine, ηE = −10 dB

– Fréquence du signal modulant *M(t)*, *FM* = 0.05 Hz

 – Durée des signaux, *T* = 100 s

synthétiser les signaux *S(t)*, *B(t)* et *X(t)* correspondant.

En vous basant sur les résultats expérimentaux obtenus dans la partie 2, choisissez un jeu de paramètres pertinent pour calibrer les filtres F1 et HI. Reporter dans le cadre ci-dessous le code de détection du signal binaire reçu.

|  |
| --- |
| CODE 6  Fs=500;  B=160;  nu\_0=100;  T=100;  Pb=5;  sigma=sqrt(Pb);  A=1;  Fm=0.05    %Creation du signal S  figure(1);  Sp = struct('Fs',Fs,'A',A,'Fc',nu\_0,'FM',Fm,'Phi',2\*pi\*rand(),'T',100,'W',[]);  [S,Sp,M] = OOK(Sp);    %Creation du bruit B  Bp = struct('sigma',sigma,'Fs',Fs,'B',B,'T',T)  figure(2);  [B,Bp]=CGN(Bp);    %Addition des deux signaux B et S  figure(3);  [X] = AddSig(S,B);  plot(X.data)    %filtrage par F1  %parametres non modifies  figure(4);  Fp = struct('Fs',Fs,'F0',nu\_0,'Dnu',16,'order',6,'class','Butterworth') ;  [Y,Fp] = BPF(X,Fp);    %Quadrateur et Filtre H1  figure(5);  Z=SquareSig(Y);    Dnu=16;  RCDnu= 20  RC=RCDnu/Dnu;  RCFp=struct('Fs',Fs,'RC',RC);  [W,RCFp] = RCF (Z,RCFp);    %Detection binaire du signal  sigma2= mean(W.data);  DetectBin = zeros(1, length(W.data));  for i = 1:length(W.data)  if W.data(i) > sigma2  DetectBin(i) = 1;  end  end |

Visualiser dans la figure 5 (en organisant les plots avec la commande subplot(4,1,·) et en **ajoutant une légende pertinente**), les signaux :

* *S(t)*
* *X(t)*
* *W(t)*
* Le signal binaire détecté obtenu par seuillage du signal *W(t)* (commenter le choix du seuil Σ choisi)

|  |
| --- |
| Figure 5 – Allures des signaux au fur et à fur de la chaîne de traitement |

Indiquez ci-dessous les valeurs des paramètres de détection utilisés.

|  |
| --- |
| Les paramètres des détection utilisés ont été pour le produit , afin de rester dans l’hypothèse d’intégration forte.  Le seuil de détection ayant été pris est la la moyenne du signal W de sortie du filtre . |

Essentiellement, quel élément de la chaine de détection va-t-il limiter le débit de transmission?

|  |
| --- |
| L’élément de la chaîne de détection qui va limiter le débit de transmission est le filtre , qui est composé d’une résistance et d’un condensateur qui peuvent introduire des régimes transitoires trop longs (perte d’information du signal d’entrée). |

Sans chercher à les estimer ici, quel(s) critère(s) permettrai(en)t de mesurer la qualité de la détection?

|  |
| --- |
| Plus le produit sera petit plus le régime transitoire sera faible. Ils restenet néabmoins nécessaires car ils permettent le filtrage du signal. Il serait donc nécessaire d’avoir une grande résistance, avec un condensateur de faible valeur. |

1. **Décodage d'un message inconnu**

Charger le signal reçu ’SignalRecu\_j’, où j est le numéro de votre binôme.

>> load SignalRecu\_j

Le signal X(t) correspond à un message codé (code ascii 7 bits) transmis par modulation d’amplitude et dégradé par un bruit additif lié au canal de transmission. Exécuter la commande :

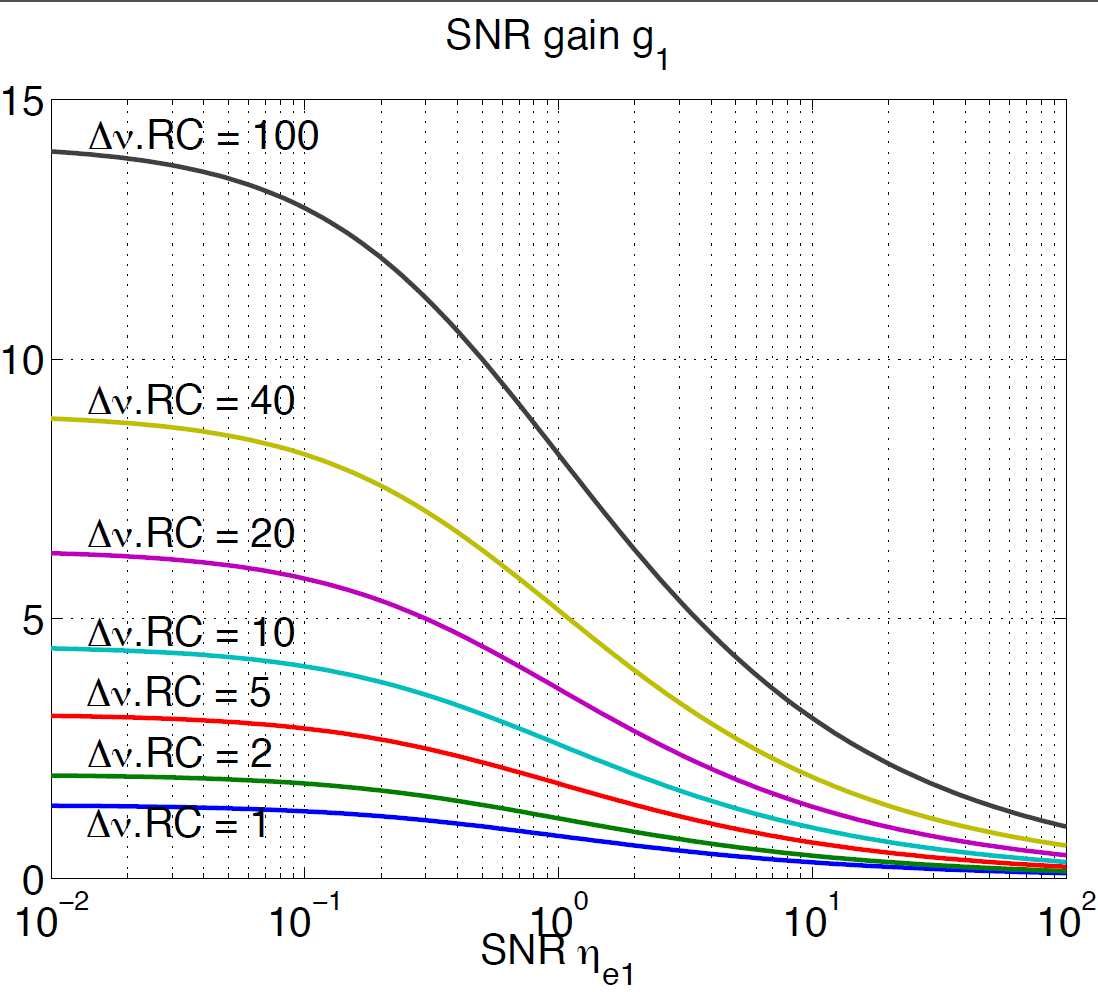
>> [TxMsg,Xp] = RxMessage\_DQ(X,Xp);

pour lancer une détection quadratique automatique sur le signal reçu *X* (la structure *Xp* contient tous les paramètres de la transmission). Ajuster en ligne, les différents paramètres de la détection jusqu’à ce que le message décodé vous semble satisfaisant. Recopier ci-dessous, le message décodé.

|  |
| --- |
| Message décodé : Attention­elle­mord.­Nous­disons­trois­fois.  Meilleur résultat obtenu pour le decodage du message avec un valeur « dethreshold value » = 0.55 |

# Annexes

# Evolution théorique du gain SNR



## Documentation routines Matlab

### OOK.m

[S,Sp,M] = OOK(Sp) Generates a ON-OFF keying modulated signal whose parameters are specified by the parameter structure Sp

S(n) = M(n).A.cos(2.pi.Fc.n/Fs + phi)

M(n) is either a binary periodic signal (0-1) oscillating at frequency FM

or a 0-1 sequence defined by W (if specified W overides FM).

OOK.m displays in the current window plot the synthesised signal.

Inputs:

Sp signal structure containing the signal parameters with following fields:

- Fs sampling frequency of the signal (in Hz)

- A amplitude of the carrier

- Fc carrier frequency (in Hz)

- FM modulating frequency (0 = no modulation) (in Hz)

- T duration of the signal (in seconds)

- W binary word to be transmitted (overides periodic modulation)

- Phi initial phase of the carrier (r.v. unif dist. over (0,2\pi))

- Class String defining the type of signal S

If varargin is left empty, each field of ’Sp’ is defined online

Outputs:

S signal structure containing the synthesised OOK signal with fields:

- data : 1-byN vector containing the data samples

- time : 1-byN vector containing the time samples

- Fs : scalar indicating the sampling frequency

Sp parameter structure (same as input)

M signal structure containing the Modulant signal (same structure as S)

Example :

Sp = struct(’Fs’,50e3,’A’,2,’Fc’,1e3,’FM’,5e1,’Phi’,0,’T’,1e-1,’W’,[])

[S,Sp,M] = OOK(Sp)

plot(S.time,S.data,M.time,M.data,’:r’)

% or

[S] = OOK()

### CGN.m

[X,Xp] = CGN(Xp) generates a filtered, centered, gaussian noise X

according to the parameters specified in the parameter structure Xp

CGN displays in the current window plot, the synthesised trace and the corresponding estimated power spectrum density.

Input

Xp parameter structure containg the follwoing fields:

- sigma : standard deviation

- Fs : scalar indicating the sampling frequency

- B : the bandwidth (in Hz, B < Fs/2)

- T : duration of the generated trace (in seconds)

If varargin is left empty, each field of ’Xp’ is defined online

Outputs

- X signal structure with the following fields:

- data : 1-byN vector containing the data samples

- time : 1-byN vector containing the time samples

- Fs : scalar indicating the sampling frequency

- Xp parameter structure (same as input)

Example :

Xp = struct(’sigma’,1,’Fs’,1000,’B’,200,’T’,10) ;

[X,Xp] = CGN(Xp) ;

% or

[X,Xp] = CGN() ;

### AddSig.m

[S] = AddSig(X,Y) Computes the sum Z of the two signals X and Y.

Inputs

X, Y Signal structures with fields:

- data : 1-by-N vector containing the data samples

- time : 1-by-N vector containing the time samples

- Fs : scalar indicating the sampling frequency

X and Y must have same lenghths and same sampling frequencies

Outputs

Z Signal structures with fields:

- data : 1-by-N vector containing the data samples

- time : 1-by-N vector containing the time samples

- Fs : scalar indicating the sampling frequency

### BPF.m

[Y,Fp] = BPF(X,Fp) filters the signal structure X with a digital band-pass filter whose parameters are specified in the Fp structure.

BPF diplays in a single window plot, the zero-pole diagram, the frequency and the impulse responses of the filter, and superimposed, the input and the output signals.

Inputs

- X input signal structure with the following fields:

- data : 1-byN vector containing the data samples

- time : 1-byN vector containing the time samples

- Fs : scalar indicating the sampling frequency

- Fp parameter structure with following fields:

- Fs : scalar indicating the sampling frequency (must beidentical to that of X)

- F0 : the central frequency (in Hz)

- Dnu : the bandwidth

- order : integer corresponding to the order of the filter

- class : text string indicating the type of the filter.

Outputs

- Y output signal structure with the following fields:

- data : 1-byN vector containing the data samples

- time : 1-byN vector containing the time samples

- Fs : scalar indicating the sampling frequency

- Fp parameter structure (same as input)

Example:

Fp = struct(’Fs’,1000,’F0’,100,’Dnu’,32,’order’,6,’class’,’BP filter’) ;

[X,Xp] = CGN() ;

Y = BPF(X,Fp) ;

### SquareSig.m

[Y] = SquareSig(X) Computes the square amplitude Y of signal X.

Inputs

X Signal structure with fields:

- data : 1-by-N vector containing the data samples

- time : 1-by-N vector containing the time samples

- Fs : scalar indicating the sampling frequency

Outputs

Y Signal structures with fields:

- data : 1-by-N vector containing the data samples

- time : 1-by-N vector containing the time samples

- Fs : scalar indicating the sampling frequency

Example:

X = OOK() ;

Y = SquareSig(X) ;

### RCF.m

[Y,RCFp] = RCF (X,RCFp) filters the signal structure X with a digital lowpass RC filter whose parameters are specified in the RCFp structure.

The z-transform of a lowpass RC filter is equal to

H(z) = B(z)/A(z) = (1-a) / (1 - a z^(-1))

where a = exp(-T/RC), and T is the sampling period.

RCF displays in a single window plot, respectively the time and the frequency responses of the filter, the input signal and the output signal.

Inputs

- X input signal structure with the following fields:

- data : 1-byN vector containing the data samples

- time : 1-byN vector containing the time samples

- Fs : scalar indicating the sampling frequency

- RCFp parameter structure with following fields:

- Fs : scalar indicating the sampling frequency (must be identical to that of X)

- RC : scalar defining the time constant RC (must be larger than 1/Fs)

Outputs

- Y output signal structure with the following fields:

- data : 1-byN vector containing the data samples

- time : 1-byN vector containing the time samples

- Fs : scalar indicating the sampling frequency

- RCFp parameter structure (same as input)