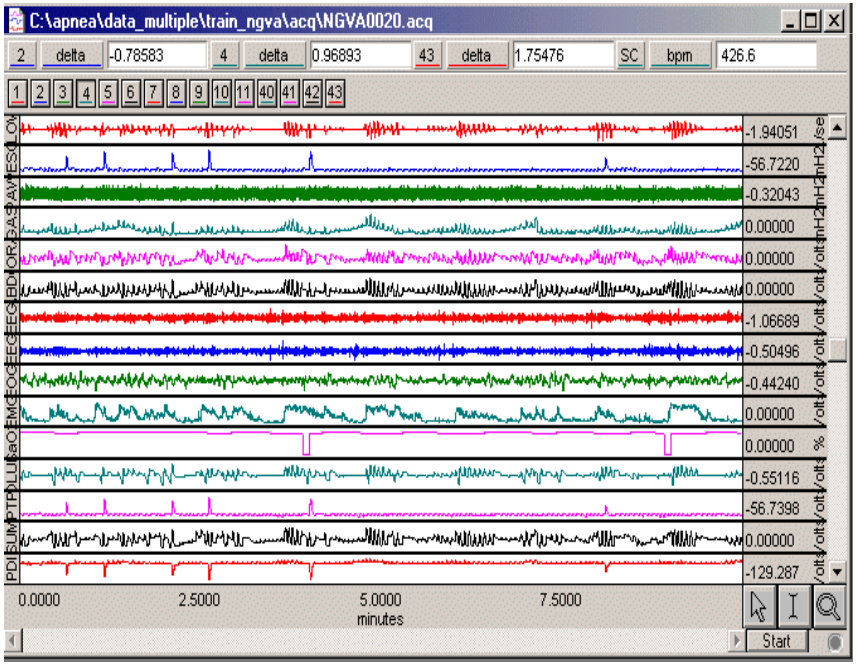
Traitement et analyse des signaux biomédicaux

Deuxième partie:

Méthodes linéaires classiques de traitement du signal dans le domaine de Neurophysiologie.



Tarik AL ANI, Département Informatique et Télécommunication (IT)
ESIEE-PARIS
E-mail : tarik.alani@esiee.fr
Url: http://www.esiee.fr/~alanit

Plan

EXEMPLES DE SIMULATION DE CE COURS :

Tous les exemples de simulation sont écrits en Scilab et fournis dans le dossier /simulation2/

Scilab (équivalent à Matlab) est un outil de calcul scientifique et de traitement de signaux. Cet outil est libre.
Version Scilab utilisée : Scialb-5.1.1 ou Scialb-5.2.2 :
http://www.scilab.org/

Le traitement et analyse du signal sont souvent plus facile si l'on peut supposer, signaux gaussiens stationnaires et linéaires. Mais nombreux signaux naturels ne sont pas stationnaires et/ou normal et/ou linéaires.

Il est important donc d'identifier la nature des signaux et éventuellement appliquer un prétraitement de faire des hypothèses plus simples.

● *Non-linéarité* peut être identifié simplement en regardant les diagrammes de dispersion, ou de distorsions harmoniques. En Matlab, voir par exemple : http://www.mathworks.com/matlabcentral/fileexchange/16062-test-of-non-linearity

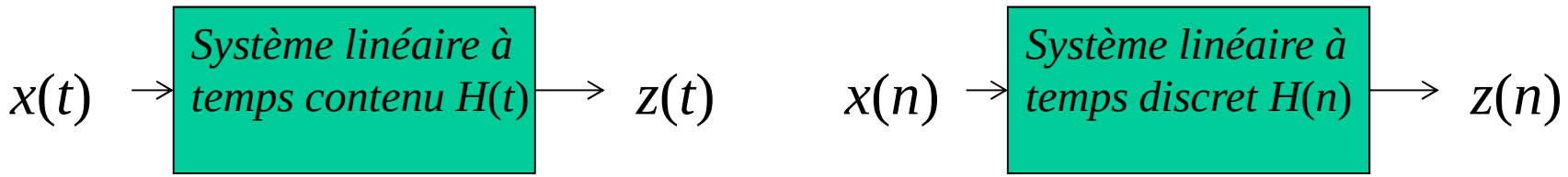
● Les propriétés non gaussiennes peuvent être identifiés en consultant l'histogramme. En Matlab, nous pouvons utiliser les fonctions « *kstest.m* » ou « *kstest2.m* » ou « *lillietest.m* » pour évaluer de la « normalité » quantitativement.

Toute analyse du signal commence par examiner les données.

0. Rappel de système linéaire et convolution

<u>Rappel</u> : *Système linéaire invariant dans le temps* (*LIT*) (Linear time-invariant system (*LTI*))

Ce système est défini par une fonction de mappage entrée-sortie $H(t)$ qui agit sur un signal d'entrée $x(t)$ pour produire un signal de sortie $z(t)$.



Rappel : Caractéristiques du système linéaire

- **Invariance dans le temps**

$$z(t-\tau) = H(x(t-\tau))=, \forall\ \tau$$

Si vous ajoutez un retard au signal d'entrée, alors vous ajoutez simplement le même retard à la sortie.

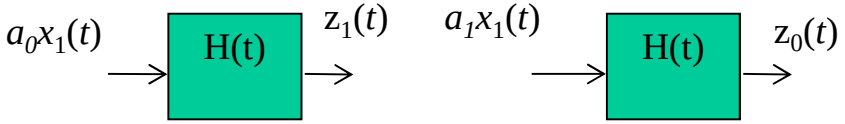
- **Causalité**

$$H(t)\neq 0,\ t \geq 0;\ H(t) = 0,\ \forall\ t < 0$$

- **Stabilité**

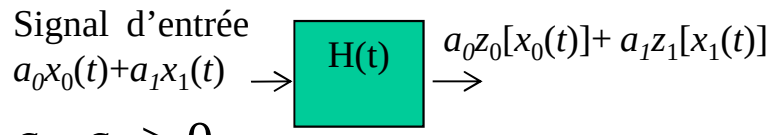
$$\sum_{n=-\infty}^{n=\infty} |H(t)| < \infty$$

- **Linéarité**



Soit $x_0(t)$ et $x_1(t)$ deux entrées différentes

H[.] est une transformation linéaire



$$H[a_0x_0(t) + a_1x_1(t)] = a_0z_0[x_0(t)] + a_1z_1[x_1(t)],\ \forall\ a_0, a_1 > 0$$

Les mêmes caractéristiques s'appliquent en temps discret (en remplaçant le variable indépendant « *t* » par « *n* ».

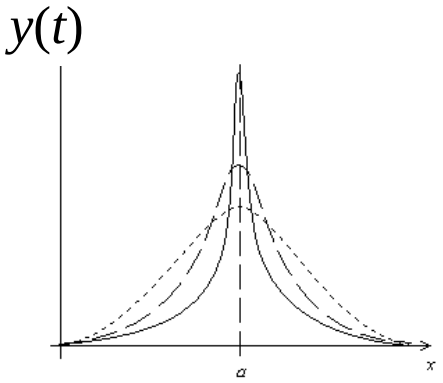
Fonction de Dirac

On dit qu'un signal correspondant à une distribution de Dirac $\delta(t)$ (noté aussi δ_t) a un spectre blanc : chaque fréquence est présente avec une intensité identique. Cette propriété permet d'analyser la réponse fréquentielle d'un système sans avoir à balayer toutes les fréquences.

δ_a = 1 ssi $t = a$, son intégrale vaut 1

= 0 sinon

On peut l'imaginer comme la limite d'une suite de courbes en cloche ou des rectangles ayant toutes la même surface 1, mais de plus en plus fines (donc de plus en plus hautes); lorsque la largeur des courbes tend vers 0, sa hauteur tend vers $+\infty$, mais la surface reste égale à 1.



Pour des raisons pratiques, on représente souvent la fonction de Dirac comme un bâton placé en a et de hauteur 1.

Réponse impulsionnelle du système

Si $x(t)$ est une impulsion de Dirac ($x_a(t) = 1$ ssi $t = a$, $x_a(t) = 0$ ailleurs), alors $H(t)$ est noté $h(t)$ est appelée la ***réponse impulsionnelle du système***.

La réponse impulsionnelle décrit la réaction du système en fonction du temps (ou éventuellement en fonction d'une autre variable indépendante qui paramètre le comportement dynamique du système).

Pourquoi la réponse impulsionnelle est utile?

- Il nous permet de prédire ce que la sortie du système va ressembler dans le domaine temporel.

Si on peut décomposer le signal d'entrée du système en une somme d'un tas de composants, alors la sortie est égale à la somme des sorties du système pour chacun de ces composants (Principes de la linéarité).

Et si nous pouvions décomposer notre signal d'entrée en une somme d'impulsions pondérées et décalées dans le temps ?

Alors, la sortie serait égale à la somme des copies de la réponse impulsionnelle, pondérées et décalées dans le temps de la même manière (Principes de la linéarité).

Pour les systèmes en temps discret, c'est possible, parce que nouss pouvons écrire tout signal $x[n]$ comme une somme de fonctions de Dirac (ou Kronecker) δ pondérées et décalées dans le temps.

$$x(n) = \sum_{m=0}^{\infty} x[m]\delta(n-m)$$

Chaque terme de la somme est une impulsion pondérée par la valeur de $x[n]$, à l'instant n.

Si nous passions $x[n]$ à travers un système *LIT* pour obtenir la sortie z[n] alors chaque impulsion pondérée et décalée dans le temps que nous introduisons donne à la sortie une copie pondérée et décalée dans le temps de la réponse impulsionnelle.

$$z(n) = x[n] * h(n) = \sum_{m=0}^{\infty} x[m]h(n-m)$$

C'est la ***convolution*** à l'instant n d'un système LIT à temps discret.

Convolution à temps continu

Soit $x(t)$ et $h(t)$ deux fonctions en temps continu, la convolution $z(t)$ de deux fonctions est définie par

$$z(t) = x(t) * h(t) = \int_{-\infty}^{+\infty} x(\tau)h(t-\tau)d\tau = \int_{-\infty}^{+\infty} x(t-\tau)h(\tau)d\tau$$

Convolution à temps discret

Soit $x(n)$ et $h(n)$ deux fonctions en temps discret, la convolution de deux fonctions $z(n)$ est définie par

$$z(n) = x(n) * h(n) = \sum_{m=-\infty}^{\infty} x[m]h(n-m) = \sum_{m=-\infty}^{\infty} h(m)x[n-m]$$

- $z(t)$ est en tout point de domaine commun à $x(t)$ et $h(t)$ est égale à l'intégrale (ou la somme si celui-ci est discret) sur l'entièreté du domaine d'une des deux fonction autour de ce point, pondérée par l'autre fonction autour de l'origine - les deux fonctions étant parcourues en sens contraire l'une de l'autre (nécessaire pour garantir la commutativité).
- Le produit de convolution généralise la représentation mathématique de la notion de *filtre linéaire*.

- Il s'applique aussi bien à des données temporelle (en traitement du signal par exemple) qu'à des données spatiale (en traitement d'image).

Rappel : Caractéristiques de convolution

Soit $x_0(t)$ et $x_1(t)$ deux entrées différentes

- **Commutativité**

$$x_0(t) * x_1(t) = x_1(t) * x_0(t)$$

- **Associativité**

$$[x_0(t) * x_1(t)\ ] * \mathrm{h(t)} = x_0(t) * [x_1(t) * \mathrm{h(t)}]$$

- **Distributivité sur l'addition**

$$x_0(t) * [x_1(t) + \mathrm{h(t)}] = [x_0(t) * x_1(t)\ ] + [x_0(t) * h(t)\ ]$$

- **Linéarité**

La convolution (*) est une transformation linéaire

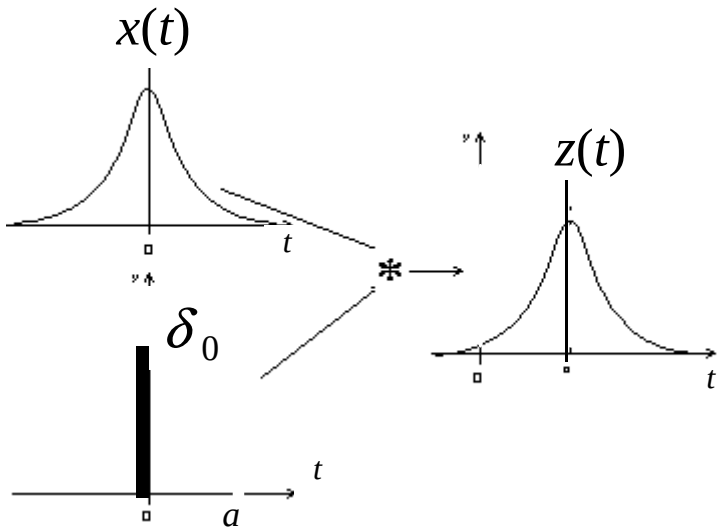
$[a_0x_0(t) + a_1x_1(t)]*\mathrm{h(t)} = a_0[x_0(t) *\mathrm{h(t)}] + a_1[x_1(t) *\mathrm{h(t)}]$, $\forall\ a_0, a_1 > 0$

Les mêmes caractéristiques s'appliquent en temps discret (en remplaçant le variable indépendant « t » par « n ».

- La fonction de Dirac (ou fonction impulsion) δ_0 est l'élément neutre de la convolution :

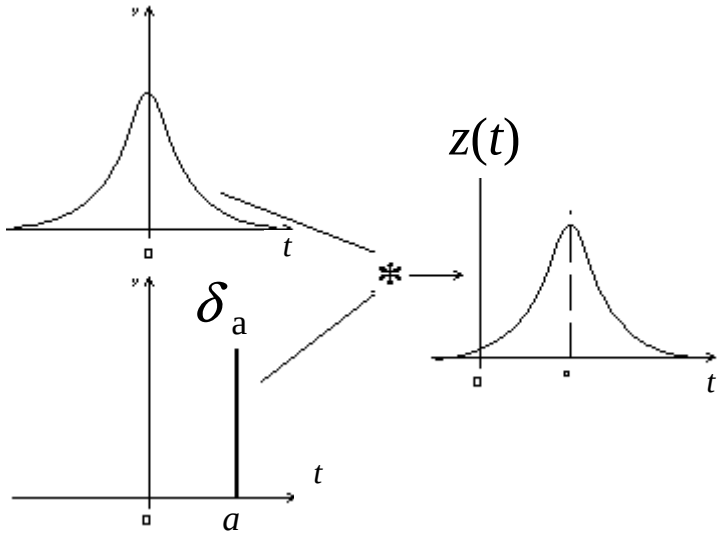
$z(t) = \delta_0 * x(t) = x(t) * \delta_0 = x(t)$

δ_0 laisse une fonction invariante.

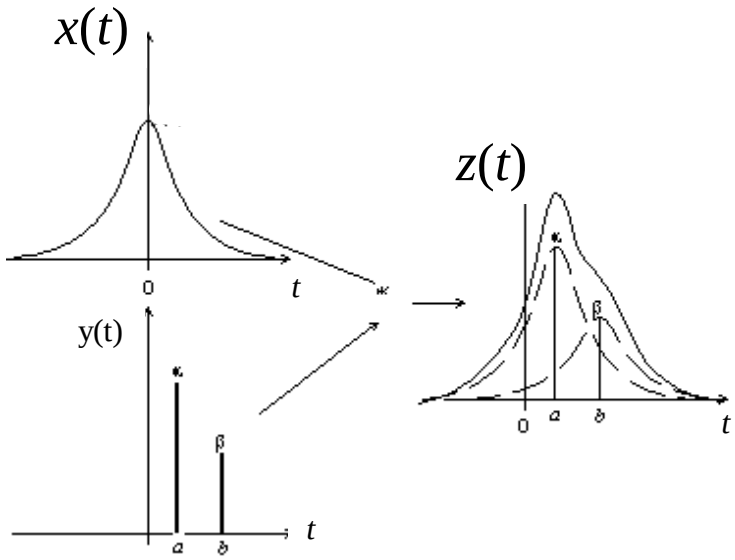


- Le produit de convolution par un dirac δ_a correspond à une translation pure de la fonction initiale d'une valeur de a.

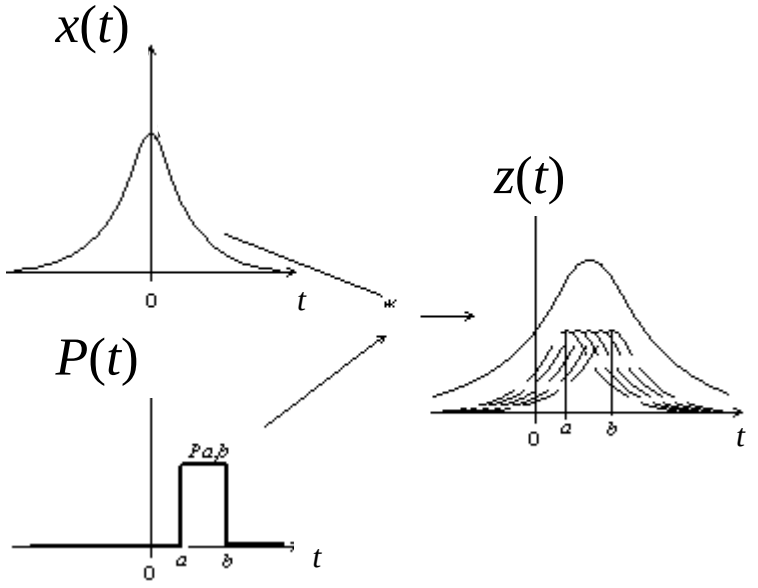
$z(t) = x(t) * \delta_a = \delta_a * x(t) = x(t-a)$



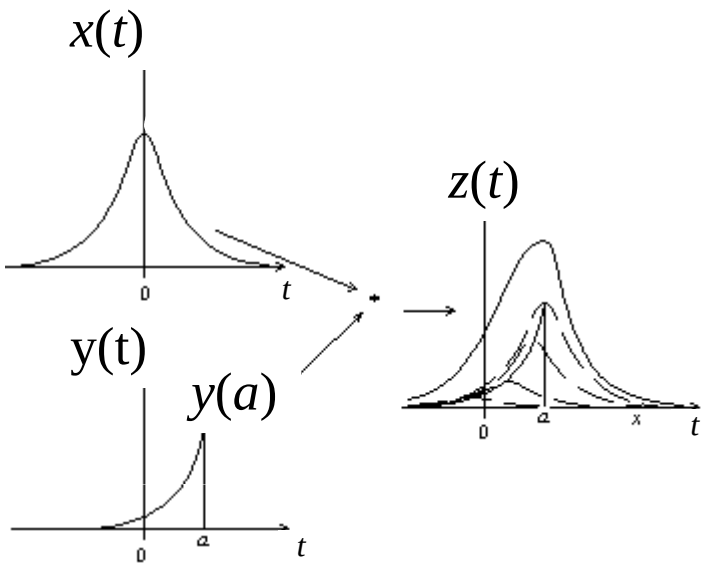
La Si l'on considère maintenant le produit de convolution par une somme pondérée de deux Diracs $y(t) = (\alpha.\delta_a + \beta.\delta_b)$, on obtient la superposition de deux courbes dilatées.



➢Considérons maintenant une fonction porte $y(t) = P_{a,b}$; c'est une fonction qui vaut $1/(b-a)$ entre a et b, et 0 ailleurs (son intégrale vaut 1). Cette fonction peut être vue comme une succession de Diracs. La convolution de $x(t)$ par $P_{a,b}$ va donc s'obtenir en faisant glisser $x(t)$ sur l'intervalle $[a;b]$. On obtient un « élargissement » de $x(t)$.



➢Si l'on considère maintenant une fonction quelconque $y(t)$, on peut voir $y(t)$ comme une succession de Diracs pondérés par la valeur de y au point considéré t. Le produit de convolution de $x(t)$ par $y(t)$ s'obtient donc en faisant glisser la fonction $x(t)$ et en la dilatant selon la valeur de $y(t)$.



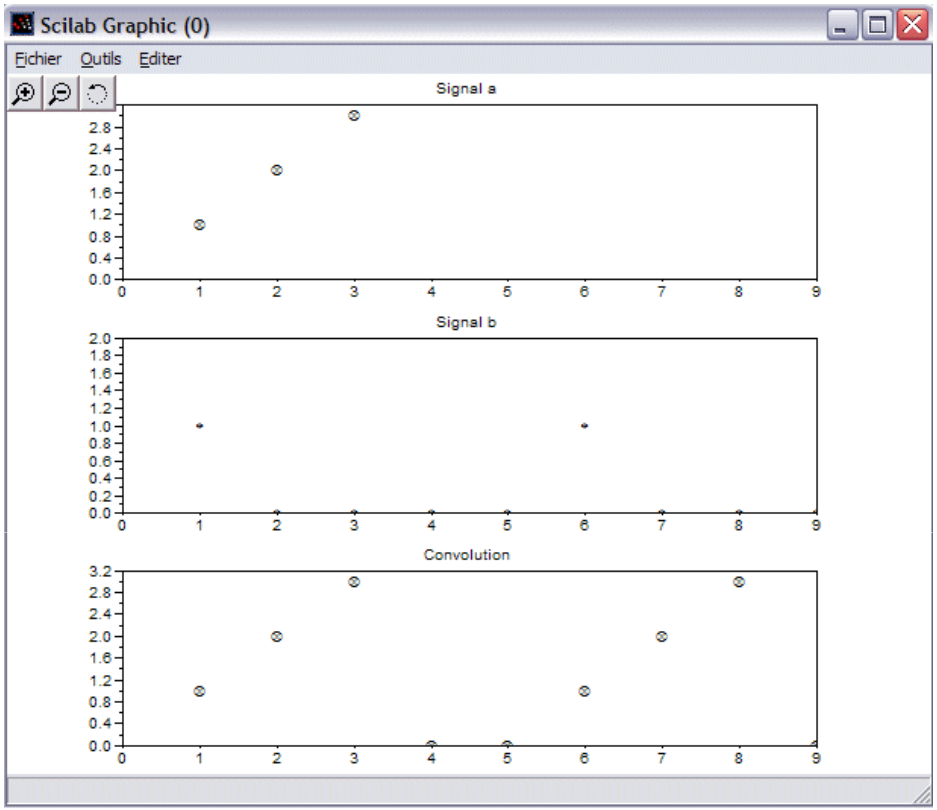
Applications de convolution

Exemple : conv1

Copier un signal dans le temps.

Source :

../simulation2/sim_convol/conv1.sce




```
x=[1 2 3]; // signal 1
y = [1 0 0 0 0 1 0 0 0]; // signal 2 de longueur
                         // différente  de celle de « x »
z = convol(x,y) // mixe les informations
                // contenues dans x et y
 z  =
!  1.    2.    3.    0    0    1.    2.    3.    0    0  !
```

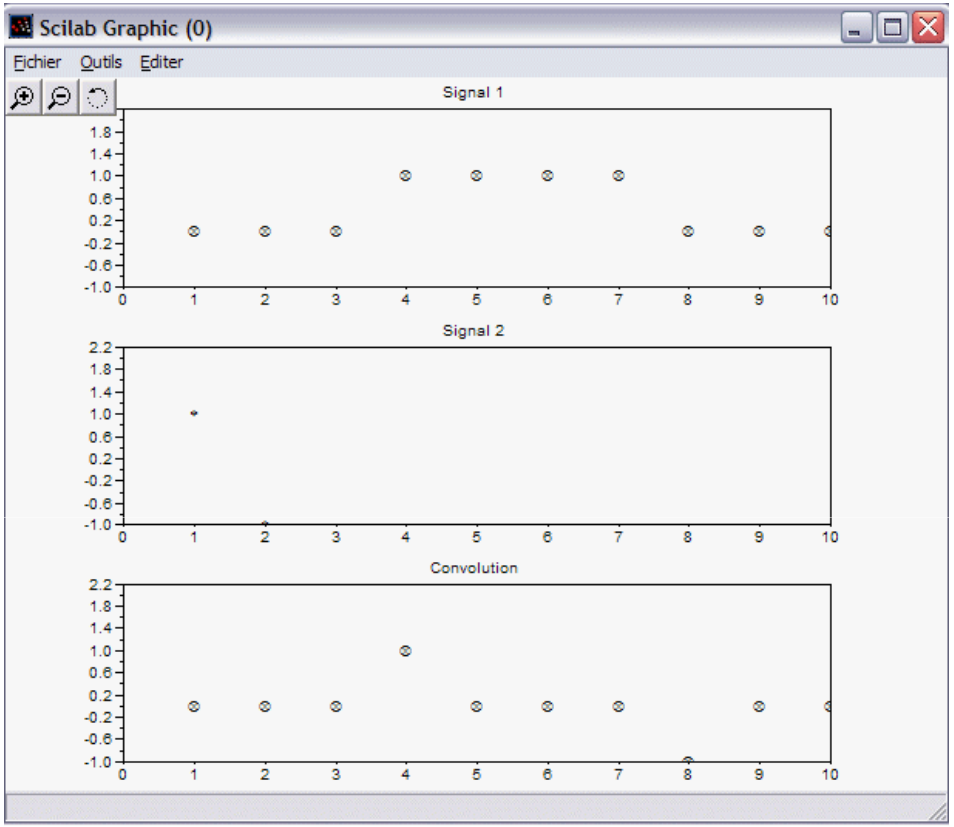
A chaque fois y =1, x est introduit dans « z ».

Exemple : conv2

Souligner le changement rapide du niveau d'un signal

source : ../ simulation2/sim_convol/conv2.sce

```
x=[0 0 0 1 1 1 1 0 0 0];  // signal 1 carré semblable à
                          // un signal de calibrage
y = [1 -1]; // signal 2 utilisé pour détecter où « x » est
            // constante
z = convol(x,y)
z  =
! 0. 0. 0. 1. 0. 0. 0. - 1. 0. 0. 0. !
```



Détecter les positions où le signal *x* n'est pas constant : filtrage de *x* avec un filtre passe-haut.

Selon les valeurs de *y* nous pouvons produire des effets très différents en *x*. Dans l'exemple précédent nous avons éliminé les changements lents mais nous pouvons diminuer les changements rapides aussi. En Neurophysiologie Clinique, la lissage (smoothing) est fréquemment employée pour éliminer les artefacts à haute fréquence.

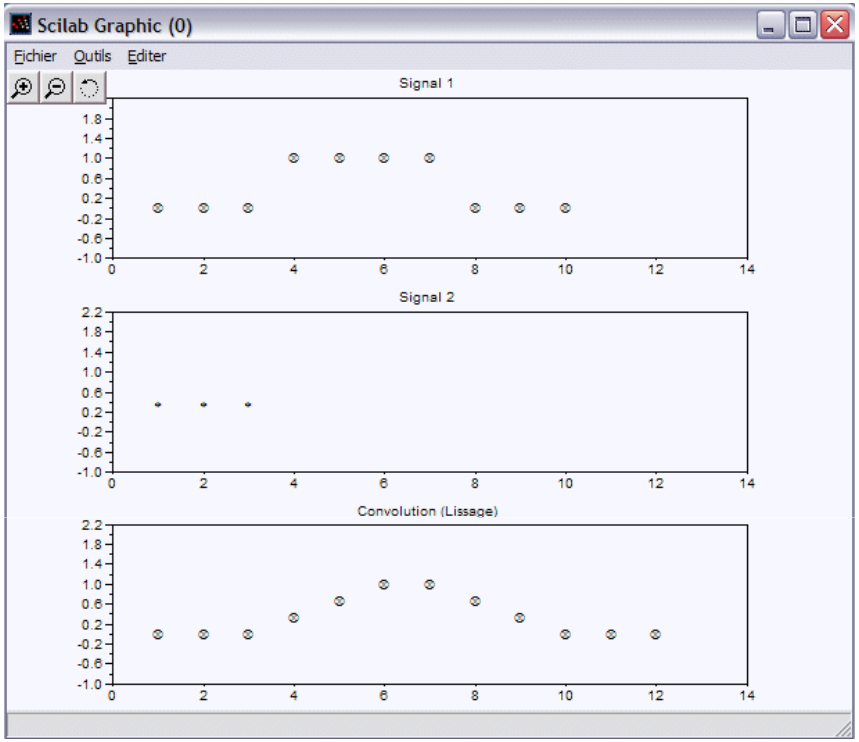
Exemple : conv3

Lissage

Le vecteur suivant permet de lisser les signaux. Chaque point obtenu est la moyenne de ce point et de ces deux voisins (avant et après).

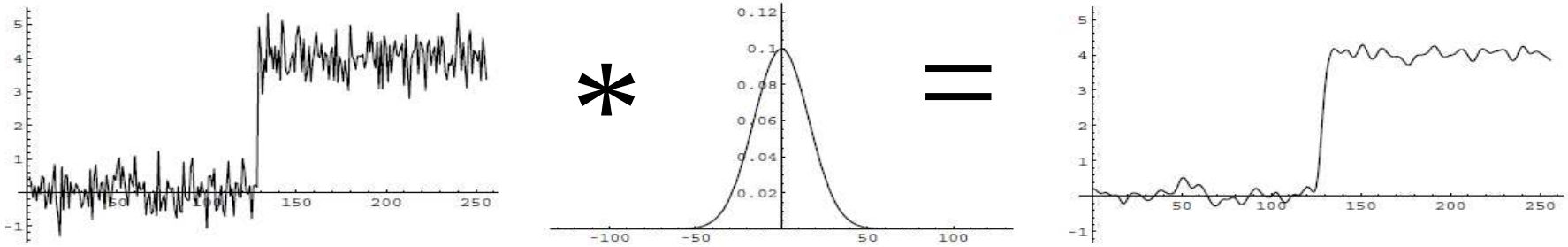
source : ../ simulation2/sim_convol/conv3.sce

```
x=[0 0 0 1 1 1 1 0 0 0]; // signal 1 carré
y = [1 /3 1/3 1/3]; // signal 2 utilisé pour lisser « x »
z = convol(x,y)
c=
!0 0 0 0.3333333 0. 6666667   1 1 0. 6666667 0.3333333 0 0 0!
```

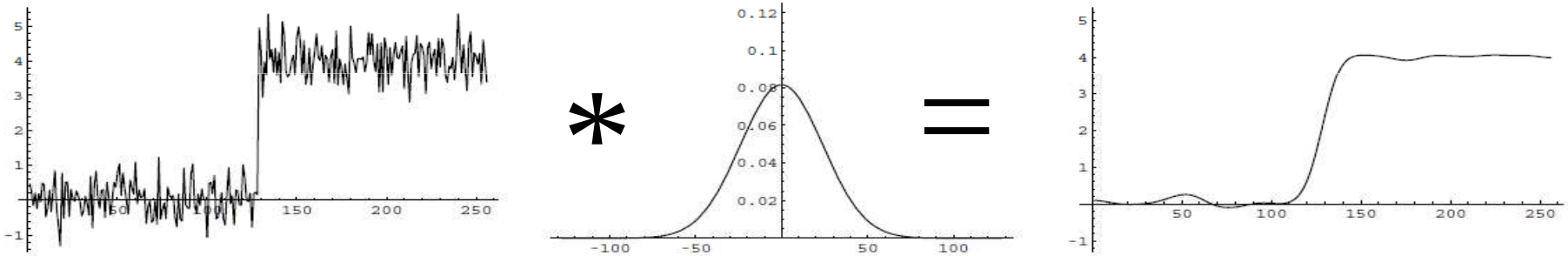


Nous avons éliminé les changements brusques qui caractérisent la transition entre le « 0 » et le « 1 » et entre le « 1 » et le « 0 ». Autrement dit, nous avons filtré le signal avec un filtre passe-bas.

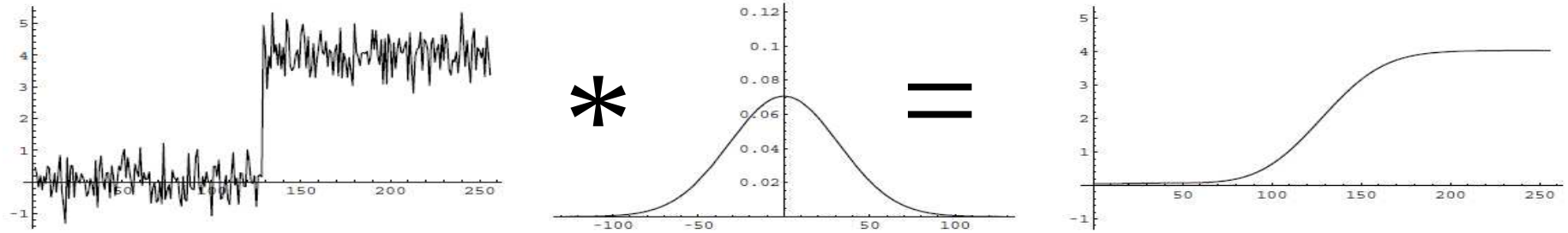
Exemple : Filtrage du bruit par Convolution



Signal échelon bruité $x(t)$ | Noyau (kernel) :Gaussienne, $N(m,\sigma) = (0,16)$ | Signal nettoyé $z(t) = x(t)*N(m,\sigma)$



Signal échelon bruité $x(t)$ | Noyau (kernel) :Gaussienne, $N(m,\sigma) = (0,24)$ | Signal nettoyé $z(t) = x(t)*N(m,\sigma)$

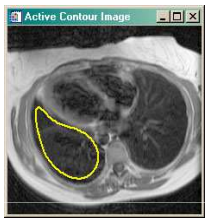


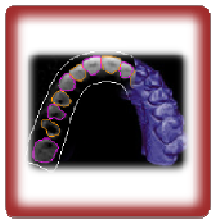
Signal échelon bruité $x(t)$ | Noyau (kernel) :Gaussienne, $N(m,\sigma) = (0,32)$ | Signal nettoyé $z(t) = x(t)*N(m,\sigma)$

Exemple conv3 :

Gradient et accentuation de bord (en image : contour), en présence du bruit.

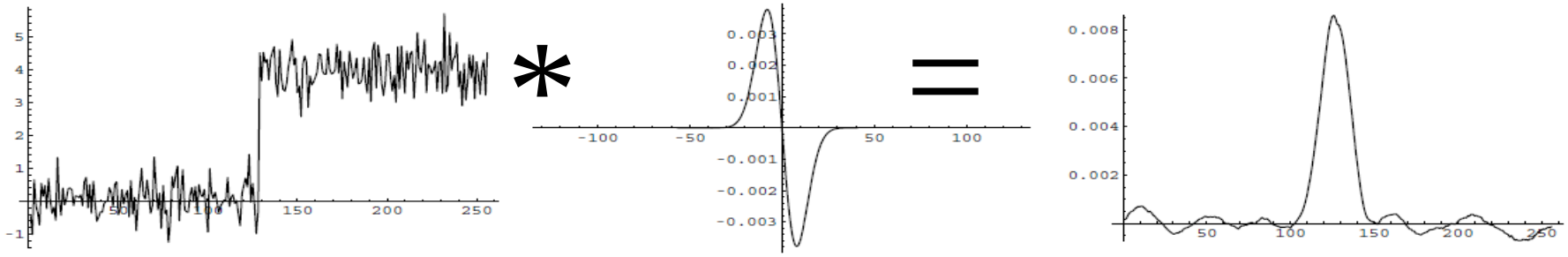
Pour détecter le changement, on peux applique la dérivée sur le signal $x(t)$ mais la dérivée augmente le bruit. Alors, il est préférable de lisser $x(t)$ avant appliquer la dérivée.





La dérivée de $x(t)$ convoluée avec un noyau $h(t)$
est équivalente à $x(t)$ convolué avec la dérivé de $h(t)$

$$h(t) * \frac{\partial}{\partial t} x(t) = \frac{\partial}{\partial t} h(t) * x(t)$$



Signal échelon bruité $x(t)$ dérivée de Gaussienne, $N(m,\sigma)$= (0,16) dérivée de $x(t)$

Exercice : Différenciation par convolution

1. Créer une fonction Matlab

[*z*] = *GaussFilt*(*y*,*sigma*,*type_filtre*)

qui applique au signal d'entrée « *y* » (un vecteur ligne) :

- une fonction gaussienne « *gauss* » normalisée et centrée (moyenne nulle) avec un écart-type donné « *sigma* ». Utiliser une plage largueur paramétrée par une valeur constante donnée « *size* » initialisé à 30 pour générer un vecteur de valeurs (- *size*/2 ≤ n ≤ *size*/2) qui représentent les variables introduites dans « *gauss* ».
- un paramètre indiquant le type de filtre :

  - si *type_filtre* =1 (utiliser la fonction matlab « filter » :

  z = filter (gauss,1, y);

  - si *type_filtre* =2 (utiliser la fonction « conv » :

  z = conv (*y*, *gauss*, '*same*');

Exercice : Différenciation par convolution (suite)

2. Créer un script Matlab pour appeler la fonction « *GaussFilt* ». Commencer par les lines suivantes :

   close all;

   randn('seed',10);

- Créer et appeler la fonction « *vepot.m* » (voir page suivante) en introduisant *fs=128; N=300;*
- Indiquer le paramètre *type_filtre* =1 ou 2
- appeler la fonction [*z*] = *GaussFilt(y,sigma,type_filtre)*

en introduisant *sigma = 10; et y=y1; ou y=y2;*

- Tracer le signal filtré *z*

Expérimenter en modifiant le paramètre *sigma*, conclure.

Exercice : Différenciation par convolution (suite)

```
function [y1,y2] = vepot(fs,N)
%Simulation d'un signal Visual Evoked Potential (VEP) bruité
% T. AL-ANI
% Novembre 2013
% Entrées :
% fs : fréquence d'échantillonnage (exemple fs=128)
% N : nombre d'enregistrements (exemple N=300)
%
% Sorties :
% y1 : un enregistrement (le premier)
% y2 : le médiane de N enregistrements

cleanvep =2*(sin(pi*[0:127]/fs).^2).*(cos(2*pi*[0:127]/fs));
cleanvep = [zeros(1,36),cleanvep,zeros(1,36)];
figure;plot(cleanvep,'b');
title('VEP non bruité (théorique)')
xlabel('Time (sec)');
ylabel('Amplitude (microVolts)');
y1 = cleanvep + 2 * randn(size(cleanvep));
figure;plot(y1,'k');
title('Une réalisation de VEP bruité')
xlabel('Time (sec)');
ylabel('Amplitude (microVolts)');
tampon=y1;
% ajoute "sweeps" avec bruit (pour simuler 300 fois des signaux bruités VEP)
for k=1:N-1
    noisevep = cleanvep + 2 * randn(size(cleanvep));
    tampon = [tampon ; noisevep];
end;
%
% calculer la médian de 300 réalisations de VEP bruité
y2 = median(tampon, 1);
figure;plot(y2,'r');
title('Médiane de 300 réalisations de VEP bruité')
xlabel('Time (sec)');
ylabel('Amplitude (microVolts)');
```

Filtrage par convolution

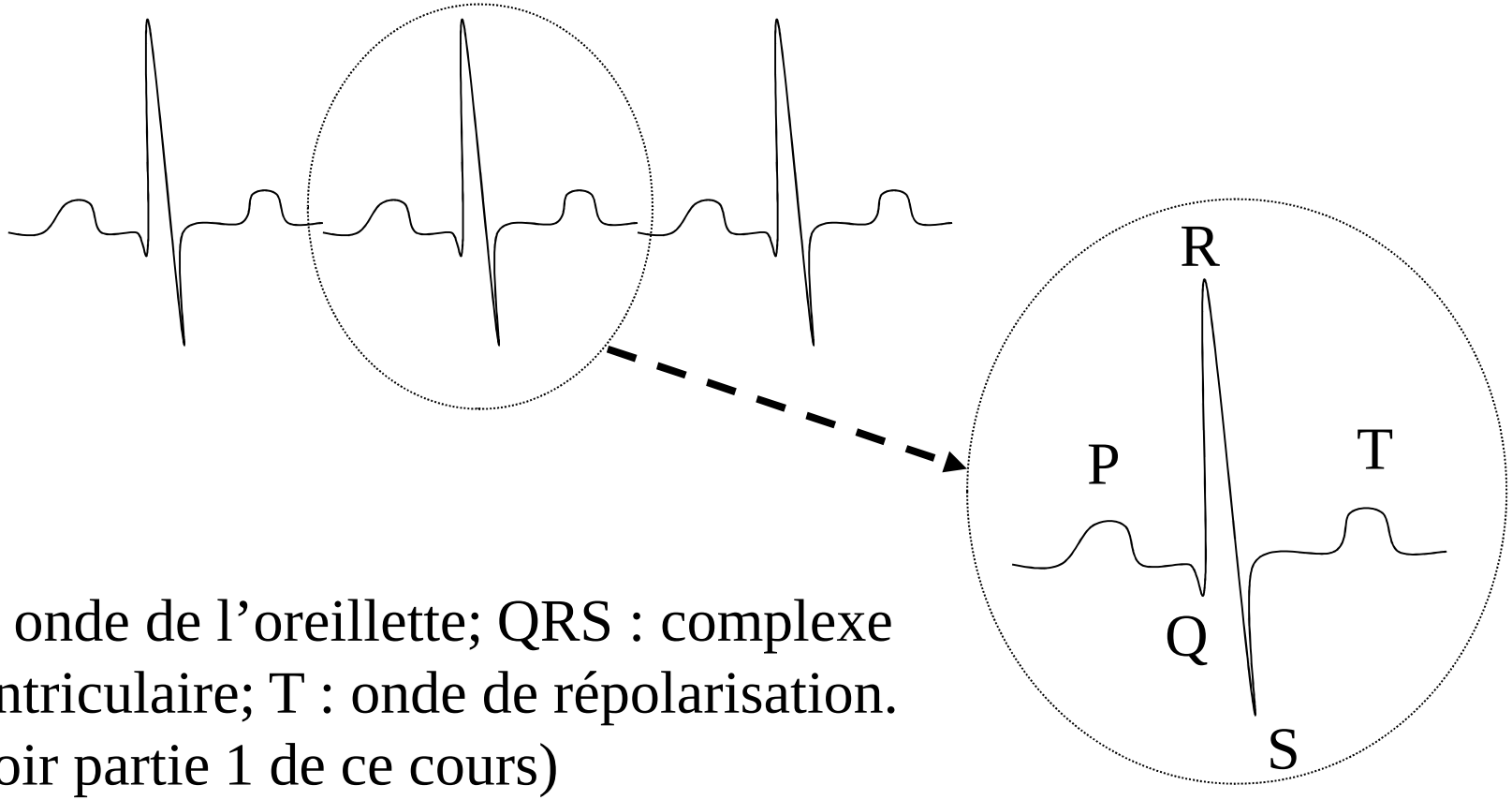
Exemple

Nous sommes intéressés par **accentuer les hautes fréquences d'un signal ECG**. Pour cela, il pourrait être intéressant d'éliminer les composantes lentes (basse fréquence) qui pourraient interférer dans le processus de détection des **complexes QRS**. Pour faire cela, nous utiliserons l'opération de convolution. Charger le signal ECG à partir de «signaux» donnée par le chemin (../simulation2/Scilab/sim_binaire_edf/) (c'est le même fichier que nous avons sauvée dans la première partie de ce cours au sujet du format EDF). Le signal ECG est le numéro 12 dans la liste : ecg=signaux(12) en Scilab ou ecg=signaux(12,:) en Matlab.

Filtrage par convolution (suite)

Exemple (suite)

Electrocardiogramme normal



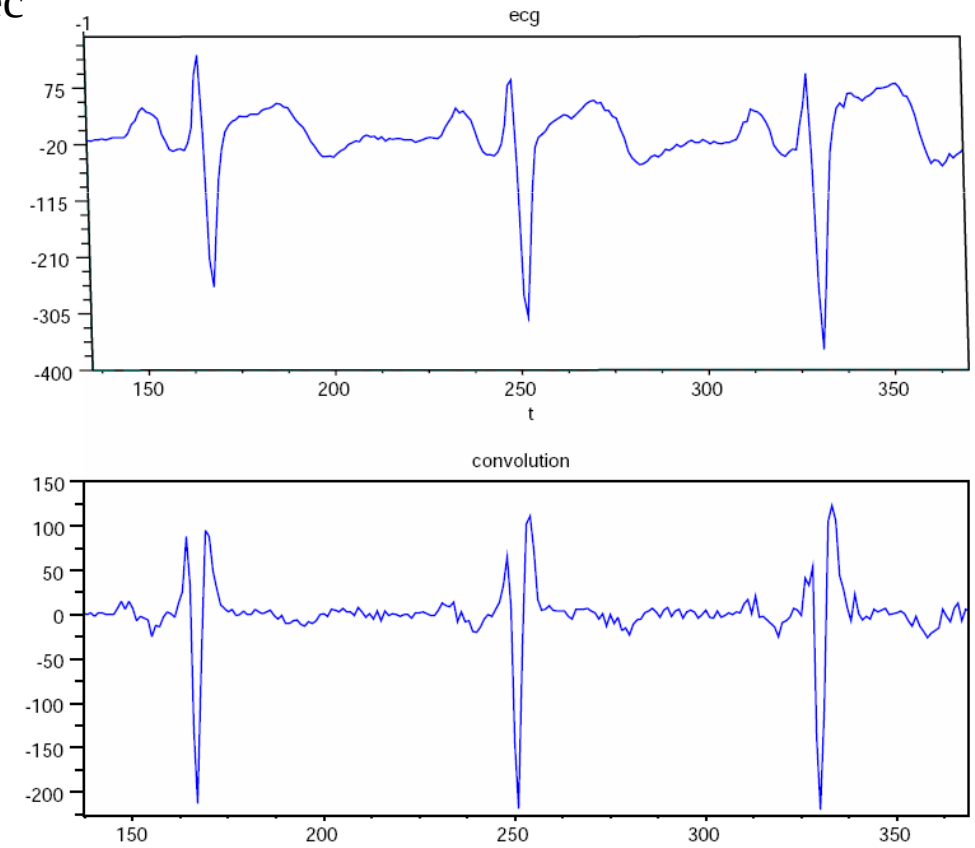
P : onde de l'oreillette; QRS : complexe ventriculaire; T : onde de répolarisation. (Voir partie 1 de ce cours)

Filtrage par convolution (suite)

Exemple (suite) : ecg_convol.sce

source : ../ simulation2/sim_ecg_convol_correl_align/ecg_convol.sce

```
xbasc();
stacksize(20000000)
my_dir=get_absolute_file_path('ec
path=string(my_dir);
load(path+'signals');
ecg = signals(12);
clear('signals');
ecgconv = convol ([1,-1], ecg);
xsetech([0,0,1,1/2]);
plot(ecg(1:1000));
xtitle('ecg','t');
xsetech([0,1/2,1,1/2])
plot(ecgconv(1:1000));
xtitle('convolution','t');
```



Synthèse des signaux par convolution

La convolution permet aussi de **créer des transitoires répétitives**.

Exemple : Simulation d'un signal EMG

Cet exemple illustre la convolution de deux signaux: le premier représente une forme semblable à un **potentiel d'unité motrice (PUM)** (motor unit potential (mup)) et l'autre représente une forme semblable au **taux de décharge rythmique** (rythmical discharge rate) de cette unité. Notre but est donc de simuler un signal EMG lié à un effort faible. Nous allons échantillonner le signal à 20 kilohertz, ainsi l'intervalle d'échantillonnage est 0.05 ms. Nous allons représenter un segment de 500 ms (0.5 s). Nous construisons ensuite deux vecteurs : le premier contient le temps et l'autre contient une forme semblable à un potentiel d'unité motrice.

Unité motrice : Ensemble des fibres musculaires innervées par un même motoneurone, qui se contractent simultanément et sont à l'origine d'un potentiel d'unité motrice.

Synthèse des signaux par convolution

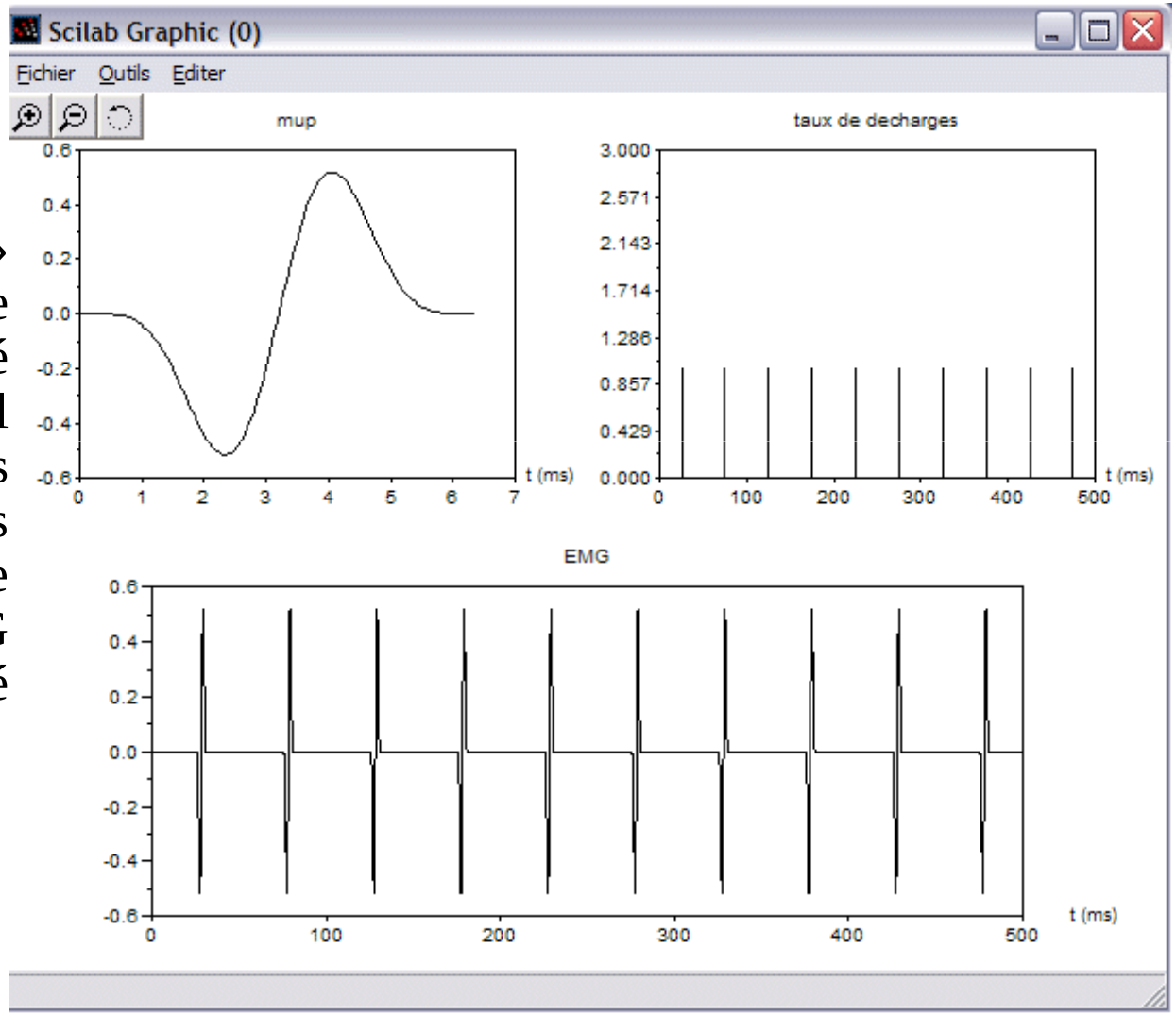
Exemple (suite)

source : ../ simulation2/sim_convol/Scilab/

```
t = [0:.05:500];
// potentiel d'unité motrice
mup = - (sin(%pi*[0:127]/128).^4) .* (sin(2*%pi*[0:127]/128));
decharges = [500 : 1000 : 9500];
taux= zeros(1,length(t));
taux(decharges)=1;
emg=convol(taux,mup); // signal EMG
xsetech([0,0,1/2,1/2]);
plot2d(t(1:length(mup)),mup);
xtitle('mup','t');
xsetech([1/2,0,1/2 1/2]);
plot2d(t(1:length(taux)),taux,1,"011","a",[0,0,500,3],[5,5,0,0]);
xsetech([0,1/2,1,1/2]);
plot2d(t(1:10000),emg(1:10000));
xtitle('EMG','t');
```

Synthèse des signaux par convolution

• Chaque valeur de « 1 » dans le trame de décharge devient un potentiel d'unité motrice dans le signal d'EMG. Si nous avions ajouté plusieurs unités avec différentes formes de taux de décharge un EMG plus réaliste aurait été obtenue.

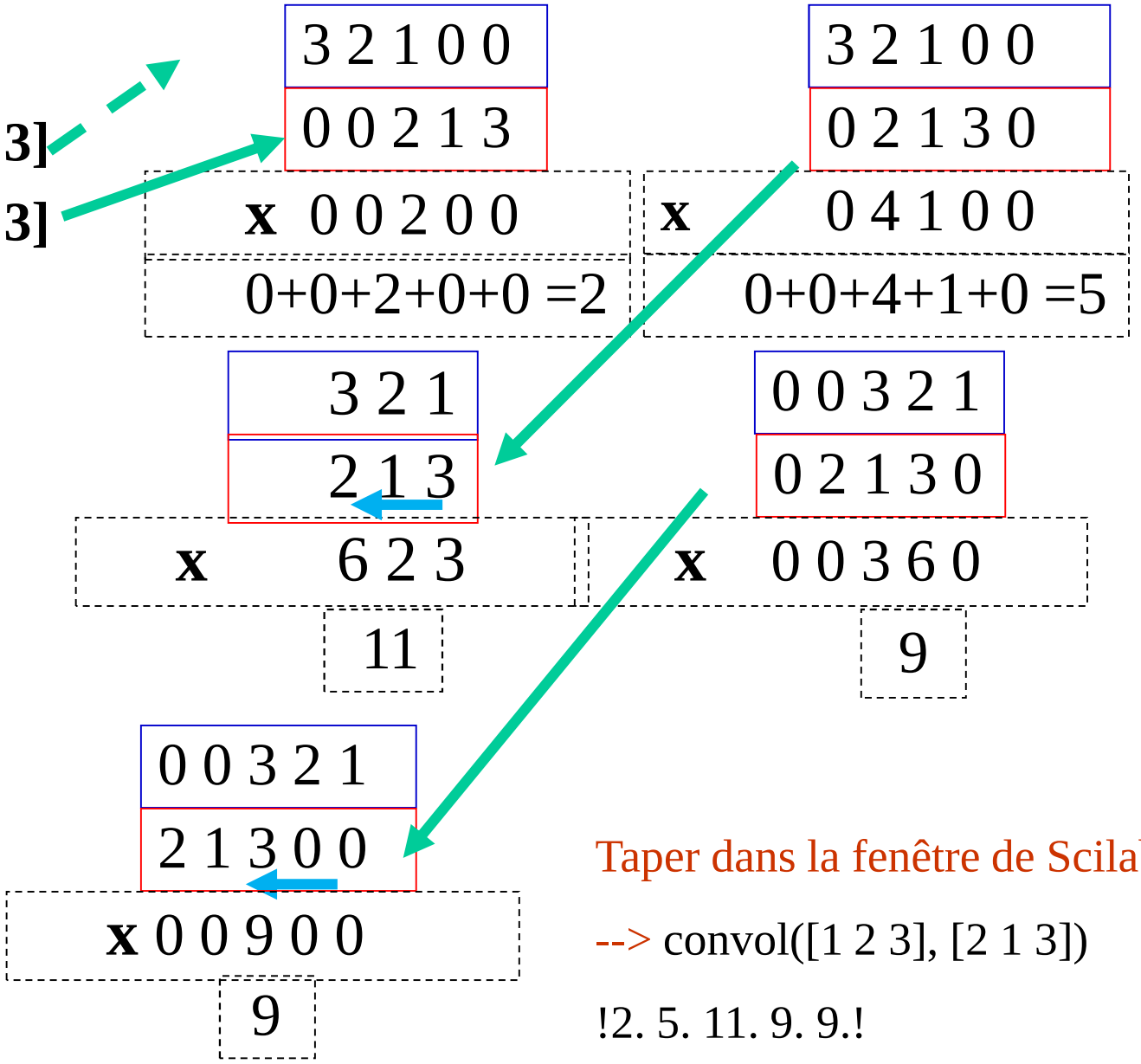


Mécanisme de calcul numérique de convolution

Exemple

Signal x : [**1 2 3**]

Signal y : [**2 1 3**]



Taper dans la fenêtre de Scilab :

--> convol([1 2 3], [2 1 3])

!2. 5. 11. 9. 9.!

Convolution et produit des polynômes

Exemple

x = [1 2 3]; y = [2 1 3];

xpoly = **poly** (x,'x', 'z')

xpoly =

$1+2x+3x^2$

ypoly = **poly** (y, ‘x', 'z')

ypoly =

$2+x+3x^2$

zpoly = xpoly **x** ypoly

zpoly =

$2 + 5x + 11x^2 + 9x^3 + 9x^4$

Taper dans la fenêtre de Scilab : **coeff** (zpoly)

ans =

! 2. 5. 11. 9. 9. !

convol (x, y)

ans = ! 2. 5. 11. 9. 9. ! // length(ans)=length (x)+length (y)-1

Synthèse des signaux par convolution

L'inverse du processus ci-dessus s'appelle la « **déconvolution** » et implique l'extraction de la forme de différents PUM participants au signal EMG. Le modèle PUM contient l'information sur la structure de l'unité motrice tandis que le modèle de décharge contient l'information sur le fonctionnement de l'activité musculaire. Les deux modèles sont mélangés pour construire l'EMG et contiennent l'information utilisée dans le diagnostic neuromusculaire.

Exercice : déconvoluer le signal « EMG » de « PUM » puis de « decharges » l'exemple précédent.

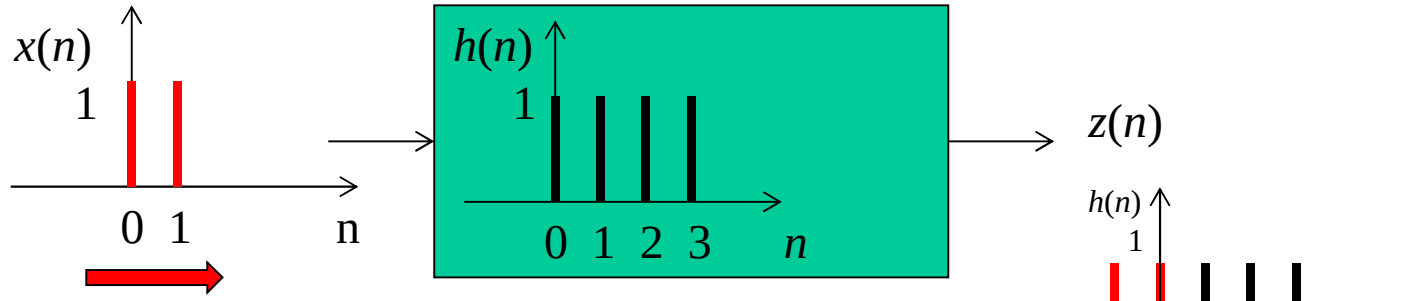
En Matlab : deconv.m

Aide en Scilab :

https://groups.google.com/forum/?fromgroups=#!topic/comp.soft-sys.math.scilab/hqVqCS4uNJQ

Synthèse des signaux par convolution

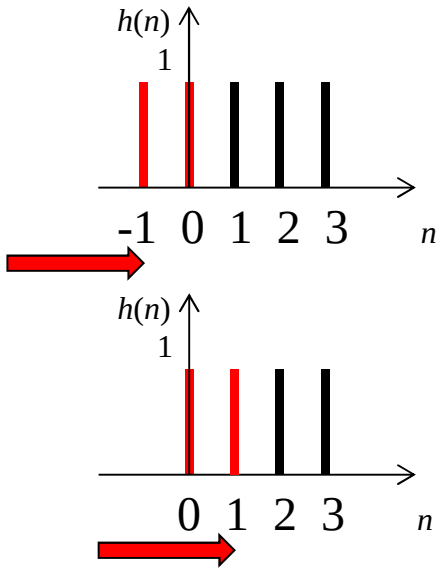
Exercice : Utiliser la formule de convolution en temps discret (voir page 10) pour calculer la convolution z entre la réponse impulsionnelle $h(n) = [1\ 1\ 1\ 1]$ et le signal d'entrée $x(n) = [1\ 1]$. Tracer la sortie $z(n)$ contre n à chaque valeur de $n = 0, 1, 2, 3, 4$. Considérer que $x(n)$ qui se déplace par rapport à $h(n)$ avec $\Delta n = 1$ de $-\infty$ à $+\infty$.



Exemple :

$n = 0$ $m = -1, m = 0$

$z_0 = x(-1)h(0-(-1)) + x(0)h(0-0) = 1 \times 0 + 1 \times 1 = 1$



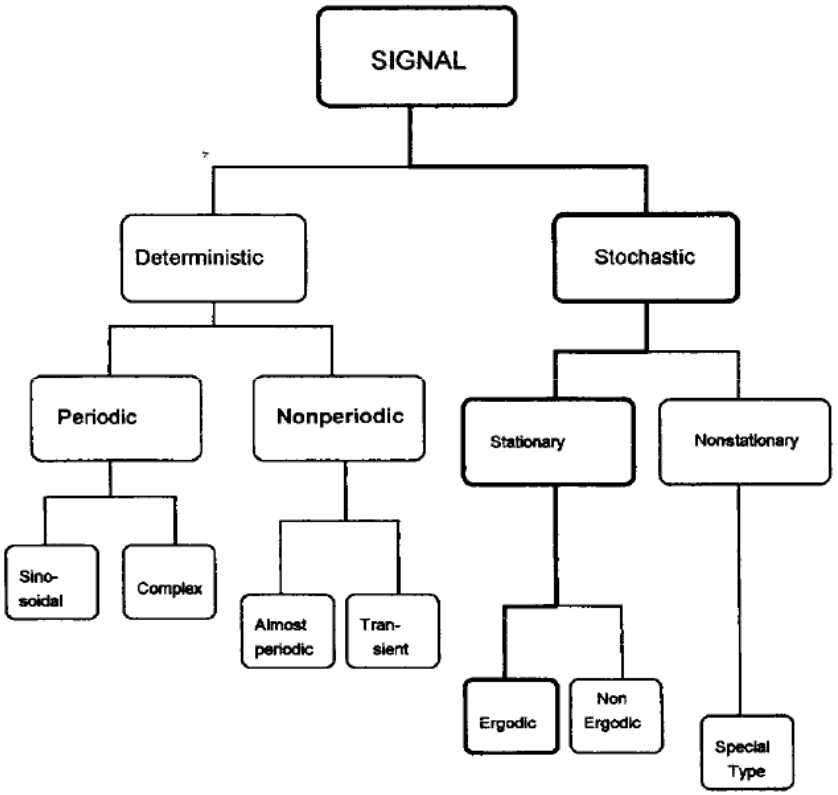
Exercice : Créer un code implémentant en Matlab l'exercice précédent en utilisant les 2 fonctions :

1. la fonction Matlab « conv.m »
2. votre propre fonction (utiliser l'équation de convolution en temps discret)

Notion de processus stochastiques

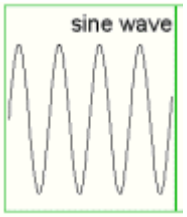
Les natures de biosignaux

- **Déterministe**
  - **Périodique** (onde qui se répète indéfiniment dans
  - **Non périodique**
    - **Transitoires** (durée finie)
- **Presque périodique**
- **Stochastique**
  - Stationnaire (les propriétés statistiques ne change pas dans le temps.
  - Non stationnaire (les propriétés statistiques change dans le temps.

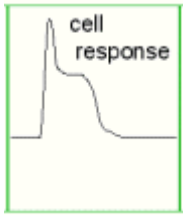


Adapté de Tfy-99.3275 Biosignal Processing
https://noppa.aalto.fi/noppa/haku/Tfy-99.3275

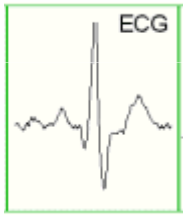
- **Déterministe**
  - **Périodique**



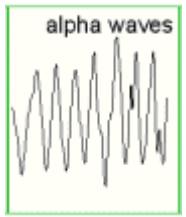
  - **Non périodique**
    - **Transitoires** (durée finie)

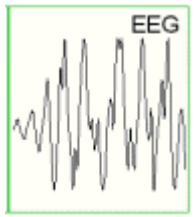


- **Presque périodique**



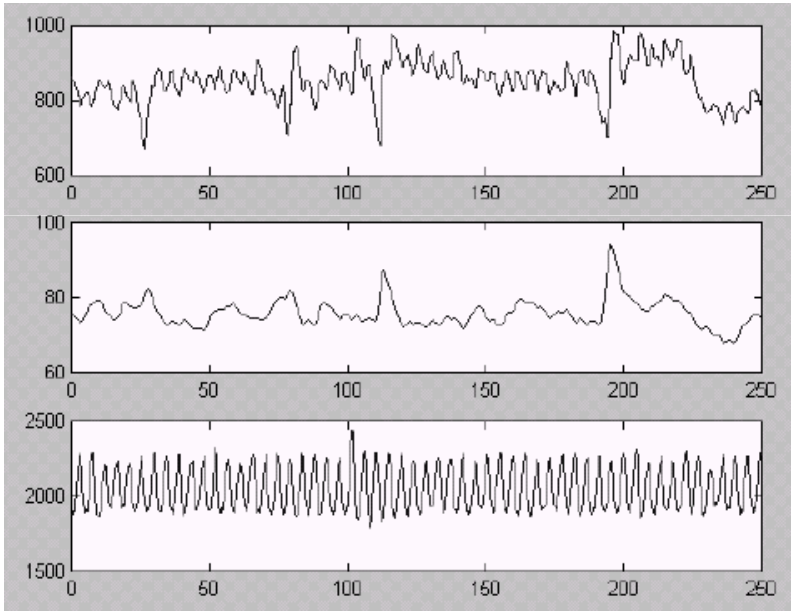
- **Stochastique**
  - **Stationnaire**





  - **Non stationnaire**

Il est rare que les signaux naturels, notamment les biosignaux, sont de nature déterministe. Ces signaux résultent souvent de causes multiples et souvent incontrôlables et non mesurables. Ces signaux alors ne puissent pas être caractérisés par une expression analytique simple. On les qualifie alors de ***stochastiques***.



Exemples de signaux biomédicaux. Haut : intervalles entre les battement du cœur (ms) ; Centre : Pression artérielle moyenne (mm Hg); Bas : Volume respiratoire.

Nous présentons les propriétés théoriques des signaux aléatoires à valeurs contenues (cas de biosignaux) : *moyenne*, *variance*, *densité de probabilité*, *autocorrélation*, *corrélation croisée*, *covariance*, *densité spectrale de puissance*, *spectrogramme et cohérence* .

Rappel : Variables aléatoires (v.a.) continues

Une variable aléatoire continue X est complètement caractérisée par sa ***fonction de densité de probabilité*** $f_X(x)$ qui calcul la probabilité que X appartienne à l'intervalle (x_1, x_2) :

$$f_X(x) = P(X \in (x_1, x_2)) = P(x_1 \leq x \leq x_2) = \int_{x_1}^{x_2} f_X(x)dx$$

Sa ***fonction de répartition*** :

$$F_X(x) = P(X \leq x) = \int_{-\infty}^{x} f_X(x)dx$$

Rappel : Variables aléatoires continues (suite)

On caractérise souvent une variable aléatoire continue par sa ***moyenne*** $\mu_X = E[X]$ et sa ***variance*** Var_X, définies par :

$$\mu_X = E[X] = \int_{-\infty}^{+\infty} x\, f_X(x)\, dx$$

$$Var_X = \sigma_X^2 = E[(X - m_X)^2] = \int_{-\infty}^{+\infty} x^2\, f_X(x)\, dx$$

Où σ_X^2 est l'écart type qui est une mesure de la dispersion d'une variable aléatoire réelle.

On caractérise complètement la relation statistique entre 2 variables aléatoires continues X et Y par une ***fonction de densité de probabilité conjointe*** $f_{XY}(x, y)$:

$$f_{XY}(x, y) = P(X \in (x_1, x_2), Y \in (y_1, y_2)) = P(x_1 \le x \le x_2, y_1 \le x \le y_2)$$

$$= \int_{x_1}^{x_2} \int_{y_1}^{y_2} f_X(x)\, f_Y(y) dx dy$$

Rappel : Variables aléatoires continues (suite)

On caractérise souvent (une estimation) $f_{XY}(x, y)$ de façon plus simple par sa ***corrélation*** σ_{XY}, ou par *sa* ***covariance*** ρ_{XY} entre X et Y, ou par la covariance normalisée entre –1 et 1 appelée ***coefficient de corrélation*** r_{XY} .

Deux variables sont dites ***indépendantes*** ssi la valeur prise lors du tirage aléatoire de la première n'a aucune influence sur le tirage de la seconde : $f_{XY}(x, y) = f_X(x)\, f_Y(y)$

Elle sont dites ***non corrélées*** ssi : $\rho_{XY} = 0$

Deux variables aléatoires indépendantes sont non corrélées, mais qu'en général la réciproque est fausse.

Rappel : Variables aléatoires continues (suite)

Comme pour les variables déterministes, on peut effectuer une opération algébrique sur les variables aléatoires. Par exemple,

- Somme Z de deux variables aléatoires indépendantes X_1 *et* X_2 .

Y *est* une variable aléatoire, et sa $f_Z(z)$, la probabilité d'une valeur particulière z *de* Z est bien entendu la somme (continue) des probabilités de chaque paire de valeurs particulières (x_1,z-x_1) dont la somme donne z.

- Si les variables sont indépendantes, la probabilité de chaque paire est donnée par le produit des probabilités de leurs éléments, et on trouve tout naturellement que la densité de probabilité de Z est le produit de convolution des densités de probabilité initiales :

$$f_Z(z) = \int_{-\infty}^{\infty} f_{X_1}(x) f_{X_2}(z - x)dx = f_{X_1}(x) * f_{X_2}(x)$$

Rappel : Variables aléatoires discrètes

Une v. a. discrète obéit à une fonction de probabilité discrète $f_X(x(t_i)) = P(X = x(t_i) = a)$ exemples : ***loi de Bernoulli***, ***loi binomiale***, ***loi Poisson***).

Processus stochastiques

Les processus stochastiques (ou aléatoires) *X* ne peuvent pas être décrits par des relations mathématiques exactes. Le résultat d'une expérience aléatoire est intrinsèquement imprédictible. Un processus aléatoire peut être décrit uniquement par certaines propriétés statistiques.

Exemple d'un processus stochastique :
On note *X(n)* le *temps d'attente* avant l'arrivée de la $n^{\text{ième}}$ événement.
C'est la ***loi exponentielle***:

$$P(a \le X(n) \le b) = \lambda \int_a^b e^{-\lambda x} dx \text{ pour } 0 \le a \le b$$

En réalité, ni des signaux purement déterministes ni des signaux purement stochastiques se produisent en pratique. Ainsi, la plupart des signaux sont un mélange des deux avec une prédominance plus ou moins prononcée d'un composant déterministe ou stochastique.

Beaucoup de biosignaux contiennent une composante importante aléatoire, en particulier ceux qui surviennent en raison de la moyennage de nombreuses sources (EEG, EMG, …).

D’autres biosignaux comme l’ECG contiennent une composante déterministe assez prononcée liées à la propagation de l'activité électrique dans les structures du cœur, bien que certaines composantes aléatoires provenant de bruits biologiques est également présent.

Classification des processus stochastique

Soit $X(temps, x(temps))$ un processus stochastique

	v.a. continu ($x_c \in \mathbb{R}^K$)	**v.a. discret ($x_d \in \mathbb{N}^K$)**
Temps continu	Processus continu à temps continu $X(t, x_c(t))$	Processus continu à temps discret $X(t, x_d(t))$
Temps discret	Processus continu à temps discret $X(n, x_c(n))$	Processus discret à temps discret $X(n, x_d(n))$

x peut être monodimensionnelle (k =1) ou multidimensionnelle (k >1) .

27/12/1654 – 16/08/1705
Jacques Bernoulli
Physiscien suisse
http://fr.wikipedia.org/wiki/Jacques_Bernoulli

- Fonction de probabilité discrète $f_{X_n}(x_n(t_i)) = P(X_n = x_n(t_i) = \text{a})$, ***loi de Bernoulli***, ***loi binomiale*** , ***loi Poisson***).

Exemples de processus stochastiques à variable aléatoire discrète :

• **loi de Bernoulli *B*(*n*, *p*)**

Une distribution discrète de probabilité, qui prend la valeur 1 avec la probabilité *P* et 0 avec la probabilité $q = 1 - P$:

$$P(X = x) = \begin{cases} p & si\ x = 1 \\ 1 - p & si\ x = 0 \\ 0 & \text{sin}\, on \end{cases}$$

ou

$$P(X = x) = p^x (1-p)^{1-x} 1_{\{0,1\}}(x)$$

Avec $E[x] = p$, $V[x] = p(1\text{-}p)$

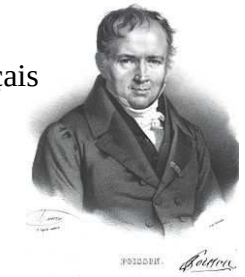
Lois liées

• **loi de binomiale** : Soit $x_1, x_2, \ldots, x_n$ une matrice de variables aléatoires de Bernoulli indépendantes et identiquement distribuées, alors leur somme *s* suit la loi binomiale :

$$s = \sum_{k=1}^{N} x_k \sim B(n, p)$$

Exemples de processus stochastiques à variable aléatoire discrète :

21/06/1781 – 25/04/1840
Siméon Denis Poisson, mathématicien, géomètre et physicien français
http://fr.wikipedia.org/wiki/Sim%C3%A9on_Denis_Poisson



Lois liées (suite)

• **loi de Poisson** : Soit $x_1, x_2, \ldots, x_m$ une suite de variables aléatoires de Bernoulli indépendantes, avec paramètres respectifs p_k, on note alors leur somme s suit la loi de Poisson :

$$s = \sum_{k=1}^{N} x_k \sim B(n, p)$$

$$s = \sum_{k=1}^{m} x_k \quad et \quad \lambda = E[s] = \sum_{k=1}^{m} p_k$$

$$P(X(n) = k) = \frac{\lambda^k}{k!} e^{-\lambda}$$

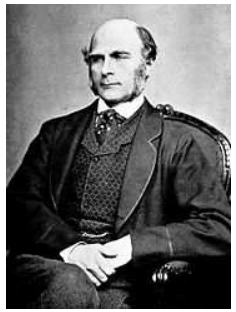
Exemple : On note *X(n)* le nombre de requêtes ou des événements remarquables par unité de temps.

- **loi normale (gaussienne)**

D'après le théorème de la limite centrale qui démontre que la somme y de N variables aléatoires indépendantes, de même moyenne *m* et de même variance σ^2, possède une loi de probabilité gaussienne $\mathcal{N}(Nm_X,\sigma_X^2/N)$ quand $N \rightarrow \infty$. Ceci explique en soi l'importance, en pratique, de cette variable aléatoire.

Ce qui revient à dire, que la convolution d'un grand nombre de fonctions de densités de probabilités tend toujours vers une fonction gaussienne.

16/02/1822 – 17/01/1911
Francis Galton, homme de science britannique
http://fr.wikipedia.org/wiki/Francis_Galton



Démonstration simple de la convergence de la loi binomial vers la loi normale (théorème de De Moivre-Laplace)

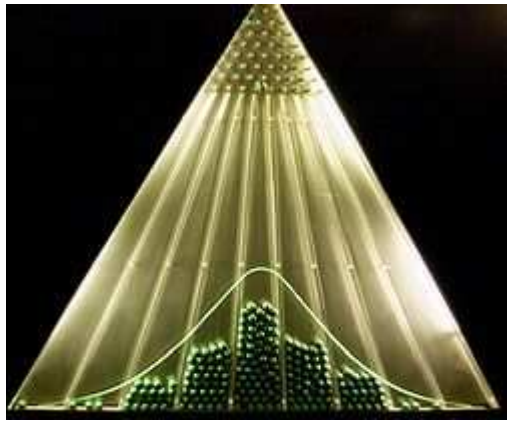
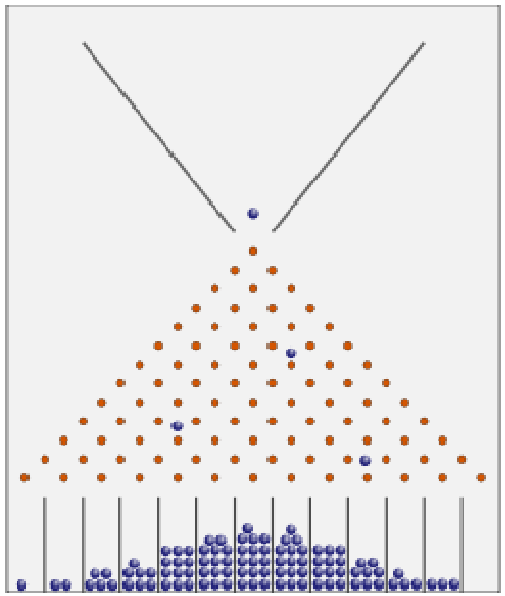


Planche de Galton : http://fr.wikipedia.org/wiki/Planche_de_Galton

Des clous sont plantés sur la partie supérieure de la planche, de telle sorte qu'une bille lâchée sur la planche passe soit à droite soit à gauche pour chaque rangée de clous. Dans la partie inferieure les billes sont rassemblées en fonction du nombre de passages à gauche et de passage à droite qu'elles ont fait.



Ainsi chaque case correspond à un résultat possible d'une expérience binomiale (en tant qu'une expérience de Bernoulli répétée) et on peut remarquer que la répartition des billes dans les cases approche la forme d'une courbe de Gauss, autrement dit : la loi binomiale converge vers la loi normale.

À une réalisation *r* particulière $x_r(t_i)$ de *X* on associera la fonction

$$f_{X_r}(x_r(t_i)) \text{ , } 1 \leq r \leq \text{R et i} \in \mathbb{N}$$

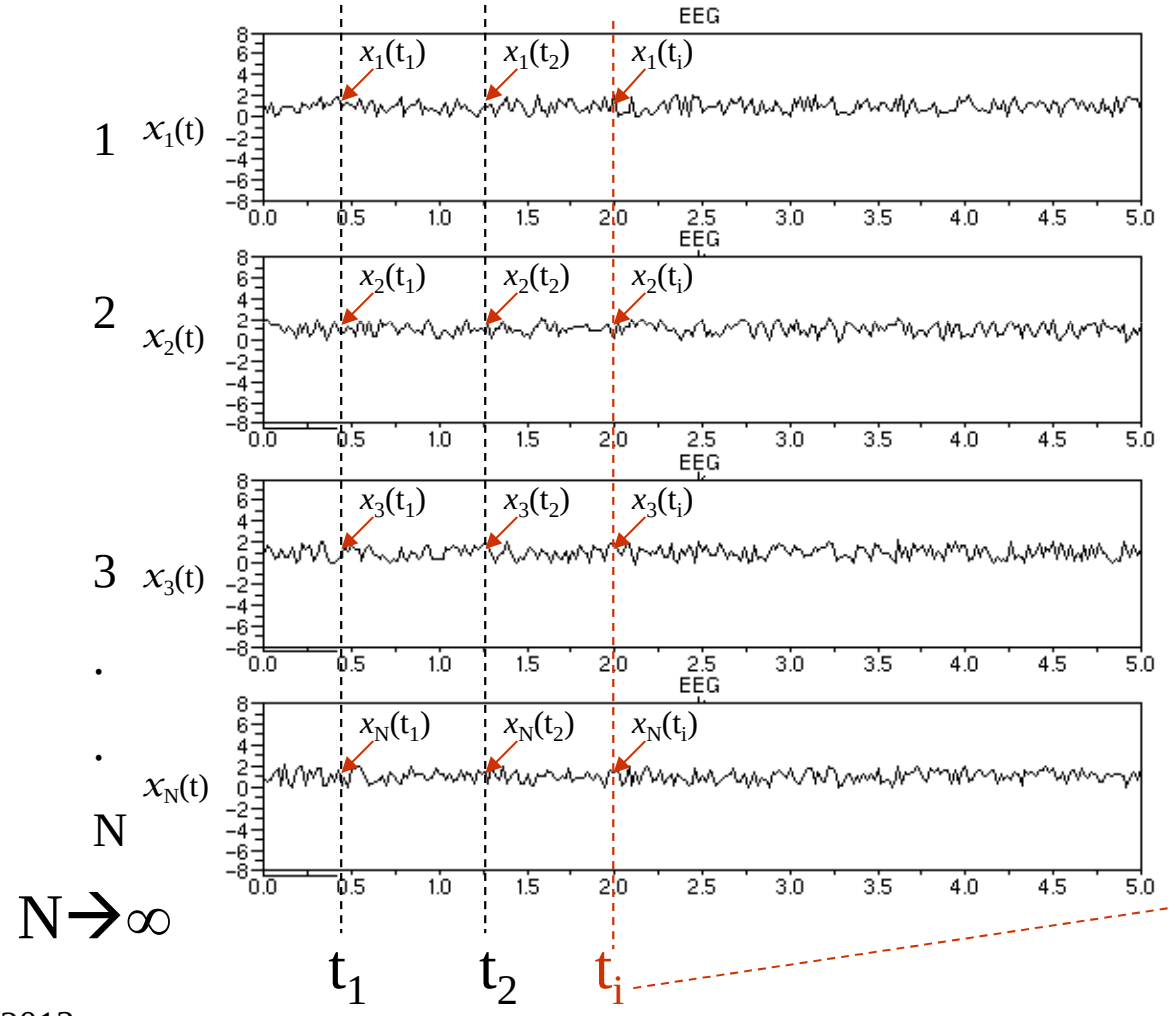
considérée comme une fonction déterministe continue (***fonction de densité de probabilité*** $f_{x_r}(x_r(t_i)) = P(a \leq x_r \leq b)$), par exemple la loi normale, loi uniforme.

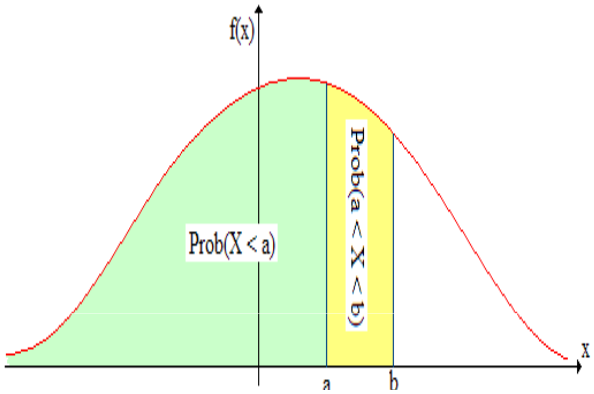
Exemple: *X* est l'activité cérébrale mesurée sur le scalpe mesurée par un électrode de surface. $x_r(t_i)$ est une réalisation numéro *r* d'une variable aléatoire continue *X* de l'EEG mesuré au temps t_i ($t_i \in \mathbb{R}$, $i \in \mathbb{N}$), où $\mathbb{N}$ est le nombre des échantillons, à partir d'une réalisation (enregistrement) définie sur une période *T*.

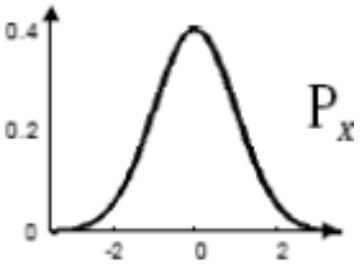
Ensemble d'un grand nombre N réalisations (ou mesures) d'une variable aléatoire continue scalaire X au temps t_i ($0 \leq t_i \leq 5$ sec, $i \in N$), N est le nombre total d'échantillons.

$$f_X(x(t_i)) = \frac{1}{(2\pi)^{1/2}\,\sigma_x^2} \exp\left(-\frac{1}{2\sigma_x^2}(x(t_i) - \mu_x)^2\right),$$

$$N \to \infty \text{ et } 1 \leq i \leq T$$



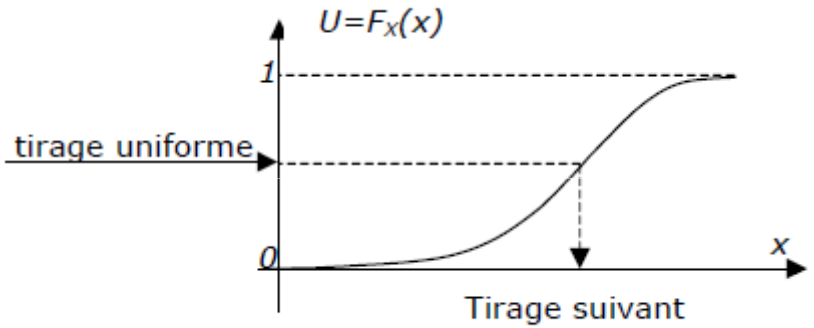




à $t_i = 2$:

$f_X(x(2))$

On montre facilement qu'il est possible d'obtenir le tirage d'une v.a. X à densité de probabilité quelconque $f_X(x)$ à partir d'un tirage de v.a. U à *densité uniforme*, en associant à chaque réalisation u de U la valeur $x=F_{X-1}(u)$ où F est la *fonction de répartition* de X.



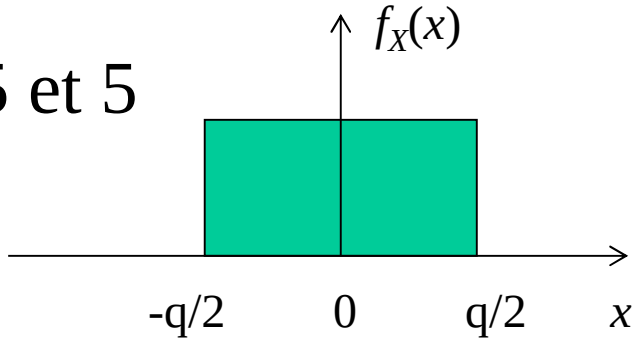
$f_X(x)$

Exercice - Variable aléatoire à distribution uniforme sur (*-q/2, q/2*)

Cette v.a. est caractérisée par une densité de probabilité constante

x=(x-1/2)*10; %v.a. uniforme entre –5 et 5

Moyenne = $m_X = 0$

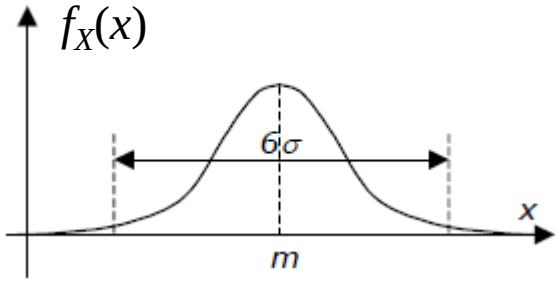


$$\sigma_X^2 = E[X^2] = \int_{-q/2}^{+q/2} \frac{x^2}{q}\, dx = \frac{q^2}{12}$$

Exercice : Sous Matlab, générer 1000 échantillons et estimer sa moyenne et sa variance, et afficher un histogramme donnant une idée de sa densité de probabilité. Conclure

Exemple - Variable aléatoire à distribution normale

Cette v.a. est caractérisée par une densité de probabilité constante



Moyenne = m_X

Variance = σ_X^2

X= x . σ_X+m_X

On montre facilement que plus de 99% des valeurs d'une v.a. gaussienne tombent dans l'intervalle (m-3σ, m+3σ).

Exercice : Sous Matlab, générer 1000 échantillons avec une *moyenne* = 8 et *variance* = 100, ensuite estimer sa moyenne et sa variance, et afficher un histogramme donnant une idée de sa densité de probabilité. Conclure

Bruits

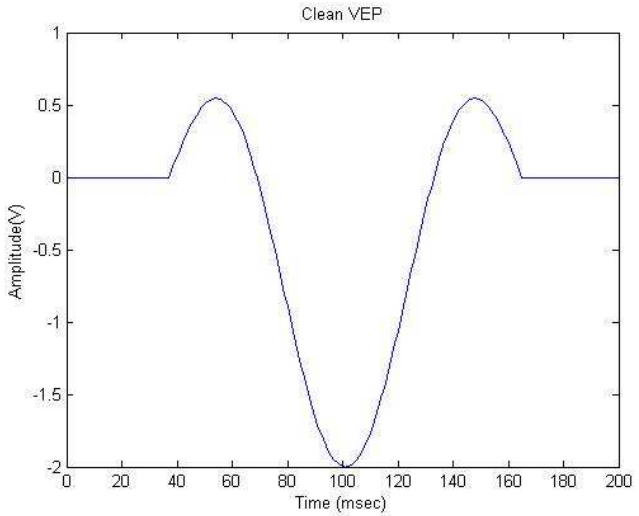
Les informations issues d'un processus biologique sont toujours perturbées par des **bruits**. Ces sources de bruits peuvent être internes ou externes au processus physiques :

Les bruits internes sont **inhérentes** aux processus : cas de fluctuation de niveau d'oxygène dans le sang ou de la température du corps. Ces bruits sont qualifiés d'internes et il n'existe pas de moyens physiques pour les éliminer.

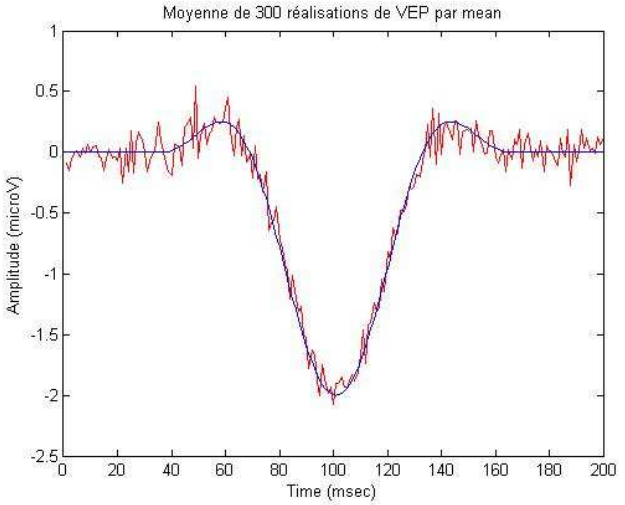
Les bruits externes proviennent de plusieurs sources environnementales. Il est toujours possible de réduire leurs influences dans le domaines fréquentiels ou temporels à l'aide de techniques appropriées de traitement du signal.

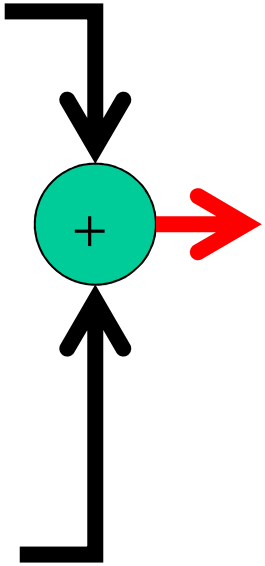
Les principaux bruits sont le **bruits blanc** (par analogie avec la lumière blanche) et le **bruit colorée**.

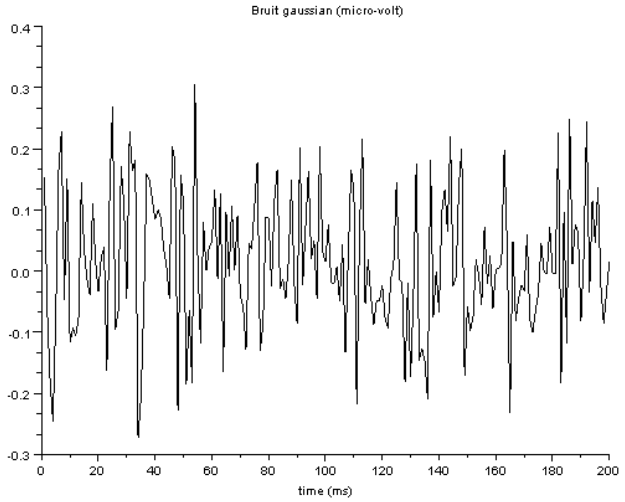
Bruit sur un signal utile



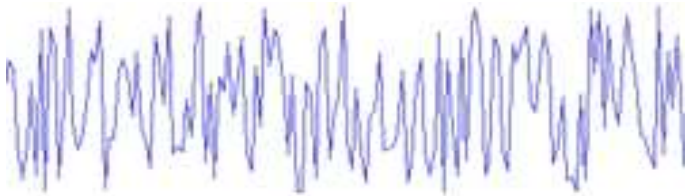
Filtrage







Bruit blanc



Si un processus stochastique est représenté par des variables aléatoires $X(t_i)$ indépendantes et suivent la *loi normale centrée réduite* (loi de Gauss de moyenne nulle et d'écart type unitaire), le processus X s'appelle dans ce cas un ***bruit blanc***. Dans ce cas, les variable, $X(t_1)$ et $X(t_2)$, $t_1 \# t_2$ ne sont pas corrélées.

Le théorème « limite central » permet d'utiliser, en général, le bruit blanc pour modéliser un bruit physique :

Si X_n , $n = 1, 2, \ldots, N$, est une suite de variables aléatoires scalaires indépendantes et de même loi, de moyenne μ_X et de variance σ_X^2, alors la loi de

$$s_N = \sum_{n=1}^{N} X_n$$

tend vers la loi normale de moyenne $N.\mu_X$ et de variance $N.\sigma_X 2$, ou

$$\frac{s_N - N\mu_X}{\sigma_X \sqrt{N}} \underset{N\to\infty}{\rightarrow} N(0,1)$$

Théoriquement, un signal aléatoire est une réalisation d'un processus aléatoire. Il est le résultat d'une expérience obtenue avec un système expérimental permettant de réaliser un grand nombre de réalisations (par exemple : enregistrements par un système d'acquisition EEG). A partir de ce nombre de réalisations, on peut calculer des statistiques de l'ensemble.

La relation entre deux échantillons aux temps k et p est exprimée par la densité de probabilité conjointe $f_{X_k,X_p}(x_k,x_p) = f(x_k,x_p|t_k,t_p)$

Rappelons que:

$$f_{X_k,X_p}(\mathrm{u} \le x_k \le u+\Delta u, v \le x_p \le v+\Delta v) \approx f_{x_k,x_p}(u,v)\Delta u \Delta v$$

Donc la connaissance, par exemple de la valeur prise par x_k peut fournir une information sur celle de x_p si :

$$f_{X_k,X_p}(u,v) \neq f_{X_k}(u)\, f_{X_p}(v)$$

c'est à dire si x_k et x_p ne sont pas indépendants.

En pratique il est difficile de travailler avec les densités conjointes. On se limite à des statistiques plus simples à calculer.

Théoriquement, la description d'un processus aléatoire continu nécessite la connaissance de cette statistique pour un ensemble infini de valeurs de *t*. Pour des considérations pratiques : **statistique d'ordre 1 et d'ordre 2** uniquement.

- **Statistique d'ordre 1**

La moyenne statistique $\mu_x(t_i)$

$$\mu_x(t_i) = E[X(t_i)] = \int_{-\infty}^{+\infty} x\, f(x|t_i)\, dx,$$

En pratique, il faut effectuer *N* réalisation pour calculer la **moyenne statistique estimée** et la **variance statistique estimée** d'une variable aléatoire:

$$m_X(t_i) = \lim_{N\to+\infty} \frac{1}{N} \sum_{n=1}^{N} x_n(t_i)$$

$$V_X(t_i) = \lim_{N\to+\infty} \frac{1}{N-1} \sum_{n=1}^{N} (x_n(t_i) - m_X(t_i))^2$$

- **Statistique d'ordre supérieur**

Une v.a. est déterminée statistiquement si l'on connaît les lois de probabilité de toutes les variables aléatoires qu'elle engendre. Nous considérons k instants différents t_1, t_2,…, t_K auxquels correspondent les variables aléatoires X_1, X_2,…, X_K provenant de même processus X.
La connaissance de la loi de densité de probabilité pour toutes les valeurs de temps t_i et pour toutes les valeurs de k possibles conduit à la connaissance de la loi temporelle de la variable aléatoire $x(t)$.

- La **loi de densité de probabilité conditionnelle conjointe** pour t_1, t_2, t_3, .., t_k est $f_{Xk}\,(x(t_1)\,,\,x(t_2)\,,\,x(t_3)\,,\,..,\,x(t_k)\,|\,t_1,\,t_2,\,t_3,\,\ldots,\,t_k)$.

La connaissance de cette loi définit la **statistique d'ordre k** du processus aléatoire.

- **Stationnarité**

Un processus stochastique est **stationnaire au sens strict** si toutes ses propriétés statistiques sont invariantes par translation dans le temps :

$f_X(x(t_1) , x(t_2) , \ldots, x(t_k) \mid t_1, t_2, \ldots, t_k)$

$= f_X (x(t_1+ \tau) , x(t_2 + \tau) , \ldots, x(t_k + \tau) \mid t_1+ \tau, t_2 + \tau, \ldots, t_k + \tau)$

alors cette loi de densité de probabilité ne dépend que de t.

- Un signal est **stationnaire au sens strict** si la densité conjointe est invariante pour toute valeur t de translation temporelle t_i, $t_i + t$, $i = 1, 2, ..$

Un processus aléatoire est stationnaire à l'ordre 2 si ses lois de densités de probabilité $f_X(x| t)$ et $f_X(x(t), x(t+\tau))$ sont invariants dans le temps, d'où :

$m_X(t_1) = m_X(t_2) = \ldots = m_X(t_k) = m_X$

$\sigma_X^2(t_1) = \sigma_X^2(t_2) = \ldots = \sigma_X^2(t_k) = \sigma_X^2$

Un signal est **stationnaire au sens strict** si la densité conjointe est invariante pour toute valeur t de translation temporelle t_i, $t_i + t$, i=1, 2, 3, ..

❑ Statistique d'ordre 2

Les termes **corrélation** et ***covariance*** statistiques permettent de mesurer le degré de similarité entre :

- 2 ou plusieurs variables aléatoires dans une même séquence ou
- 2 ou plusieurs segments dans une même séquence ou
- 2 ou plusieurs séquences de processus aléatoires.

La comparaison entre plusieurs séquences peut produire une ***matrice de corrélation*** ou ***une matrice de covariance***.

Pour une réalisation donnée n, nous considérons deux instants différents t et $t+\tau$, où τ est un retard donné) auxquels correspondent les variables aléatoires $x(t_1)$ et $x(t_2)$ provenant de même processus X, alors la **fonction d'autocorrélation statistique** du processus aléatoire $\sigma_{XX}(\tau)$

$$\sigma_{XX}(\tau) = E[x(t)\,x(t+\tau)] = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} x(t)\,x(t+\tau)\,f(x(t)\,x(t+\tau)|\tau)\,dx(t)\,dx(t+\tau)$$

Le terme **autocorrélation** permet de mesurer le degré de similarité entre deux variables de même processus aléatoire à deux instants différents du temps t et $t+\tau$. La **fonction d'autocorrélation** du signal peut être extraite de la densité conjointe de 2 échantillons et elle fournit une information assez précise sur la dépendance entre deux échantillons aux temps t et $t+\tau$.

- Pour un signal stochastique

$$\sigma_{XX}(\tau) = \lim_{T\to\infty} \frac{1}{T} \int_{-\frac{T}{2}}^{+\frac{T}{2}} x(t)x(t+\tau)dt$$

Où T est la largueur de la fenêtre d'observation.

σ est toujours à valeurs réelles et est une fonction paire, à valeur maximale à τ = 0.

- Pour un signal échantillonné

$$\sigma_{XX}(\tau) = \frac{1}{N} \sum_{n=1}^{N-\tau+1} x(n)x(n+\tau-1)$$

pour τ = 1, 2, .., N+1, où N est le nombre d'échantillons.

- Pour un signal déterministe à durée infinie

$$\sigma_{XX}(\tau) = \frac{1}{T}\int_{-\frac{T}{2}}^{+\frac{T}{2}} x(t)x(t+\tau)dt$$

- Pour un signal déterministe à durée finie si le signal existe uniquement dans l'intervalle $[t_1 \;\; t_2]$

$$\sigma_{XX}(\tau) = \int_{t_1}^{t_2} x(t)x(t+\tau)dt$$

- Pour un signal déterministe périodique avec une période T

$$\sigma_{XX}(\tau) = \int_{t_0}^{t_0+T} x(t)x(t+\tau)dt$$

Propriétés de la fonction d'autocorrélation

- Soit X un processus stochastique réel

$$\sigma_{XX}(\tau) = \sigma_{XY}(-\tau)$$

Pour $\tau = 0$, $\sigma_{XX}(0) = E[x(t)^2] \geq 0$

Si X est centré (i.e. $m_X = 0$), $\sigma_{XX}(0) = \sigma_x^2$

Cette propriété permet d'accéder graphiquement à la valeur de **variance** σ_x^2. Pour toutes valeurs de τ :

$$\left|\sigma_{XX}(\tau)\right| \leq \sigma_{XX}(0)$$

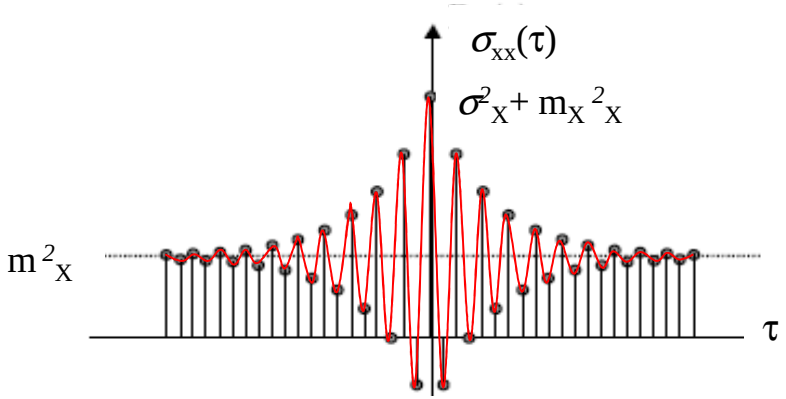
Si x n'est pas centré ($m_X \neq 0$), $\sigma_{XX}(0) = \sigma_x^2 + m_x^2$

Propriétés de la fonction d'autocorrélation (suite)

- En général, quand $\tau \rightarrow +\infty$, on observe que pour des processus aléatoires non périodiques, la fonction d'autocorrélation tend vers une valeur constante, mais la règle n'est pas absolue

$$\lim_{\tau \to +\infty} \sigma_{XX}(\tau) = E[x(t)]\,E[x(t+\tau)] = m_x^2$$

L'intérêt de cette propriété est de fournir directement une estimation du carré de la moyenne d'une processus aléatoire par lecture graphique. La fonction a souvent une vague allure de sinusoïde amortie.



- Si le signal est transitoire ou périodique, la fonction d'autocorrélation est nulle.

Fonction de corrélation croisée statistique (ou intercorrélation)

La fonction d'autocrrélation croisée entre $x(t)$ et $y(t+\tau)$:

$$\sigma_{XY}(\tau) = \frac{1}{T}\int_0^T x(t)y(t)dt$$

ou en forme discrète

$$\sigma_{XY}(\tau) = \frac{1}{N}\sum_{k=1}^{N} x(k)y(k+\tau)$$

$$\sigma_{XY}(\tau) = E[x(t)y(t+\tau)]$$

$$\sigma_{YX}(\tau) = E[y(t)x(t+\tau)]$$

$$\sigma_{XY}(\tau) = \sigma_{YX}(-\tau)$$

Fonction d'autocovariance statistique (ou covariance)

$$\rho_{XX}(\tau) = E\left[(x(t) - \mu_x(t))(x(t+\tau) - \mu_x(t+\tau))\right] = \sigma_{XX}(\tau) - \mu_X(t)\mu_X(t+\tau)$$

Si X est un processus stationnaire, alors $\rho_{XX}(\tau) = \sigma_{XX}(\tau) - \mu_X^2$

La covariance d'une variable avec elle-même est appelée ***variance***.

Fonction de covariance croisée statistique

$$\rho_{XY}(\tau) = E\left[(x(t) - \mu_x(t))(y(t+\tau) - \mu_y(t+\tau))\right] = \rho_{XY}(\tau) - \mu_x(t)\mu_y(t+\tau)$$

La corrélation est un nombre pur mais la covariance a pour unité le produit des unités des variables. Le coefficient de corrélation est en effet une notion similaire à la covariance qui permet d'éviter un problème lié à la définition de la covariance : celle-ci change selon l'unité choisie pour les valeurs des variables.

- Pour un signal stochastique

$$\rho_{XY}(\tau) = \lim_{T\to\infty} \frac{1}{T} \int_{-\frac{T}{2}}^{+\frac{T}{2}} x(t)y(t+\tau)dt$$

$$\rho_{YX}(\tau) = \lim_{T\to\infty} \frac{1}{T} \int_{-\frac{T}{2}}^{+\frac{T}{2}} y(t)x(t+\tau)dt$$

Où T est la largueur de la fenêtre d'observation.

- Pour un signal échantillonné

$$\rho_{XY}(\tau) = \frac{1}{N} \sum_{n=1}^{N-\tau+1} x(n)y(n+\tau-1)$$

$$\rho_{YX}(\tau) = \frac{1}{N} \sum_{n=1}^{N-\tau+1} y(n)x(n+\tau-1)$$

pour $\tau = 1, 2, .., N+1$, où N est le nombre d'échantillons.

- Pour un signal à durée infinie

$$\rho_{XY}(\tau) = \frac{1}{T}\int_{-\frac{T}{2}}^{+\frac{T}{2}} x(t)y(t+\tau)dt$$

- Pour un signal à durée finie, si le signal existe uniquement dans l'intervalle $[t_1 \;\; t_2]$

$$\rho_{XY}(\tau) = \int_{t_1}^{t_2} x(t)y(t+\tau)dt$$

- Pour un signal périodique avec une période T

$$\rho_{XX}(\tau) = \int_{t_0}^{t_0+T} x(t)y(t+\tau)dt$$

Pour un processus stationnaire :
Une fonction de corrélation est stationnaire au sens large si sa moyenne est indépendante de l'origine des temps et si sa fonction d'autocorrélation ne dépend que de τ.

$\rho_{XX}(t_l, t_k) = \sigma_{XX}(\tau) - \mu_X^2$,
Si $\mu_X = 0$, alors $\rho_{XX}(\tau) = \sigma_{XX}(\tau)$

Propriétés des fonctions de corrélation et de covariance croisées σ_{XY} et ρ_{XY}

$$\sigma_{XY}(\tau) \leq \sqrt{\sigma_{XX}(0)\sigma_{YY}(0)}$$

$$\left|\rho_{XY}(\tau)\right| \leq \sqrt{\rho_{XX}(0)\rho_{YY}(0)} = \sigma_X \sigma_Y$$

Deux processus aléatoires sont **non corrélés** pour les valeurs de τ quand $\rho_{XY}(\tau) = 0$ et ils sont orthogonaux pour les valeurs de τ quand $\sigma_{XY}(\tau) = 0$.

Si l'un des processus aléatoires est à moyenne nulle : $E[x(t)] = 0$ ou $E[y(t)] = 0$, alors la non-corrélation et l'orthogonalité se confondent.

Les deux processus sont statiquement indépendants si :

$\rho_{XY}(\tau) = \rho_{YX}(\tau) = 0 \ \forall\tau$

$\sigma_{XY} = \sigma_{YX} = m_x m_y$

Matrice de corrélation

$$\sigma_{XY}(\tau) = \begin{bmatrix} \sigma_{XX}(\tau) & \sigma_{XY}(\tau) \\ \sigma_{YX}(\tau) & \sigma_{YY}(\tau) \end{bmatrix}$$

Matrice de covariance

$$\rho_{XY}(\tau) = \begin{bmatrix} \rho_{XX}(\tau) & \rho_{XY}(\tau) \\ \rho_{YX}(\tau) & \rho_{YY}(\tau) \end{bmatrix}$$

Ergodicité

La notion d'ergodicité est fondamentale pour le traitement de signaux bio-médicaux car elle permet de considérer un seule enregistrement temporelle d'un processus aléatoire au lieu de considérer un ensemble enregistrements pour obtenir des statistiques à partir des moyennes obtenues au cours du temps.

$$m_X = \frac{1}{\mathrm{T}} \sum_{t=1}^{T} x_t, \; t \in \mathrm{IN}$$

$$V_X = \frac{1}{\mathrm{T\text{-}1}} \sum_{t=1}^{T} (x_t - m_X)^2$$

- **Ergodicité** (suite)

L’ergodicité permet ainsi d’obtenir des statistiques à partir des moyennes obtenues au cours du temps. Lors de l'application des techniques de traitement du signal, la stationnarité et l'ergodicité des signaux sont fréquemment (et implicitement) supposé, et de nombreuses techniques peuvent être utiles même si ces hypothèses ne sont pas strictement respectées.

Un processus stochastique $X(t, x)$ intéressons-nous à une réalisation particulière temporelle i : $X_i(t, x)$.

La ***moyenne temporelle*** m_X du processus $X(t, x)$ est définie sur un domaine finie [$-T/2$ $T/2$] par :

$$m_X = \lim_{T\to+\infty} \frac{1}{T} \int_{-\frac{T}{2}}^{+\frac{T}{2}} x(t)dt,$$

Alors, la fonction d'***autocorrélation temporelle*** $\sigma_{XX}(\tau)$:

$$\sigma_{XX}(\tau) = \lim_{T\to+\infty} \frac{1}{T} \int_{-\frac{T}{2}}^{+\frac{T}{2}} x(t)x(t+\tau)dt,$$

Les limites sur T peuvent, dans certains cas, ne pas exister. Intuitivement, ce limite existent pour des processus stationnaires, et prennent des valeurs indépendantes de la sélection de la réalisation $X_i(t, x)$.

Un processus stochastique $X(t, x)$ possède la propriété d'ergodicité si m_X et σ_{XX} temporelles correspondent exactement à m_X et σ_{XX} statistiques .

Les notions d'ergodicité et de stationnarité sont indépendantes et donc l'ergodicité n'entraîne pas la stationnarité et réciproquement.

Exercice : Autocorrélation d'un signal sinusoïdal

Créer un script Matlab pour tracer la séquence d'autocorrelation d'un signal sinusoidal *x* d'une fréquence de 1 Hz, fréquence d'échantillonnage de 200 Hz. Utiliser la fonction *xcorr*(*x*) de Matlab pour créer ce script : sigma_xx = xcorr(x)

Exercice : Autocorrélation d'un signal sinusoïdal

Créer un script Matlab pour tracer la séquence d'autocorrelation d'un signal sinusoidal *x* d'une fréquence de 1 Hz, fréquence d'échantillonnage de 200 Hz. Créer une fonction Matlab
sigma_xx = autocorr(*x*)
en utilisant la formule suivante pour la créer et l'appeler par le script.

$$\sigma_{XX}(\tau) = \frac{1}{N} \sum_{n=1}^{N-\tau+1} x(n)x(n+\tau-1)$$

pour $\tau = 1, 2, .., N+1$, où N est le nombre d'échantillons.

Exercice : Corrélation croisée d'un signal sinusoïdal avec le même signal mais bruité par un bruit Gaussian de moyenne nulle et variance unité.

Créer un script Matlab pour tracer la séquence de ocorrélation croisée d'un signal sinusoidal *x* d'une fréquence de 1 Hz, fréquence d'échantillonnage de 200 Hz, le même signal mais bruité par un bruit Gaussian de moyenne nulle et variance unité.

l. Utiliser la fonction *xcorr*(*x*,*y*) de Matlab pour créer ce script : sigma_xx = *xcorr*(*x*)

Exercice : Corrélation croisée d'un signal sinusoïdal avec le même signal mais bruité par un bruit Gaussian de moyenne nulle et variance unité.

Créer un script Matlab pour tracer la séquence d'autocorrelation d'un signal sinusoidal *x* d'une fréquence de 1 Hz, fréquence d'échantillonnage de 200 Hz. Créer une fonction Matlab

sigma_xx = *xcorr*(*x*,*y*)

Ecrire une fonction Matlab qui utilise la formule suivante et l'appeler par le script.

$$\sigma_{XY}(\tau) = \frac{1}{N} \sum_{n=1}^{N-\tau+1} x(n)\, y(n+\tau-1)$$

pour $\tau = 1, 2, .., N+1$, où N est le nombre d'échantillons.

Coefficient de corrélation de Pearson entre deux ensemble d'échantillons de variables aléatoires

La fonction covariance croisée entre deux variables aléatoires X et Y est mesurée par

$$\rho_{xy} = \mathrm{E}[(X-m_X)(Y-m_Y)] = \frac{1}{N-1}\sum_{k=1}^{N}(x_k-m_X)(y_k-m_Y)$$

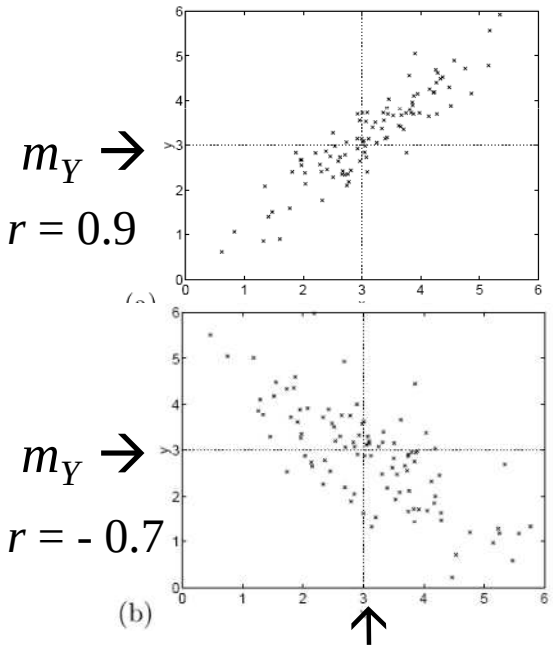
Où m_X et m_Y sont les moyens de deux variables. ρ_{xy} dépend des échelles de X ou Y. Si X a tendance d'être au-dessus de sa moyenne quand Y est au-dessus de sa moyenne alors ρ_{xy} sera positif. Si X a tendance d'être au-dessous de sa moyenne quand Y est au-dessus de sa moyenne alors ρ_{xy} sera négatif.

Le ***coefficient de corrélation de Pearson*** est la covariance normalisée (indépendante des échelles de X ou Y).

$$r = \frac{\sigma_{XY}}{\sigma_X \sigma_Y}, \quad -1 \leq r \leq 1$$

(a) Si $r = -1$ → corrélation négative parfaite
(b) Si $r = 1$ → corrélation positive parfaite

Remarque : Dans certains signaux biomédicaux, les échelles de x et y sont les mêmes. Dans ce cas la covariance peut être utilisée directement.



Adapté de (Penny., 2000)

Le coefficient de Corrélation croisée (*coefficient de corrélation de Pearson* r_{xy})

Par définition :

$$r_{XY}(\tau) = \frac{\sigma_{XY}}{\sigma_X \sigma_Y} \leq \sqrt{\sigma_{XX}(0)\sigma_{YY}(0)}$$

Le coefficient de corrélation croisée fournit des renseignements très utiles sur le **degré de dépendance linéaire** des deux signaux:

Si $r_{XY}(\tau) = 1$, $\forall\ \tau$, alors il y a dépendance linéaire entre x et y de la forme $y(t) = a\ x(t) + b$, avec a et b sont des coefficients constants.

Régression linéaire

Modéliser par une fonction linéaire la relation entre deux variables x et y : étant donné un ensemble de N échantillons de couples de données $\{x_i, y_i\}$, nous cherchons à estimer y_i à partir de x_i en utilisant un modèle linéaire : $m_{Yi} = a\ x_i + b$

La régression avec une variable d'entrée scalaire (X) est souvent appelée la ***régression linéaire uni-variée*** pour le distinguer de la ***régression linéaire multi-variée*** où nous avons une variable d'entrée vectorielle (**X**).

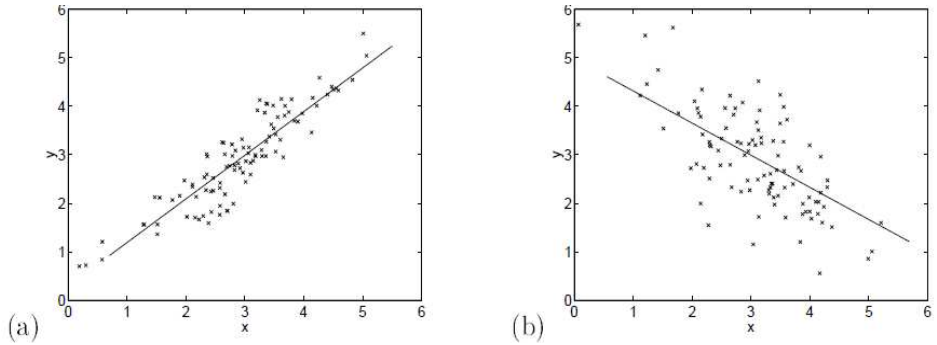
La fidélité du modèle aux données peut être mesurée par la fonction de coût de moindre carré :

$$E = \sum_{k=1}^{N} (y_k - m_{Yk})^2$$

Les valeurs de *a* (la pente de la ligne) et *b* qui réduisent au minimum la fonction E peuvent être calculées en mettant les premières dérivées (par rapport à *a* et *b*) de la fonction de coût au zéro et résolvant les équations simultanées résultantes. La solution est

$$a = \frac{r}{\sigma_X^2}, \quad b = m_Y - a\, m_X$$

On démontre que la droite passe par un point de coordonnées (m_Y , m_X), Cela permet l'adaptation par la moindre carrés d'une ligne de régression à un jeu de données comme indiqué dans la Figure suivante :



La ligne de régression linéaire est adaptée en réduisant au minimum la distance verticale entre elle même et chaque point de données. Les lignes évaluées sont

$$(a)\ \hat{y} = 0.9003\,x + 0.2901, \text{(b)}\ \hat{y} = -0.6629\,x + 4.9804$$

Adapté de (Penny., 2000)

Le modèle s'adaptera à certaines données mieux que d'autres; celles que ce modèle s'adaptent bien constituent le signal et celles que ce modèle ne s'adapte pas bien constituent le bruit. La puissance du bruit est mesurée par la variance du bruit : $\hat{\sigma}_e^2 = \frac{1}{\text{N-1}} \sum_{k=1}^{N} (y_k - m_Y)^2$

et la puissance du signal est donnée par $\hat{\sigma}_Y^2 - \hat{\sigma}_e^2$

Le ***rapport signal sur bruit*** est donnée par

$$SNR = \frac{\hat{\sigma}_Y^2 - \hat{\sigma}_e^2}{\hat{\sigma}_e^2}$$

Nous pouvons démontrer que

$$r = \frac{\sigma_X}{\sigma_Y} a$$

Ainsi le signe de la pente *a* de la ligne de régression définit le signe de la corrélation. Cependant, La corrélation est aussi une fonction des écart-types des variables *x* et *y*; par exemple si *x* est très grand, il est possible d'avoir une corrélation forte bien que la pente puisse être très petite : *r* mesure uniquement la corrélation linéaire. Il est tout à fait possible que deux variables aient une relation fortement non-linéaire (c'est-à-dire non-linéairement corrélées) malgré que *r* =0.

La puissance de la corrélation peut aussi être exprimée en terme de quantités du modèle de régression linéaire

$$r^2 = \frac{\hat{\sigma}_Y^2 - \hat{\sigma}_e^2}{\hat{\sigma}_Y^2}$$

Cette quantité mesure la proportion de la variance décrite par un modèle linéaire, une valeur de $r^2 = 1$ indique que ce modèle décrit parfaitement la relation entre X et Y.

2. Application de corrélation

Introduction

Soit *X* et *Y* deux processus aléatoires,

- Corrélation « *corr*(**X**,**Y**) en Scilab ou Matlab » est proche de la convolution (nous pouvons calculer la corrélation en employant la convolution) : permet de mesurer le degré de la relation entre deux signaux aléatoires (vecteurs) **X** et **Y**.
- Autocorrélation : « *corr*(***X***) » et Corrélation croisée : « corr(**X**, **Y**) » ou « corr(**Y**, **X**) »

Introduction

Avec un calcul et une analyse simples de la fonction d'auto-corrélation, nous pouvons découvrir quelques caractéristiques importantes au sujet d'un processus aléatoire. Celles-ci incluent :

- à quelle rapidité notre signal ou processus aléatoire change en fonction de temps
- si notre processus a une composante périodique et ce qui pourrait être la fréquence estimée.
- localiser les transitoires.
- mesurer les retards et faire l'alignements entre signaux.

Exemple Auto-Corrélation d'un signal EEG de rythme alpha

source : ../ simulation2/sim_eeg_autocorr/eeg_autocorr.sce

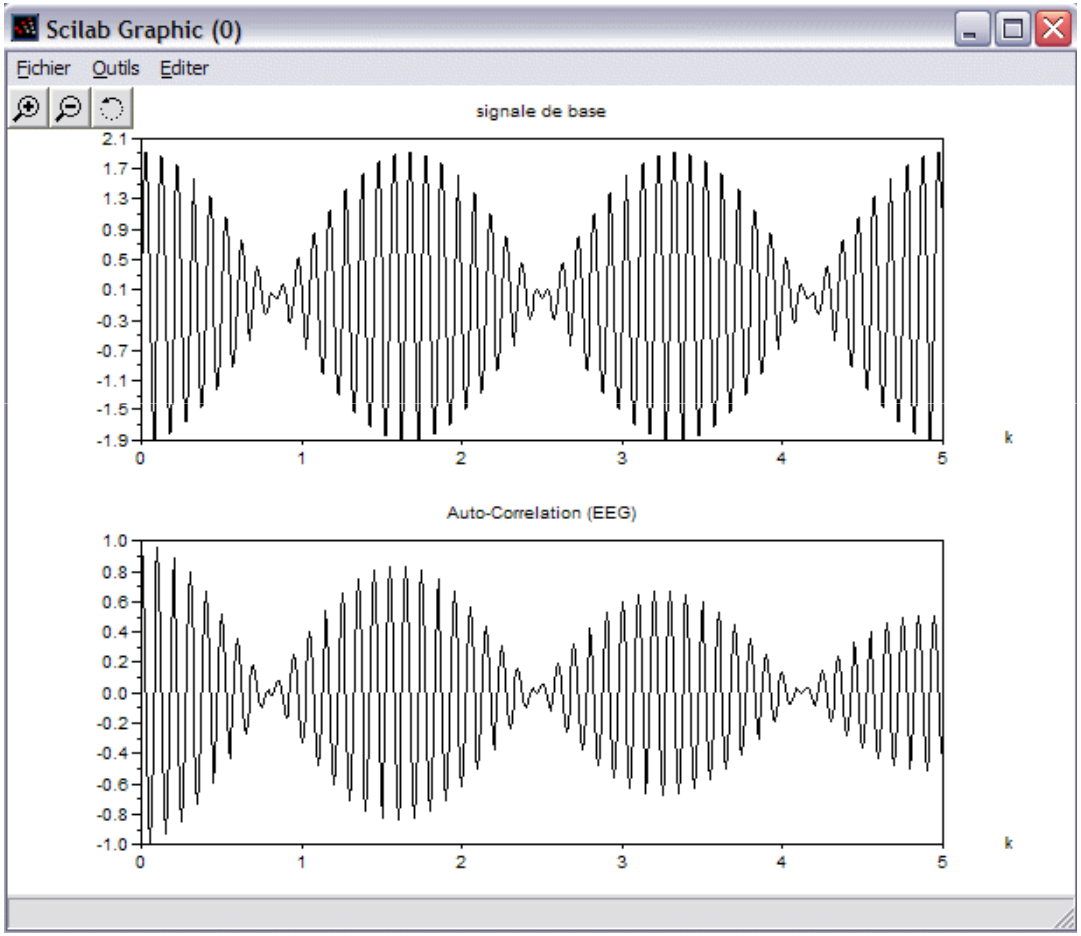
```
// Signal
k=[0:0.01:10]; // 10 sec de simulation. Période d'échantillonnage=100 sec
// signale de base (rythme alpha) : déterministe
signal = sin(2 * %pi * 9.7 * k) + sin(2 * %pi * 10.3 * k);
//Auto-Corrélation de EEG déterministe
autocorr_signal = corr (signal,1000);
xbasc()
xsetech ([0,0,1,1/2]) // 1ère sous-fenêtre
plot2d(k(1:500),signal(1:500)); xtitle('signale de base','k'); // 1ère tracé

xsetech ([0,1/2,1,1/2]) // 2ème sous-fenêtre
plot2d(k(1:500),autocorr_signal(1:500));
xtitle('Auto-Correlation (EEG),'k'); // 2ème tracé
```

Exemple Auto-Corrélation d'un signal EEG de rythme alpha (suite)

À première vue les deux traces semblent très semblables.

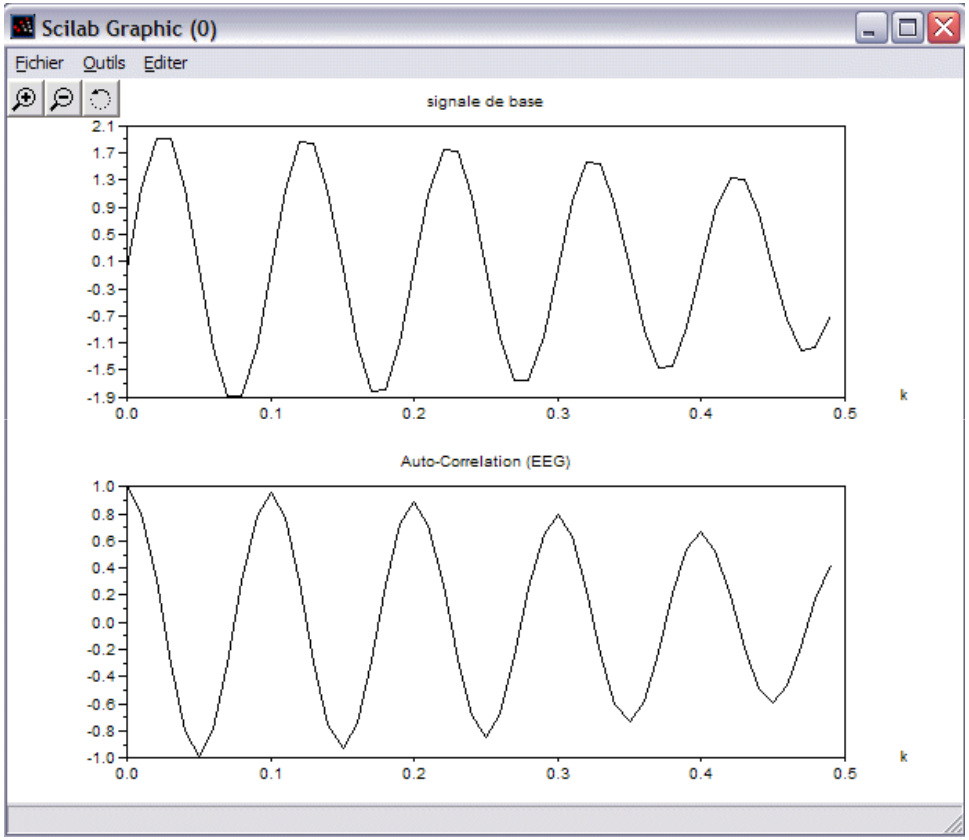
• Le tracé de l'autocorrélation du signal semble **diminuer en l'amplitude**.

• Une autre différence : tandis que **les bornes** du tracé supérieure est de -2 à 2, celles du tracé inférieure (l'autocorrélation) est de -1 à 1.



Exemple a Auto-Corrélation d'un signal EEG de rythme alpha (suite)

Nous pouvons voir l'autocorrélation comme manière de **détecter la périodicité** dans les signaux. Le code précédent a été modifié pour représenter seulement quelques valeurs pour améliorer la résolution : zoom de la figure précédente pour accentuer la périodicité à 10 hertz (période de 0.1 s).



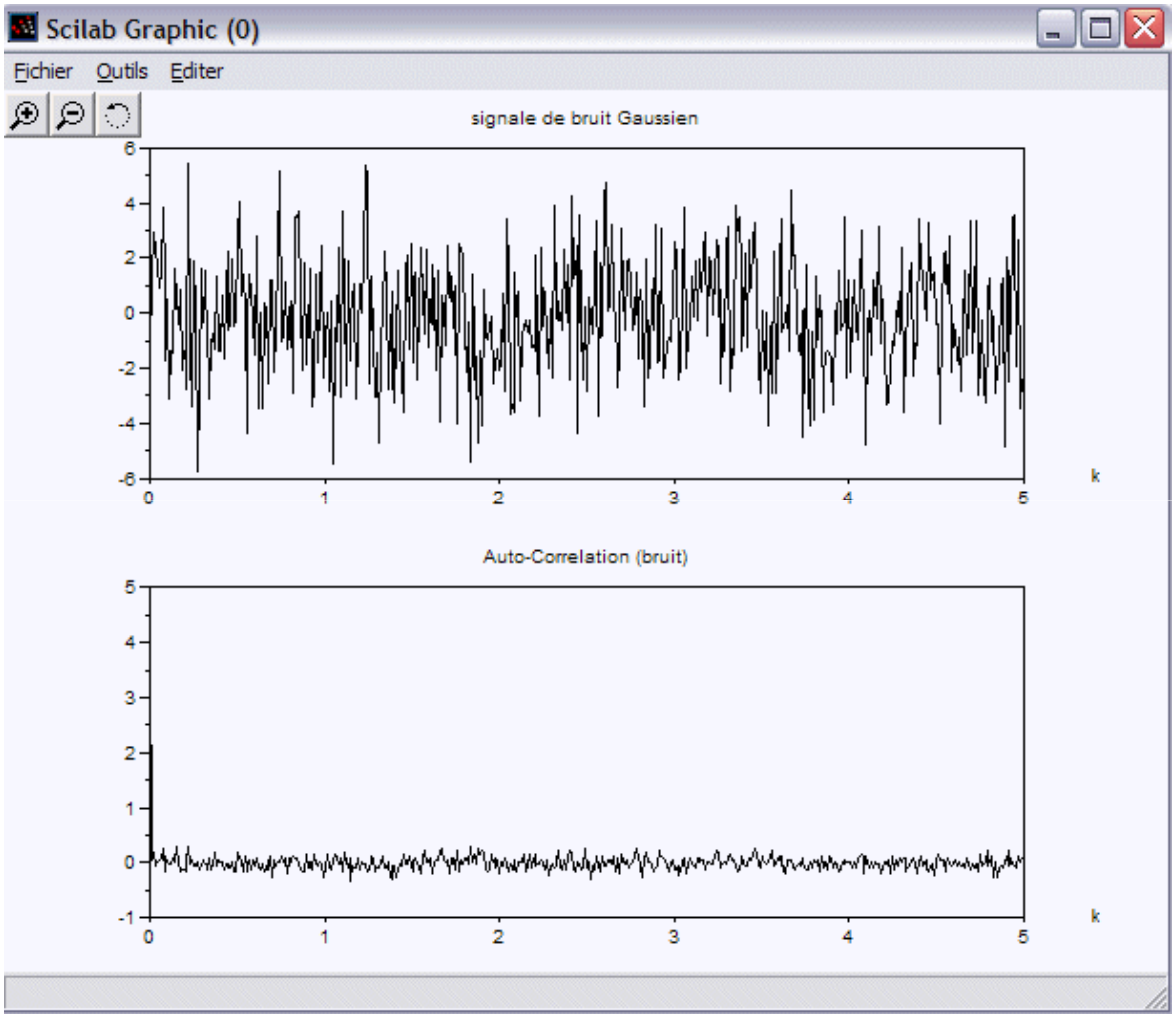
Exemple b Auto-Corrélation d'un signal de bruit

source : ../ simulation2/sim_eeg_autocorr/eeg_autocorr.sce

```
// Signal
k=[0:0.01:10]; // 10 sec de simulation. Période d'échantillonnage=100 sec
signal = sin(2 * %pi * 9.7 * k) + sin(2 * %pi * 10.3 * k); // signale de base (rythme alpha) : déterministe
// Nous utilisons de bruit Gaussien (bruit blanc : contient toutes les fréquences))
// Quand nous employons « rand » précédemment ayant utilisé l'option « normal » nous
// obtenons une distribution des valeurs dont le moyen est « 0 » et dont l'écart type est « 1 ».
// Si nous multiplions par « 2 » nous obtenons une distribution normale (bruit) des valeurs avec une
// moyenne de « 0 » et un écart type de « 2 ».
rand('normal')
rand('seed',0)
bruit = 2 * rand(signal);
//Auto-Corrélation de bruit
autocorr_bruit = corr (bruit,1000); // auto-corrélation
xbasc()
xsetech ([0,0,1,1/2]) // 1ère sous-fenêtre
plot2d(k(1:500),bruit(1:500)); // 1ère tracé
xtitle('signale de bruit Gaussien','k');
xsetech ([0,1/2,1,1/2]) // 2ème sous-fenêtre
plot2d(k(1:500),autocorr_bruit(1:500)); // 2ème tracé
xtitle('Auto-Correlation (bruit) ','k');
```

Exemple b Auto-Corrélation d'un signal de bruit (suite)

L'auto-corrélation du bruit est de moyenne 0 sauf à l'origine où sa valeur représente la variance du signal (plus ou moins 4 puisque l'écart-type = 2).



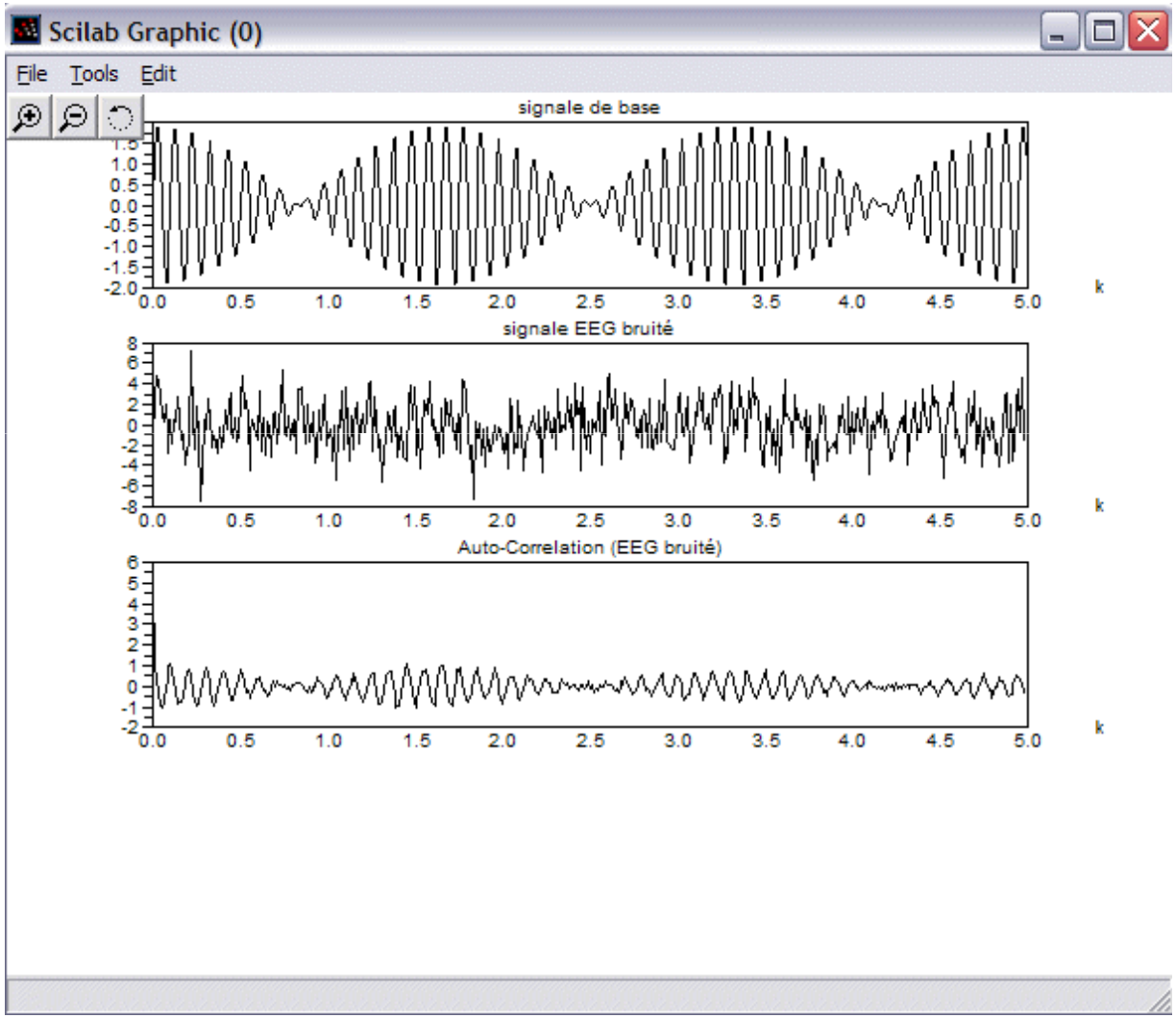
Exemple c Corrélation croisée entre un signal EEG de rythme alpha et un bruit Gaussien

source : ../ simulation2/sim_eeg_autocorr/eeg_autocorr.sce

```
signal_bruit = signal+bruit;
//Auto-Corrélation de EEG avec le bruit
autocorr_signal = corr (signal_bruit,1000); // auto-corrélation
xbasc()
xsetech ([0,0,1,1/4]) // 1ere sous-fenêtre
plot2d(k(1:500),signal(1:500));
xtitle('signale de base','k'); // 1ere tracé
xsetech ([0,1/4,1,1/4]) // 2ème sous-fenêtre
plot2d(k(1:500),signal_bruit(1:500));
xtitle('signale-bruit','k'); // 2ème tracé
xsetech ([0,2/4,1,1/4]) // 3ème sous-fenêtre
plot2d(k(1:500),crosscorr_signal(1:500)); // 2ème tracé
xtitle(‘Auto-Correlation (EEG+Bruit)','k');
```

Exemple c Corrélation croisée entre un signal EEG de rythme alpha et un bruit Gaussien (suite)

La auto-corrélation permet d'extraire des informations concernant le signal d'origine, ici la périodicité du signal.



Mécanisme de la corrélation

Exemple

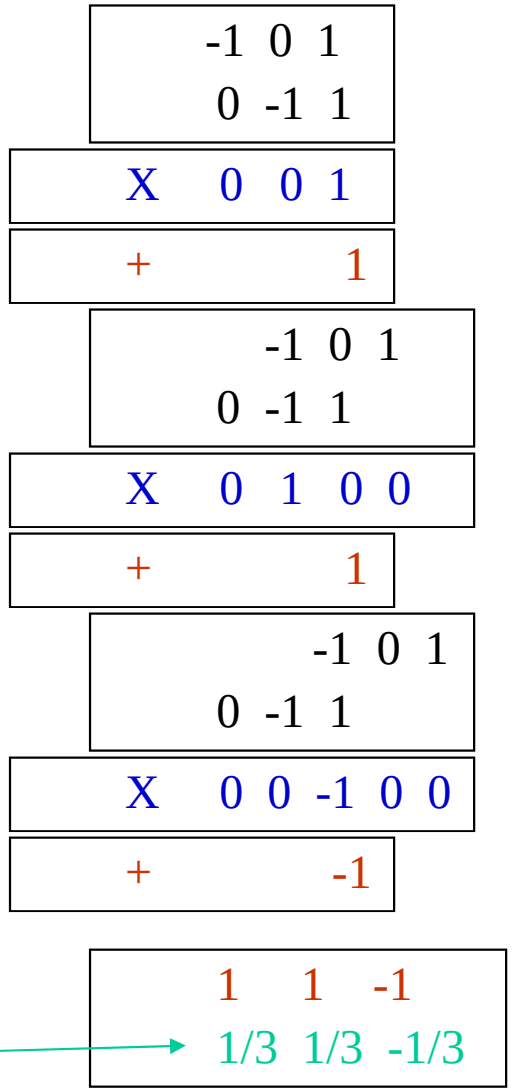
Signal a : a=[**-1 0 1**]

Signal b : b=[**0 -1 1**]

1. Pour chaque signal, soustraire sa moyenne de chaque chiffre, calculer le produit puis la somme des chiffres de ce produit.

2. déplacer progressivement le 2ème signal, calculer le produit puis la somme des chiffres de ce produit.

3. diviser chaque chiffre de résultat par la longueur de vecteur



Corrélation croisée

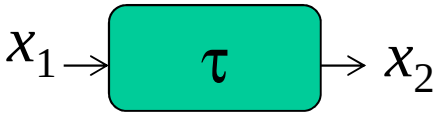
--> corr(a,b,3)

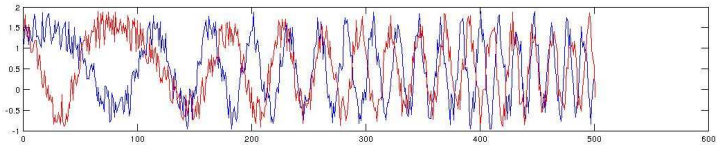
ans = ! 0.3333333 0.3333333 - 0.3333333 !

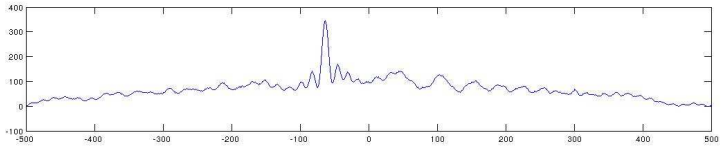
La corrélation n'est pas commutative

La corrélation peut être utilisée pour mesurer précisément le retard subi par un signal.

Imaginons qu'un signal échantillonné bruité x_1 est décalé (x_2) dans le temps par un déphasage ou un retard τ inconnu et bruité. Supposons que les bruits rajoutés à l'entrée et à la sortie sont gaussiens mais indépendants. La corrélation croisée maximale donne la meilleure estimation de ce τ.







```
[xc,lags]=xcorr(x1,x2);
[m,i]=max(xc);
tau=lags(i); % le retard est mesuré à partir de l'origine.
```

Corrélation croisée comme un outil pour aligner les signaux

La corrélation peut être utilisée pour mesurer la similarité entre plusieurs signaux.

Exemple

A. Corrélation croisée de deux segments de ECG

Utiliser la liste scilab « signals » (la variable que nous avons sauvée dans la première partie de ce cours au sujet du format EDF) pour extraire deux segments de battements ECG :

battement1 tous les échantillons entre ndx1=50 et 120

battement2 tous les échantillons entre ndx1=130 et 200

Tracer les signaux battement1, battement2 et le résultat de la corrélation croisée.

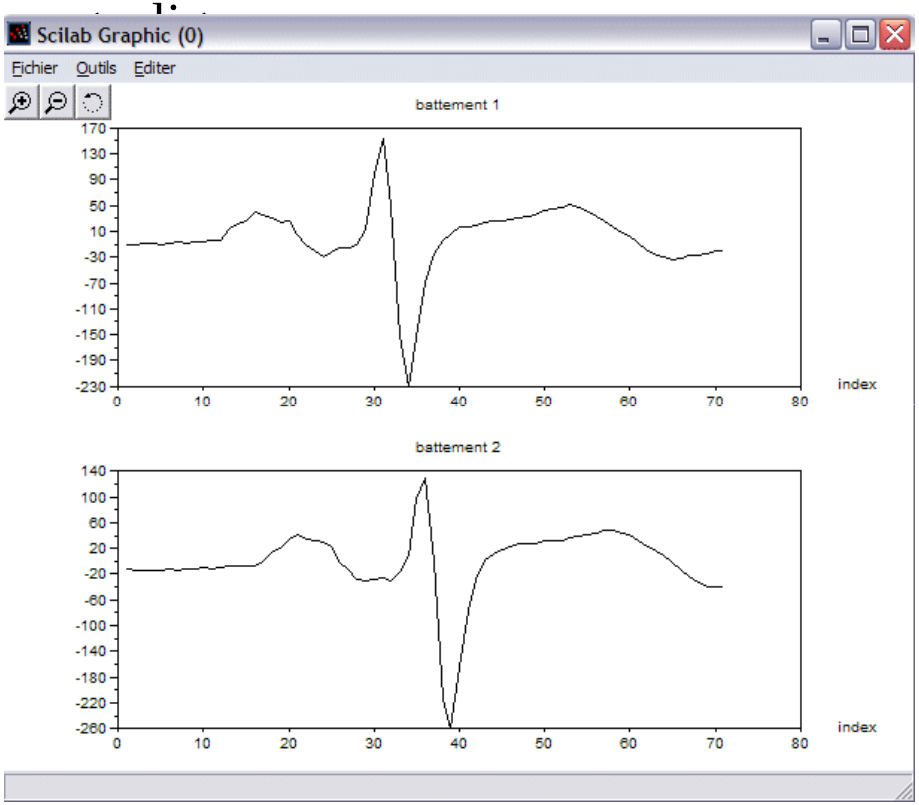
B. Aligner les deux segments de ECG

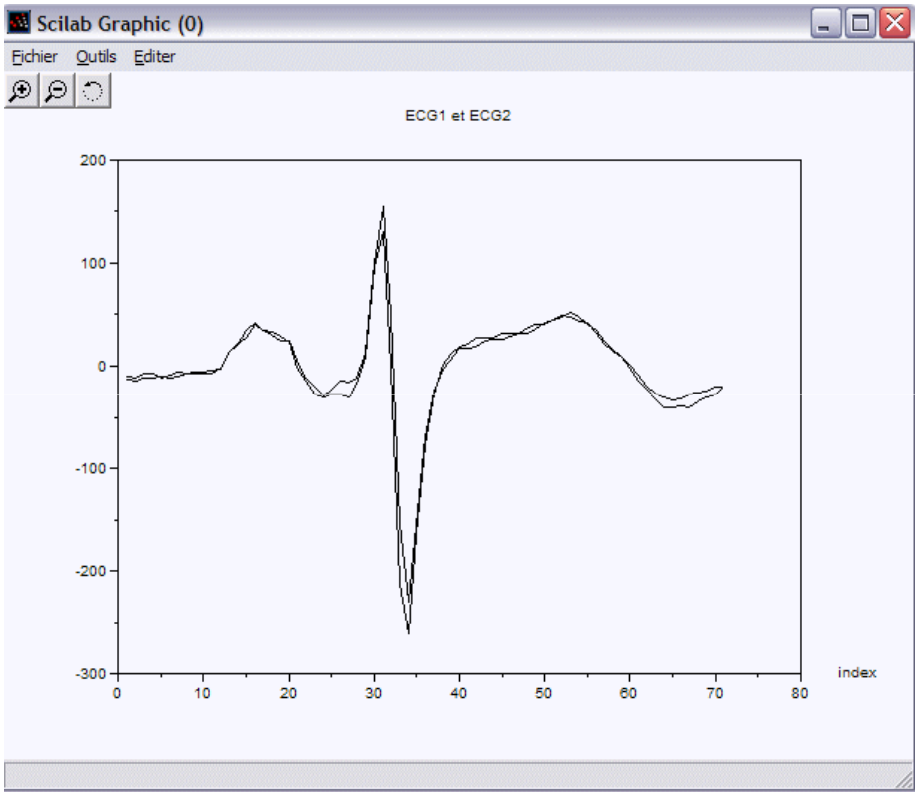
Utiliser le résultat de la corrélation croisée pour aligner les deux segments. Tracer les deux segments alignés sur le même graphe.

Corrélation comme un outil pour aligner les signaux

Exercice (suite) Résultat de la simulation

source: ../simulation2/sim_corr/Scilab/sim_ecg_convol_correl_align/





Si notre signal contient un artefact important, il peut dans ce cas altérer le processus de l'alignement. Il faut dans ce cas effectuer un pré-filtrage du signal (passe-haut ou passe-bas).

Corrélation comme un outil pour extraire les transitoires à partir d'un signal

Puisque la corrélation peut être utilisée pour mesurer la similarité entre plusieurs signaux, elle peut alors être utilisée pour détecter les transitoires.

Exemple

Supposons que nous sommes en train de travailler avec des signaux semblables à un signal EMG échantillonné à 20 kilohertz (l'intervalle d'échantillonnage est 0.05 ms) (voir exemple 0.6). Le signal contient deux formes analogues à un potentiel d'unité motrice (motor unit potential (pum)) et nous essayons de les localiser.

Corrélation comme un outil pour extraire les transitoires à partir d'un signal

Exemple (suite)

source : ../simulation2//sim_corr/Scilab/sim_emg_trans/ emg_trans.sce

```
inc=05;
// potentiel d'unité motrice
mup = - (sin(%pi*[0:127]/128).^4) .* (sin(2*%pi*[0:127]/128));
N=1000;
decharges = [4300  7500];
taux= zeros(1,N);
taux(decharges)=1;
emg=convol(taux,mup); // signal EMG
// Contamination avec de bruit Gaussian
rand('normal');
rand('seed',0);
emg_bruit = emg + 0.08*rand(emg); // écart-type=0.8
// Le processus de l'extraction implique l'utilisation d'un modèle de la forme que
// nous voulons détecter. Dans notre cas, pour détecter la forme « du potentiel
// d'unité motrice » nous employons un modèle qui contient la forme originale.
```

Corrélation comme un outil pour extraire les transitoires à partir d'un signal

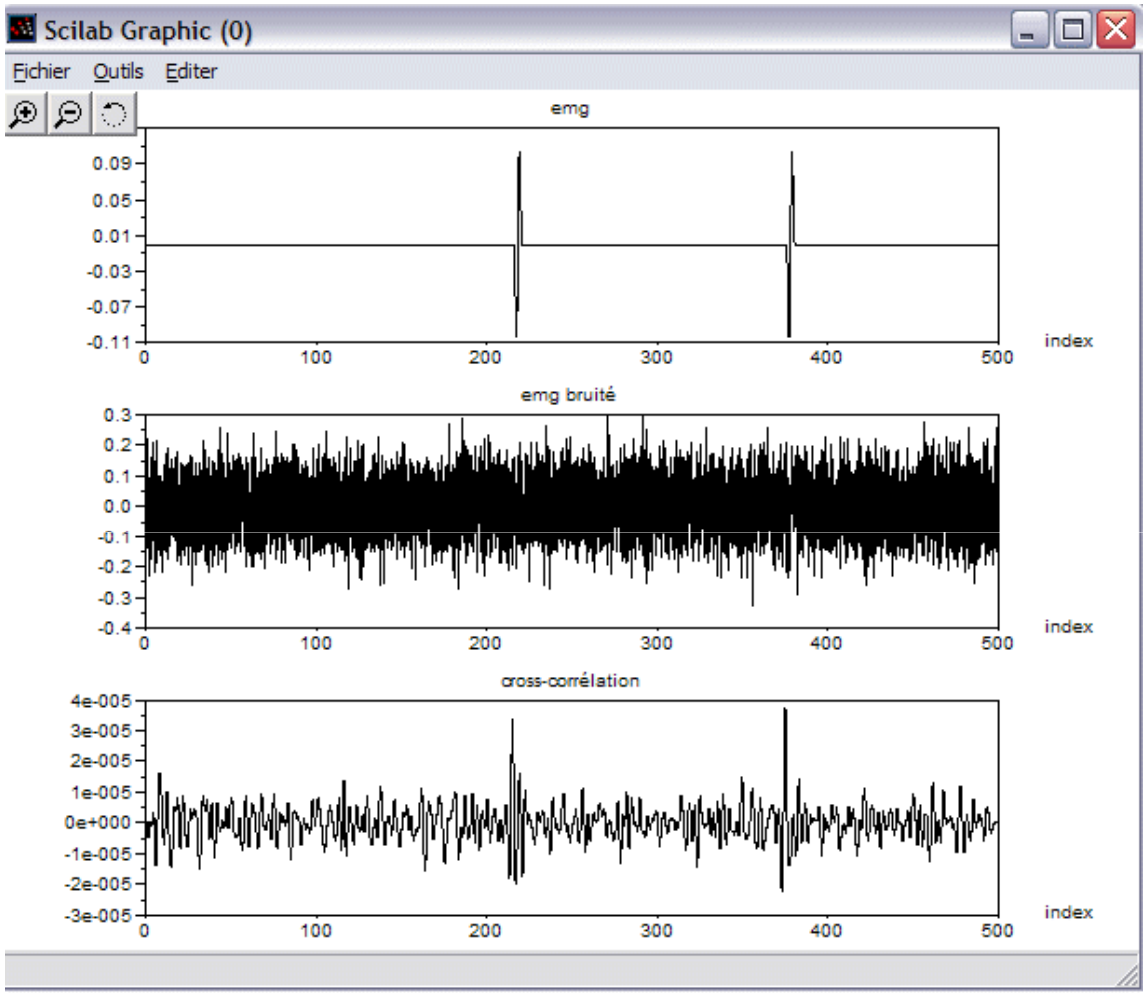
Exemple (suite)

```
// L'utilisation de la corrélation nécessite que les deux signaux ont la même longueur.
mup2 = [mup, zeros(1,N-length(mup))];
crosscorr = corr (mup2,emg_bruit,N);
// Tracer
xsetech([0,0,1,1/3]);
plot2d((1:N)*inct,emg);
xtitle('emg', 'index');
xsetech([0,1/3,1,1/3])
plot2d((1:N)*inct,emg_bruit);
xtitle('emg bruité ', 'index');
xsetech([0,2/3,1,1/3]);
plot2d((1:N)*inct,crosscorr);
xtitle('crosscorr', 'index');
```

Corrélation comme un outil pour extraire les transitoires à partir d'un signal

Exemple (suite)

- Notez que la corrélation détecte assez bien la position des potentiels d'unité motrice même dans un environnement bruité.
- Si nous avons un modèle nous pouvons localiser cette forme dans le signal même dans un environnement bruité.



Fonctions aléatoires gaussiennes

A. Cas de N réalisations d'une variable aléatoire scalaire

Une fonction aléatoire est appelée **gaussienne** si pour tous les instants t_i, la variable aléatoire X de N réalisations (échantillons) d'un processus aléatoire X : $x_1(t_i)$, $x_2(t_i)$, $x_2(t_i)$, … , $x_N(t_i)$, possède une **densité de probabilité gaussienne** :

$$f_{x_{ti}}(x_n(t_i)) = \frac{1}{(2\pi)^{1/2}\,\sigma_x^2}\exp(-\frac{1}{2\sigma_x^2}(x_n(t_i)-\mu_x)^2),$$

$$1 \le n \le N \text{ et } 1 \le i \le T$$

Une gaussienne scalaire est entièrement définie par deux paramètres μ et σ.

$m_X(t_k)$ est la **moyenne statistique estimée** à l'instant t_k de N échantillons

$$m_X(t_k) = \frac{1}{N}\sum_{n=1}^{N} x_n(t_k)$$

$V_X(t_k)$ est la **variancee statistique estimée** de N échantillons

$$V_X(t_k) = E[(x_k - m_X(t_k))^2] = \frac{1}{N-1}\sum_{n=1}^{N}(x_n - m_X(t_k))^2$$

Si le processus est **stationnaire à l'ordre deux** :

$m_X(t_k) = m_X$, $V_X(t_k) = V_X$, $\forall k$, $\sigma_{XX}(t_k, t_l) = \sigma_{XX}(\tau)$, où $\tau = t_l - t_k$

$\rho(t_k, t_l) = E[(X_k - m_X)(X_l - m_X)] = \sigma_{XX}(\tau) - m_X^2$.

Processus stochastique gaussien ergodique

Le processus est ergodique si

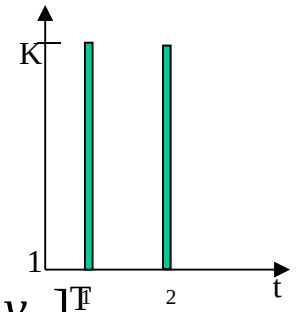
$$\int_{-\infty}^{+\infty} \left|(\sigma_{XX}(\tau) - \mu_X)\right|^2 d\tau < \infty$$

On considérera très souvent que les bruits possèdent les propriétés de processus stochastiques gaussiennes stationnaires à l'ordre deux et ergodiques.

Fonctions aléatoires gaussiennes

B. Cas d'un processus stochastique composé de K différents processus stochastiques

Soit le processus composite $X=\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \ldots \mathbf{x}_t, \mathbf{x}_{t+1}, \ldots \mathbf{x}_T\}$



Une fonction multivariée aléatoire est Gaussienne si pour tous les instants *t*, le vecteur aléatoire $\mathbf{x}_t$ est composé de *K* variables aléatoires différentes (provenant d'un nombre de processus stochastiques différents $Y_1, Y_2, .., Y_K$) : $\mathbf{x}_t = [y_1, y_2, y_2, \ldots, y_K]^T$ possède une **densité de probabilité multidimensionnelle Gaussienne**.

$$f_X(\mathbf{x}(t)) = \frac{1}{(2\pi)^{K/2}\,(\det(\boldsymbol{\Sigma}))^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x}(t)-\boldsymbol{\mu}_X)^T\,\boldsymbol{\Sigma}^{-1}\,(\mathbf{x}(t)-\boldsymbol{\mu}_X)\right)$$

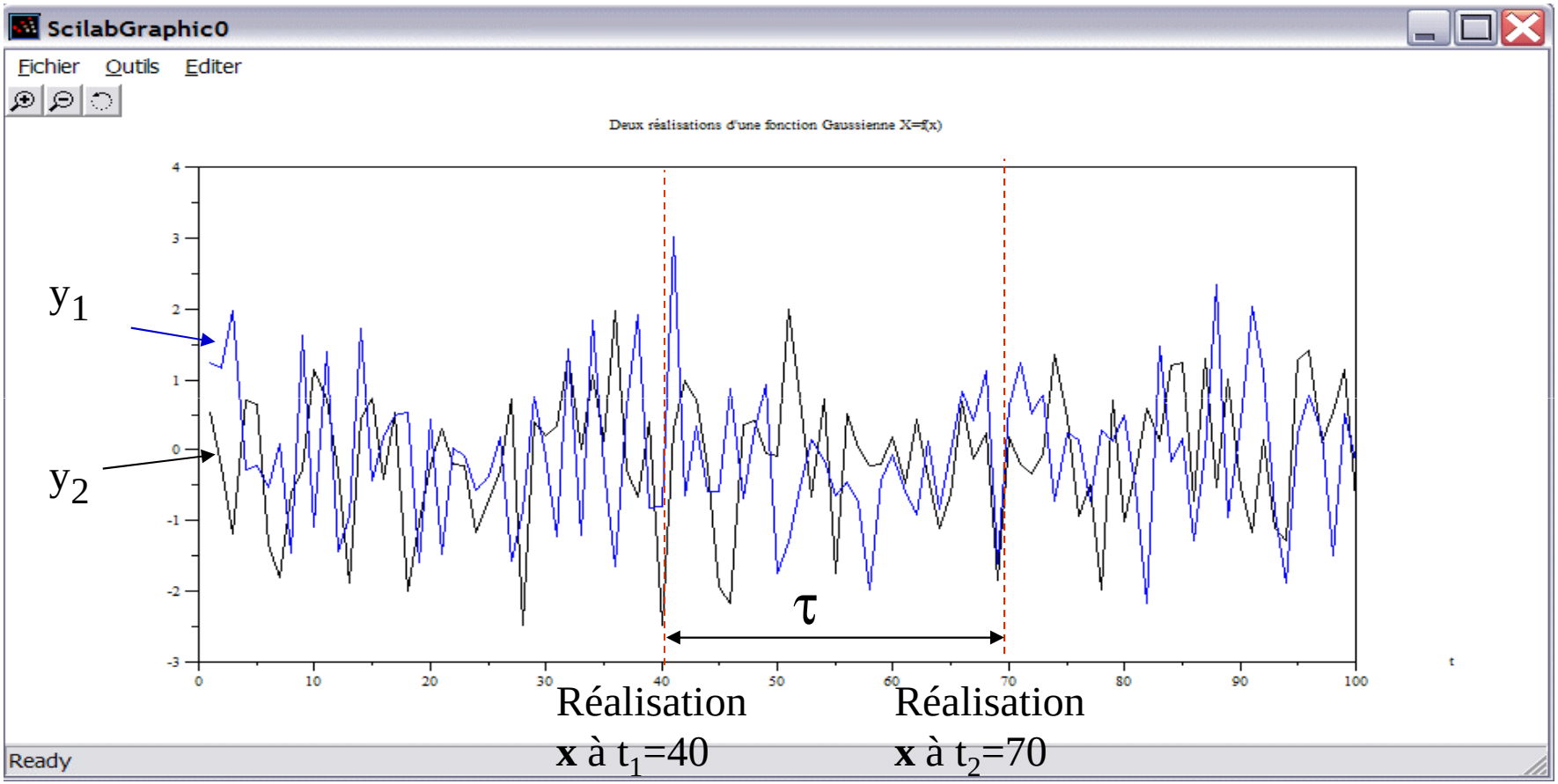
Une Gaussienne est entièrement définie par deux paramètres μ_X et Σ.

μ_X est le **vecteur des valeurs moyennes** de dimension K :

$\mu_X = [\mu_{y1}, \mu_{y2}, \mu_{y3}, \ldots, \mu_{yK}]^T$ et

det(Σ) est le déterminant de la **matrice de covariance** Σ de dimension KxK contenant les coefficients d'autocorrélation et de cross-corrélations entre les composantes de $\mathbf{x}_t$.

Exemple d'une réalisation d'une fonction gaussienne vectorielle X **d'un vecteur 2D aléatoire continue x =** $[y_1, y_2]$. Il s'agit de 2 processus aléatoires.



$$\Sigma = \begin{bmatrix} \rho_{11} & \rho_{12} & \rho_{1K} \\ \rho_{21} & \rho_{22} & \rho_{2K} \\ . & . & . \\ . & . & . \\ \rho_{K1} & \rho_{K2} & \rho_{KK} \end{bmatrix}$$

Où $\rho_{ij} = \text{cov}(y_i, y_j), \forall i, j \in \{1, 2, \ldots, K\}$
la ***covariance*** entre les variables y_i et y_j est $\rho_{ij} = E[(y_i - \mu_{yi}) (\mathbf{y}_j - \mu_{yj})^T]$,
Si les variables y_i et y_j sont indépendantes, alors $\rho_{ij}(\tau) = 0$ et $\sigma_{ij}(\tau) = \mu_{yi}\, \mu_{yj}\ \forall\ \tau$ et $\forall\ i \neq j \in \{1, 2, \ldots, K\}$ et dans ce cas la matrice de covariance est une matrice diagonale.

Entropie

Le concept de mesure d'information, en statistiques appelé entropie, a été introduit par Shannon [Shannon, 1948]. Ce concept est relié à la probabilité de l'occurrence d'un événement (variable aléatoire). Supposons que cet événement a M possibilités de se produire avec une probabilité p_i, alors l'information liée à l'occurrence de la i^{th} possibilité sera

$$I_i = - log\ p_i$$

La valeur espérée de l'information est l'***entropie*** :

$$H_n = -\sum_{i=1}^{M} p_i \log p_i$$

L'*entropie* est une mesure de l'incertitude associée aux résultats d'un événement. Plus l'*entropie* est élevée, plus élevée l'***incertitude*** sur la possibilité de l'occurrence de cet événement. La valeur la plus élevée de l'entropie est obtenue lorsque toutes les possibilités sont également probables.

Dans la pratique, pour les séries chronologiques (ou temporelles) l'*entropie* est calculée à partir des distributions d'amplitude. La gamme d'amplitude A d'un signal brut échantillonné est divisé en K intervalles disjoints I_i, pour $i = 1,\ldots,K$.

La distribution de probabilité peut être obtenue à partir du rapport de la fréquence des échantillons N_i tombant dans chaque casier I_i et le nombre total d'échantillon N :

$$p_i = N_i / N$$

La distribution $\{p_i\}$ de l'amplitude du signal échantillonné est alors utilisée pour calculer l'entropie.

3. Filtres

Effet de l'échantillonnage

Les échantillonnages étudiés se font à période T_e. Le signal continu de départ est donc multiplié par la fonction d'échantillonnage ε(t).

$$\hat{x}(t) = x(t)\varepsilon(t) = \sum_{-\infty}^{+\infty} x(kT_e)\delta(t - kT_e)$$

La transformée de Fourier devient donc :

$$\hat{x}(f) = x(f)E(f) = \frac{1}{T_e}\sum_{-\infty}^{+\infty} x(f - \frac{n}{T_e})$$

où x(f) et E(f) sont les TF de x(t) et ε(t) respectivement.

Effet de l'échantillonnage (suite)

Pour traiter des signaux échantillonnés, il est important de noter que quelques fréquences deviennent non différentiables des autres signaux; à une fréquence d'échantillonnage f_e les seules fréquences principales sont dans la gamme 0 à $f_e/2$ Hz. Toutes fréquences supérieures à $f_e/2$ Hz seront des ***fréquences de repliement*** (aliases) de l'une de ces fréquences principales.

En générale, si f_0 est une fréquence principale, alors ses fréquences de repliement sont

$$f_k = f_0 + k f_e \text{ , } k = \pm 1, \pm 2, \pm 3, \ldots$$

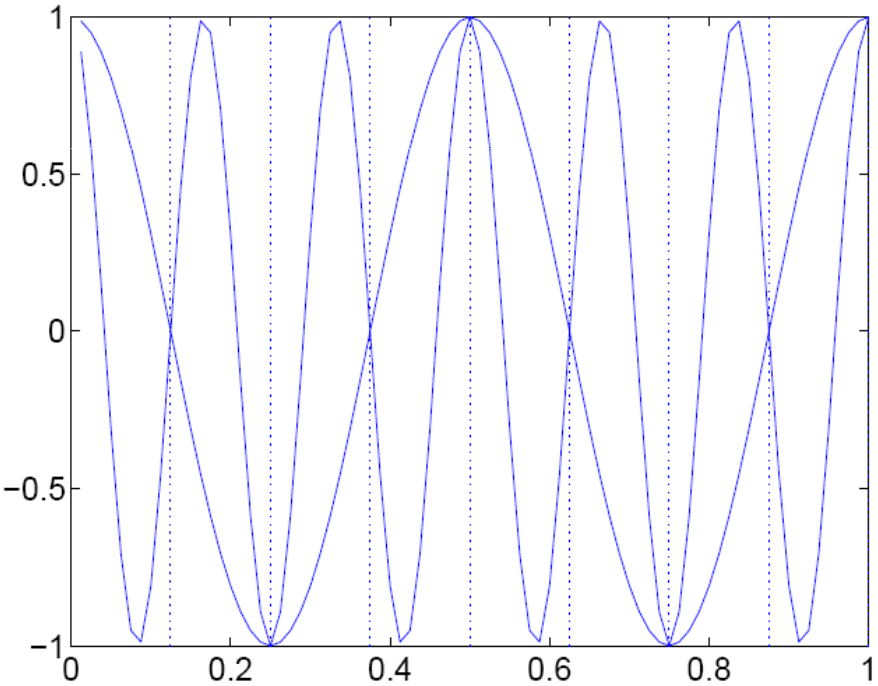
Effet de l'échantillonnage (suite)

La figure suivante montre un signal cosinus à fréquence f_0 = 2 Hz (courbe pleine). A des instants du temps donnés par les lignes pointillées qui correspondent à une fréquence d'échantillonnage f_e de 8 Hz,les fréquences sans repliements sont 0 → 4Hz.

Les fréquences de repliement sont : f_{-1} = -6 , 10, -14, 18, …. Hz

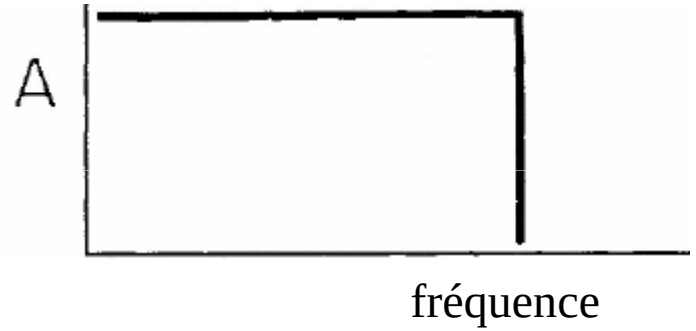
A titre d'exemple, le signal à f_{-1} = -6 Hz est un signal de repliement de signal à f_0 = 2 Hz.

A cause de la phénomène de repliement, nous devons s'assurer qu'il n'existe pas des fréquences supérieur à f_e /2. Ainsi, avant échantillonner le signal il est obligatoire de filtrer ce signal en utilisant un filtre analogique passe-bas.



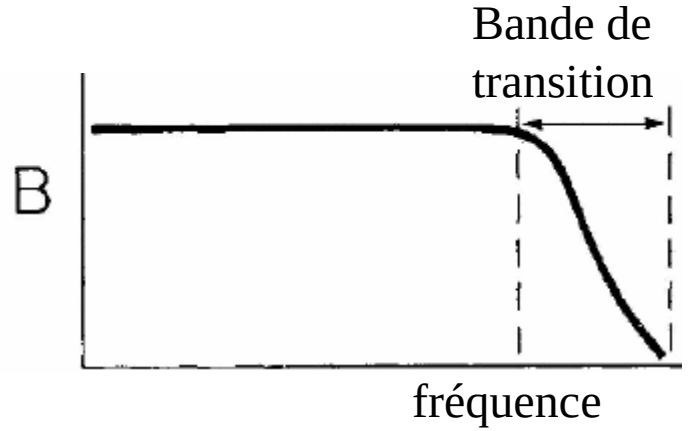
A. Filtre analogique anti-repliement idéal

Éviter la conception d'un filtre anti-repliement avec une réponse "trop raide" sur la moitié de la fréquence d'échantillonnage.



B. Filtre analogique anti-repliement réel

Dans la pratique, la fréquence de coupure (f_c) de le filtre peut être choisi un peu au-dessus $f_{\max}$.
Ensuite, sélectionnez la fréquence d'échantillonnage $f_e > 3f_c$.



Filtres numériques

Les signaux peuvent contenir beaucoup de composants de fréquences. Certains d'entre eux peuvent être d'intérêt comme les bandes de fréquence des rythmes EEG; d'autres comme l’artefact de la ligne électrique peuvent être indésirables. Pour supprimer la partie indésirable de contenu spectral d'un signal échantillonné on utilise généralement des filtres.

Afin d’implémenter un filtre, il doit être ***causal***. Autrement dit, la réponse du filtre doit dépendre uniquement des entrées actuelles et passées. Si le filtre doit dépendre aussi des sorties passées, il peut encore être aussi causale par l'introduction de la rétroaction avec des délais appropriés.

Nous présentons ici que les ***filtres numériques*** implémentés dans les logiciels (pas les dispositifs matériels électroniques utilisés. Par exemple, les filtres anti-repliement avant l'échantillonnage).

Les pluparts des filtres numériques réalisent l'action suivante: Ils calculent une combinaison linéaire d'un certain nombre n_b de dernières entrées x et un certain nombre n_a de dernières sorties y pour évaluer la sortie courante $y[n]$:

$$y[n] = b(1) * x[n] + b(2) * x[n-1] + \ldots + b(n_b+1) * x[n-n_b] - a(2) * y[n-1] - \ldots - a(n_a+1) * y[n-n_a]$$

Le nombre maximum des échantillons utilisés dans la création de chaque échantillon de sortie est appelé l'***ordre du filtre*** (n). Dans la représentation précédente de l'équation de différence, l'ordre n est le plus grand nombre entre n_b et n_a ($n = max(n_b, n_a)$).

Il existe 3 possibilités :

➢Si $n_a = 0$ et $n_b \neq 0$, alors le filtre est appelé ***Filtre à réponse impulsionnelle finie*** (***finite impulse response*** (***FIR***)), ce qui signifie qu'une fois que le filtre est alimenté par une impulsion non-nulle la réponse aura une durée finie. Après des intervalles de temps n_b la sortie revient à zéro. Ce filtre est parfois appelée ***filtre à moyenne mobile***, (***MM*****)**.

➢Si $n_a \neq 0$ et $n_b = 0$, le filtre est appelé ***filtre récursif*** ou ***filtre autorégressive*** (***AR***).

➢Si $n_b \neq 0$ et $n_a \neq 0$, c'est le type le plus général du filtre. Il est aussi appelé ***IIR*** ou ***ARMA-autorégressive à moyenne mobile*****.**

Le fonctionnement du filtre est beaucoup plus facile à comprendre dans le domaine fréquentiel. D'abord nous allons réorganiser l'équation précédente

$$y[k] = -a_1 * y[k-1] - \ldots - a_{na} * y[k-n_a]$$

$$b_0 + b_1 * x[k-1] + b_2 * x[k-2] + \ldots + b_{nb} * x[k-n_b]$$

En appliquant la transformée en Z :

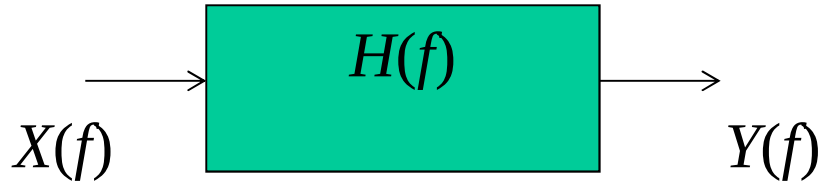
$$A(z)Y(z) = B(z)X(z) \quad \text{ou} \quad Y(z) = H(z)X(z)$$

$H(z) = B(z)/A(z))$ est appelé la ***fonction de transfert***

$$\frac{Y(z)}{U(z)} = \frac{b_0 + b_1 z^{-1} + \ldots + b_m z^{-m}}{1 + a_1 z^{-1} + \ldots + a_n z^{-n}} = \frac{B(z)}{A(z)}$$

En substituant $z = exp\ (\ j2\pi\ f\ )$, nous pouvons obtenir une fonction de transfert qui est fonction de fréquence

$$H(f) = G\ (\ f\ )\ exp\ (\ j\varphi\ (\ f\ ))$$



Alors, à partir de $Y(z) = H(z)X(z)$, qui devient $Y(\ f\ ) = H(\ f\ )X(\ f\ )$, nous voyons que le fonctionnement d'un filtre est la multiplication de chacune des composantes de Fourier du signal $X(\ f\ )$ par le nombre complexe $H\ (\ f\ )$, c-à-d que le filtre modifie l'amplitude et la phase de chaque composante.

Une mesure très utile peut être dérivée de la phase de la fonction de réponse fréquentielle $H(f)$: le ***retard de groupe***. Le retard de groupe

$$\tau_g(f) = -\frac{d\varphi(f)}{df}$$

est une mesure du retard temporel introduit par le filtre à la composante de signal de fréquence *f*.

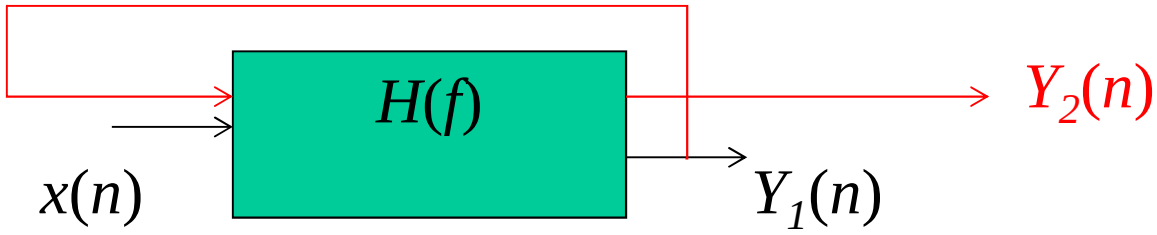
Si la phase φ dépend linéairement de la fréquence, alors $\tau_g(f)$ = const. Cela signifie que toutes les composantes de fréquence sont également retardées. La structure de la phase du signal n'est pas déformée, ce qui est essentiel, si le filtrage est utilisée comme une étape de prétraitement pour des méthodes plus avancées de traitement du signal qui sont sensible à la phase.

Un tel retard de phase linéaire est la propriété des filtres *FIR*. En cas d'applications hors ligne les retards apportés par le filtre, dans n'importe quelle forme (pas seulement les linéaires), peuvent être corrigés. La technique repose sur l'application de même filtre deux fois.

Après filtrage du signal dans la direction avant, la séquence filtrée est inversée dans le temps et parcours à travers le filtre. En raison de la propriété de retournement temporel de la transformée de Fourier, la composante $X(f)$ du signal d'origine est multiplié effectivement par

$$H_{\text{eff}}(f) = G(f)\,exp(j\phi(f)) \,.\, G(f)\,exp(-j\phi(f)) = G(f)^2$$

Cette opération double ainsi l'ordre du filtre.



Dans MATLAB cette technique est implémentée par la fonction « *filtfilt.m* » qui utilise le filtre *IIR* pour obtenir un filtre avec un déphasage (ou retard) nul. En Scilab, il faut appliquer 2 fois le filtre « *iir.m* », voir Figure ci-dessus.

Conception et application de filtres

Les applications pratiques de filtres nécessitent qu'ils répondent à certaines exigences. Souvent, les caractéristiques d'un filtre sont exprimées comme les propriétés des réponses en amplitude (la valeur absolue de la fonction de transfert), tels que :

•la ***fréquence de coupure*** ou de la ***bande de fréquence*** à être atténuée ou passé,

• la ***quantité de l'atténuation*** des composantes spectrales indésirables,

•la ***pente du filtre***, ou l'***ordre de la fonction de transfert***.

En précisant les caractéristiques du filtre, une unité appelée ***décibels*** [*dB*] est couramment utilisée. Deux niveaux de signal P et P_0 diffèrent par n décibels, si

$$n = 10 \log_{10} \frac{P}{P_0}$$

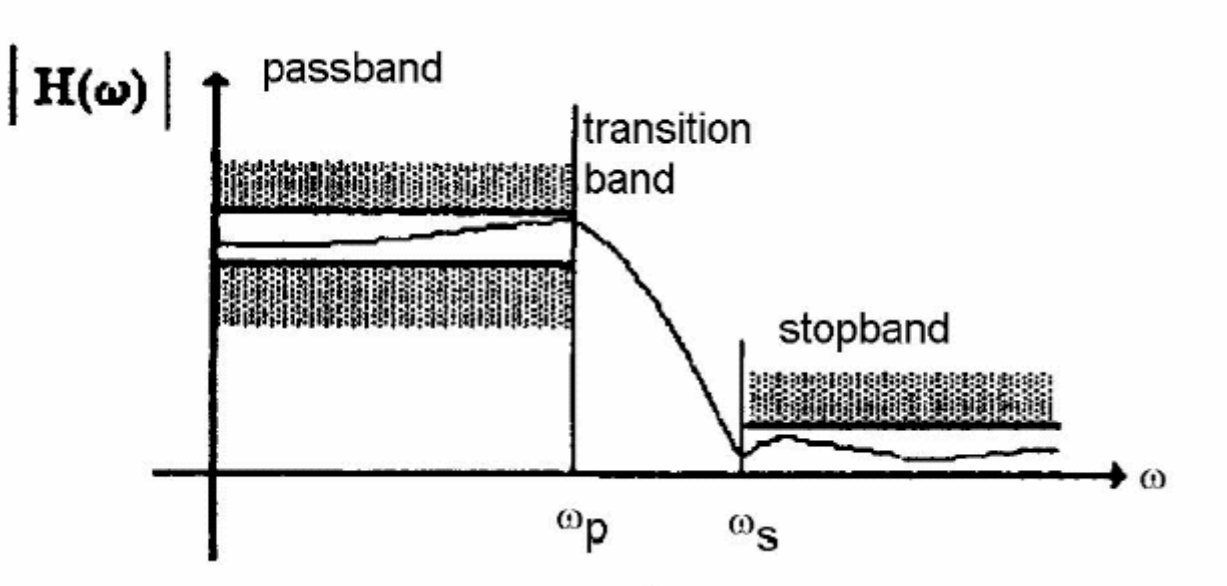
Les termes utilisés pour la spécification des caractéristiques de filtrage sont:

BP (en anglais ω_p): bande passante, bande de fréquence qui doit être transmis à travers le filtre sans atténuation.

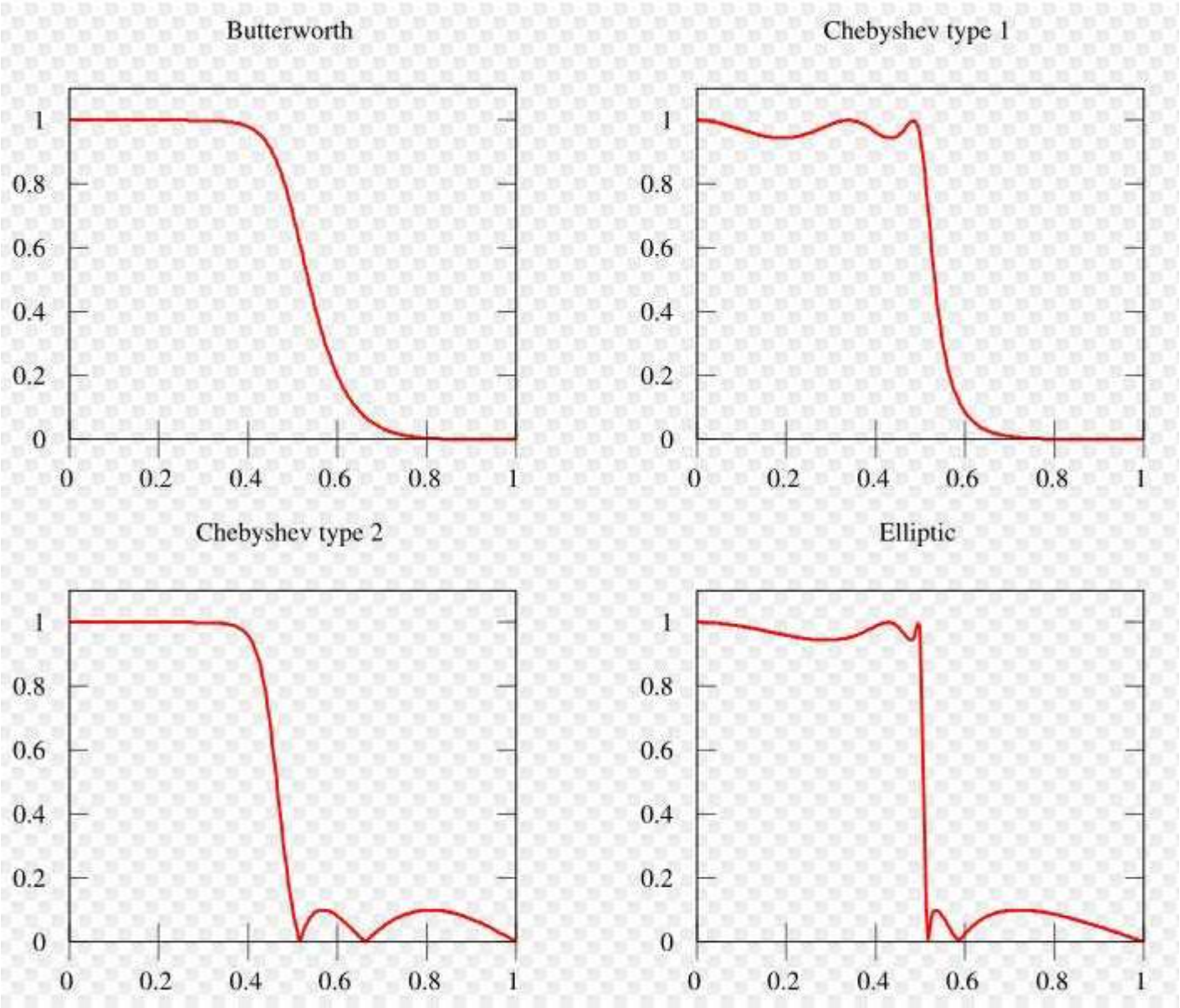
SB (en anglais ω_s) : stop-bande, bande de fréquence qui doit être atténuée.

O_p (en anglais $\mathbf{R_p}$) : les ondulations autorisées pour le passe-bande. exprimés en [*dB*].

O_s (en anglais $\mathbf{R_s}$) atténuation minimum souhaitée dans le stop-bande exprimée en [dB].



- ω_P – Passband edge frequency
- ω_s – Stopband edge frequency
- Passband (R_p) and stopband (R_s) attenuation
- Transition width(s)


Butterworth
Chebyshev type 1
Chebyshev type 2
Elliptic
0
0.2
0.4
0.6
0.8
1

Les exigences concernant les caractéristiques du filtre sont généralement contradictoires, par exemple, pour un ordre de filtre donné l'augmentation de la raideur du filtre produit des grandes ondulations. Par conséquent, certains processus d'optimisation doivent être appliqués et le filtre résultant est alors un compromis entre l'ensemble des conditions.

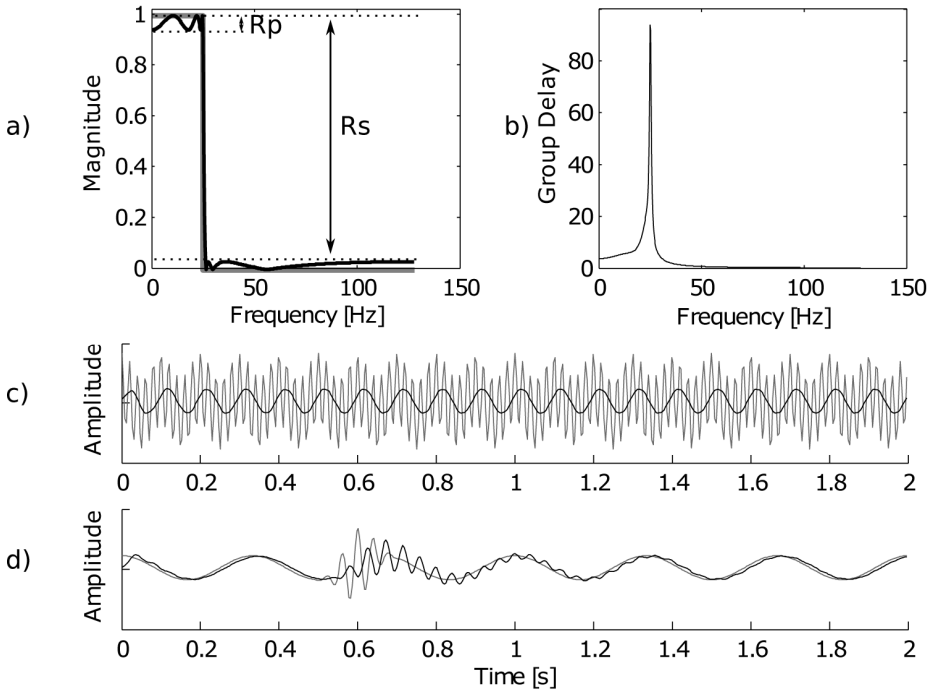
Une décision doit être prise pour choisir parmi les conceptions possibles le meilleur filtre pour l'usage prévue. Les Toolboxes de Matlab ou Scilab, facilitent la conception et la validation des filtres. Une tracé de la réponse désirée en amplitude et l'amplitude réelle permet de vérifier si la réponse réelle de la fonction $H(f)$ est assez proche de celle désiré.

On doit faire attention à la pente des bords de la réponse en amplitude (la largeur de la transition entre le stop-, et le passe-bande). L'augmentation de l'ordre du filtre doit faire la transition plus raide. Mais en même temps, il faut aussi considérer l'ampleur des ondulations. Ils peuvent être plus petits en diminuant l'ordre du filtre ou en augmentant la largeur de transition.

Si le filtre est à utiliser en mode standard unidirectionnel ($n_b \neq 0$ et $n_a \neq 0$), il est également conseillé d'observer la fonction de retard du groupe pour évaluer les retards de différentes composantes de fréquence. Ces retards peuvent être évalués au moyen de la fonction « *grpdelay.m* » de Matlab Signal Processing Toolbox ou « *group.sci* » en Scilab.

Un exemple d'illustration des propriétés d'un filtre en utilisant Matlab.

Adapté de (K J Blinowska., J Zygierewicz 2012)



a) Magnitude $|H|$. Les lignes en gris représentent une réponse idéale, en noire la magnitude du filtre conçu en utilisant un filtre elliptique d'ordre 5 (voir les différents types en Matlab). Les flèches indiquent les ondulations autorisé de passe-bande ($\boldsymbol{R_p}$) et l'atténuation minimum dans le stop-bande ($\boldsymbol{R_s}$).

b) Réponse de la fonction Matlab de retard du groupe (*grpdelay.m*) .

c) Application du filtre conçu pour un signal sinusoïde composé d'un 10 Hz et 50 Hz, gris : entrée, noir : sortie.

d) Illustration du retard et les effets de bord du filtre: le signal d'entrée (gris) est un sinusoïde de 3Hz avec une partie transitoire de 25 Hz à 0,6 s. La sortie en noire montre (i) dans le premier 0,1 s une distorsion de bord du filtre (ii) des retards: la sortie 3 Hz sinusoïde est légèrement retardée par rapport à l'entrée; le transitoire de 25 Hz est plus retardé et propagé, ce qui qui est montré par la fonction de retard du groupe (b).

Il existe 2 types de ***filtres Infinit Input Response*** (***IIR***) et ***Finite Input Résponse*** (***FIR***).

- Le filtre ***IIR*** est défini pas sa façon de répond à une seule impulsion à l'entrée. La sortie du filtre est bouclée sur son entrée. La réponse à une seule impulsion doit s'amortir graduellement vers zéro mais théoriquement n'éteindra jamais la valeur zéro.

La sortie du filtre ***IIR*** est donnée par :

$$y[n] = \sum_{k=0}^{n_a - 1} a_k y[n-k] + \sum_{k=0}^{n_b - 1} b_k x[n-k]$$

où le premier terme est un modèle autorégressif qui inclut les coefficients de bouclage (feedback coefficients) et le second terme est an modèle FIR.

Ce type de filtre est appelé modèle ***Autorégressif à Moyenne Mobile*** (Autoregressive Moving Average (ARMA)).

Les filtres ***IIR*** peuvent être conçus en implémentant les versions numériques de filtres analogiques (e.g. ***Butterworth***, ***Chebyshev*** et ***Bessel***).

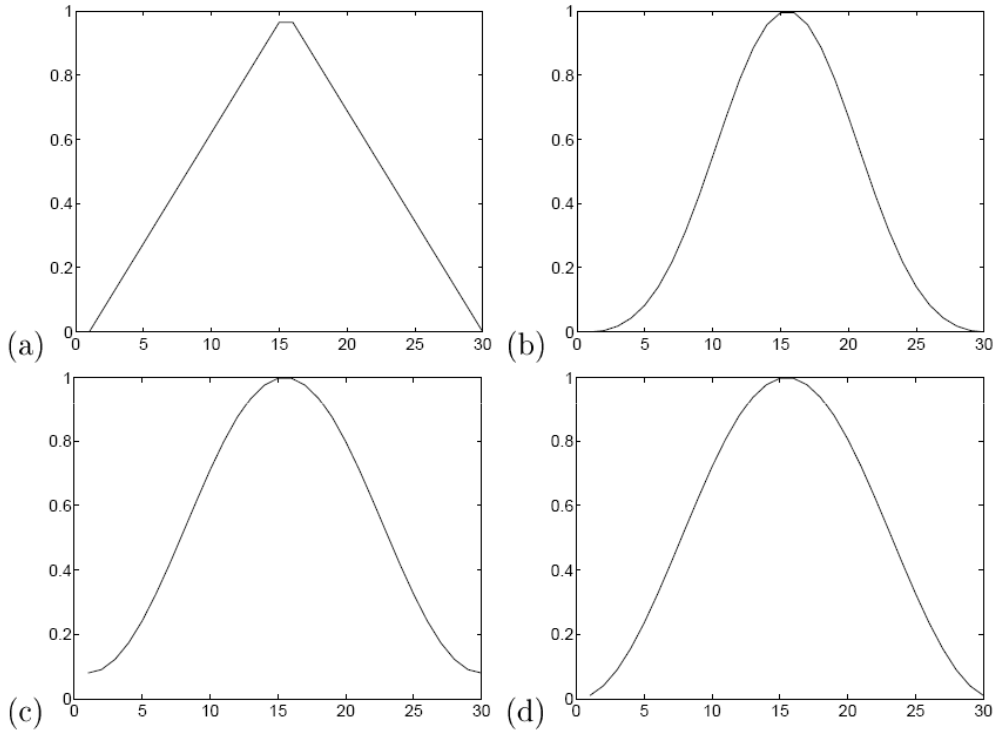
- Le filtre FIR d'ordre *p* est donnée par :

La réponse de ce filtre dépende uniquement sur les entrées courantes et passées.

$$y[n] = \sum_{k=0}^{n_b - 1} b_k \, x[n-k]$$

Le filtre ***FIR*** utilise des **fenêtres**.

- Le filtre ***FIR*** utilise différents types de fenêtres, par exemple ***Bartlett***, ***Blackman***, ***Hamming*** et ***Hanning***



Coefficients du filtre de (a) Bartlett (triangulaire), (b) Blackman, (c) Hamming et (d) Hanning pour p=30. Les fenêtres les plus courbées signifie que leurs caractéristiques fréquentielles sont définies d'une façon plus aigue.

FIR est un filtre qui fait un déphasage linéaire, FIR est un filtre passe-bas, passe-haut, et passe-bande qui utilise la technique de ***fenêtrage*** « ***windowing*** »

Fenêtrage :

Le **fenêtrage** est utilisé dès que l'on s'intéresse à un signal de longueur volontairement limité. En effet, un signal réel ne peut qu'avoir une durée limitée dans le temps; de plus, un calcul ne peut se faire que sur un nombre de points fini.

Exemple: Analyser une partie du signal. Imaginez que nous avons un signal de 1000 valeurs mais nous sommes intéressés à une partie de ce signal, par exemple 100 valeurs.

Toute observation étant de durée limitée, on applique forcément une fenêtre par rapport à un signal théorique infini.

Pour observer un signal sur une durée finie, on le multiplie par une fonction **fenêtre d'observation** (également appelée ***fenêtre de pondération*** ou d'***apodisation***). Au lieu d'étudier le signal $s(t)$, on étudie le signal tronqué : $s_h(t) = s(t)h(t)$.

Exemple : ***fenêtre rectangulaire*** (ou porte)

$$h(t) = \begin{cases} 1 & \text{si } t \in [0, T] \\ 0 & \text{sinon.} \end{cases}$$

Ainsi, quand on multiplie un signal $s(t)$ par cette fenêtre, on n'obtient plus que les T secondes de ce signal : on l'observe donc que sur une durée T.

Fenêtre triangulaire (de Bartlett) :

$$h(t) = \begin{cases} \frac{2t}{T} & \text{si } t \in [0, \frac{T}{2}[\\ \frac{2(T-t)}{T} & \text{si } t \in [\frac{T}{2}, T] \\ 0 & \text{sinon.} \end{cases}$$

Fenêtre de Hann :

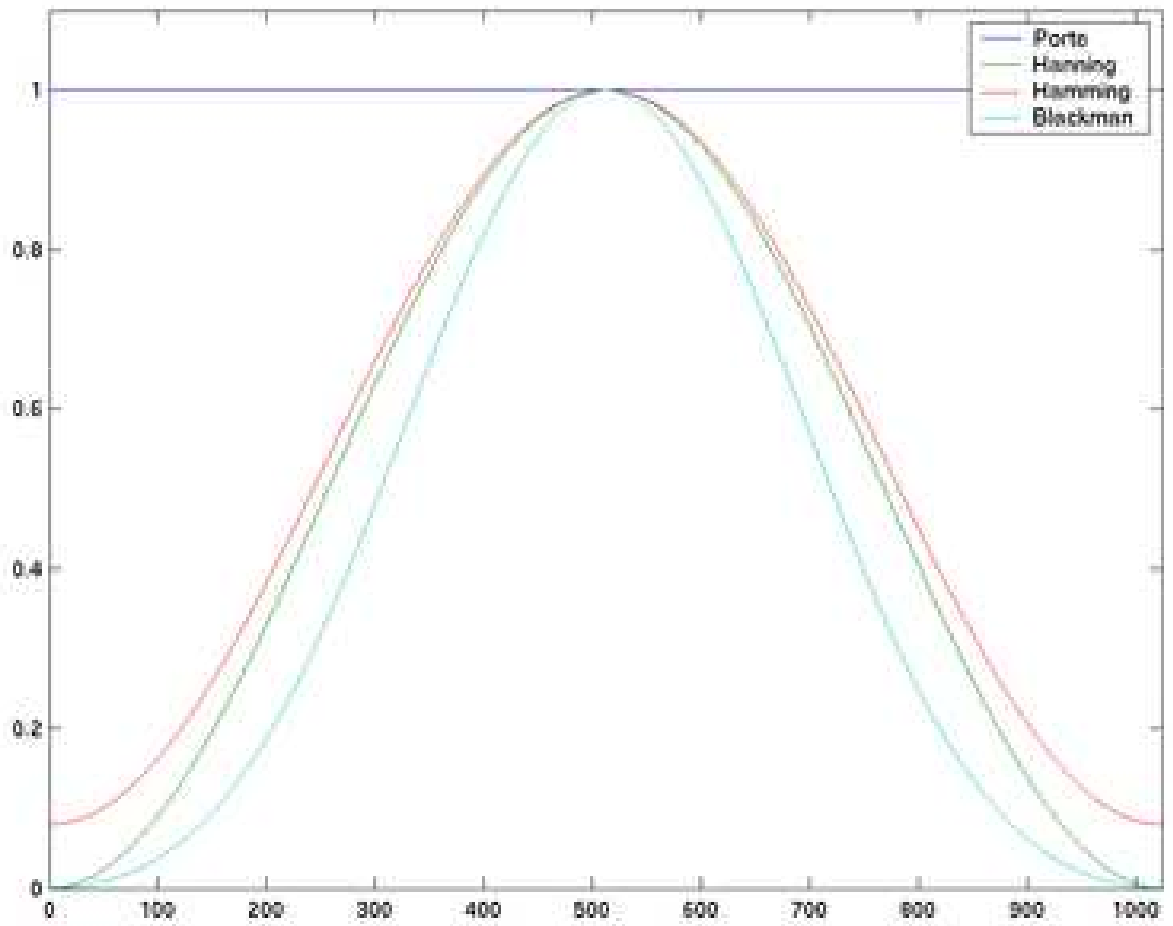
$$h(t) = \begin{cases} 0.5 - 0.5\cos 2\pi\frac{t}{T} & \text{si } t \in [0, T] \\ 0 & \text{sinon.} \end{cases}$$

Fenêtre de Hamming :

$$h(t) = \begin{cases} 0.54 - 0.46\cos 2\pi\frac{t}{T} & \text{si } t \in [0, T] \\ 0 & \text{sinon.} \end{cases}$$

Fenêtre de Blackman :

$$h(t) = \begin{cases} 0.42 - 0.5\cos 2\pi\frac{t}{T} + 0.08\cos 4\pi\frac{t}{T} & \text{si } t \in [0, T] \\ 0 & \text{sinon.} \end{cases}$$


Porte
Hanning
Hamming
Blackmnn
1
0.8
0.6
0.4
0.2
0
0
100
200
300
400
500
600
700
800
900
1000

Dans le domaine des fréquences

En passant dans le domaine fréquentiel via une ***transformée de Fourier*** (*TF*), on obtient le produit de convolution :
$S_h(f) = S(f)*H(f)$, où $H(f)$ est la TF de la fenêtre.

L'utilisation d'une fenêtre de pondération va changer la transformée de Fourier du signal.

La TF du signal analysé (H(f)) est convoluée avec la TF de la fenêtre (S(f)).

Idéalement, pour ne pas biaiser le spectre initial, il faudra que l'allure de la fenêtre spectrale soit une fonction de Dirac. Or, le signal temporel ayant un spectre en fonction de Dirac est un signal constant infini, ce qui est impossible en pratique.

Les allures spectrales des fenêtres de pondérations présentent une succession de lobes : **pour se rapprocher d'une fonction de Dirac, il faut que le lobe principal soit le plus étroit possible, tandis que les lobes secondaires doivent être les plus faibles possible**.

Mais, plus le lobe principal d'une fenêtre aura tendance à être étroit, plus ses lobes secondaires seront importants...
Il y a donc toujours un compromis à faire entre la largeur du lobe principal et l'importance des lobes secondaires.

Exemple

lobe principale

lobes secondaires

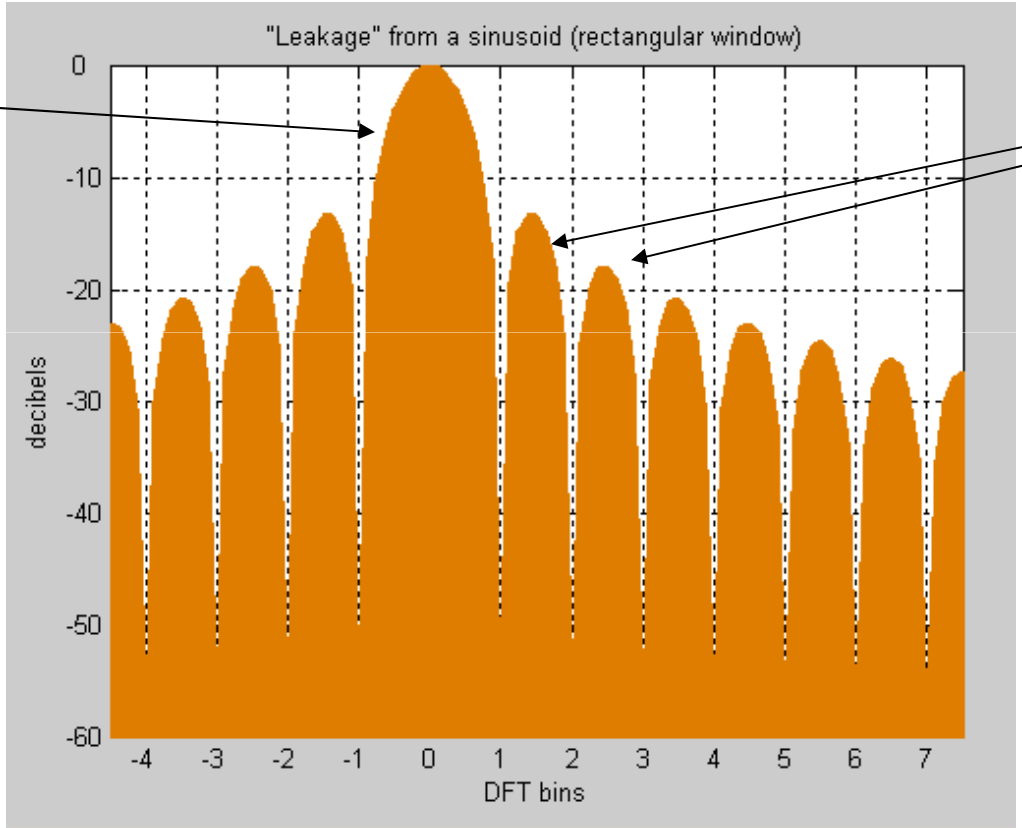
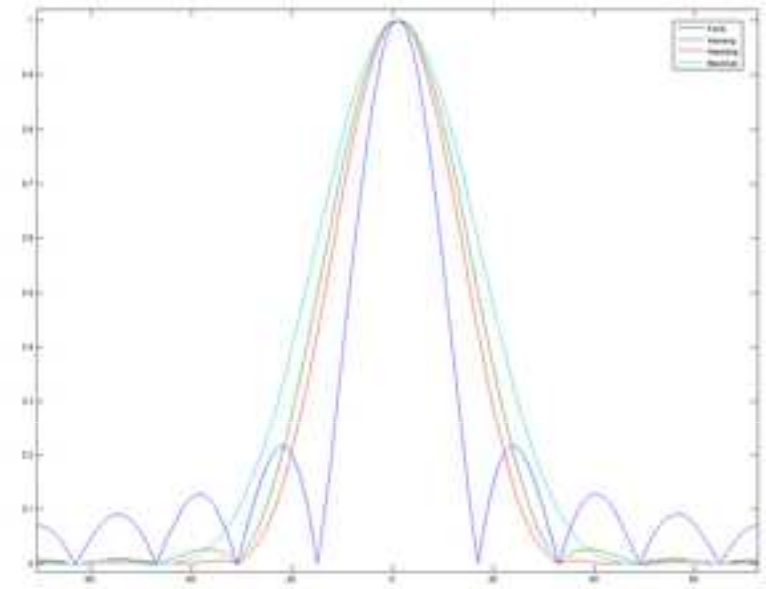
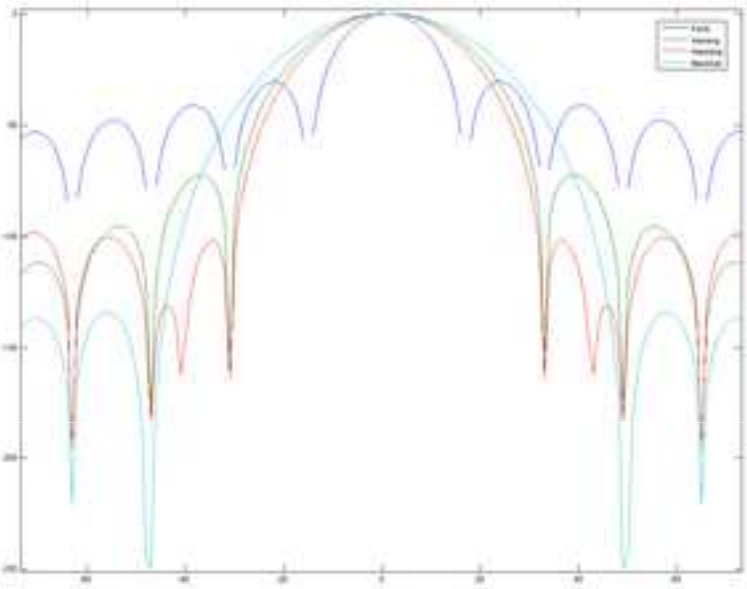


Fig. 6

Représentation linéaire



Représentation linéaire



Représentation logarithmique

La fenêtre rectangulaire (bleu) a le lobe principal le plus étroit, mais ses lobes secondaires sont les plus importants; au contraire, celle de Blackman (vert) a les plus faibles lobes secondaires, mais un lobe principal plus large.

Fenêtrage (suie):

Le type de fenêtre est à choisir selon ce qu'on souhaite observer d'un spectre : la **localisation des maximums**, la **localisation de l'énergie** selon les fréquences, les **valeurs des maximums**, etc.

Voici quelques caractéristiques principales des fenêtres d'analyse courantes :

Fenêtre	Bande passante (bins)	Perte au pire des cas (dB)
Rectangulaire	1.21	3.92
Triangulaire	1.78	3.07
Hann	2.00	3.18
Hamming	1.81	3.10
Blackman	1.81	3.45

Fenêtrage (suie):

Influence de la taille de la fenêtre

Plus la fenêtre choisie aura une grande durée temporelle, plus elle sera étroite dans le domaine fréquentiel. Ainsi, en prenant une fenêtre infiniment large dans le domaine temporel, on aboutit à la limite à un fonction de Dirac dans le domaine fréquentielle, qui est l'élément neutre du produit de convolution.

Ainsi, pour une fenêtre infiniment large, on retrouve le spectre « réel » du signal analysé, qui correspond effectivement à la TFD d'un signal de durée infinie.

Fenêtrage (suite) :

En **Scilab**, la fonction « window » calcule une fenêtre symétrique de types différents :

win_l =window(type, n)

Type de la fenêtre:
're' : rectangulaire
'tr' : triangulaire
'hm' :hammning
'hn' :hannning

Longueur de la fenêtre

Paramètre de la fenêtre Kaiser alpha >0

vecteur de dim. 2 de paramètre de fenêtrage.

par=[dp,df])

dp (0 < dp< .5) : contrôle de la largeur de lobe principal.

df (df>0) : contrôle de la taille latérale de lobe.

Seulement la valeur de l'un de ces deux paramètres devrait être indiquée l'autre devrait placer l'égale à -1.

Autres commandes pour les fenêtres de type :

'kr' : kaiser (pseudo optimale)
'ch' : chebychef

(pour plus d'info : help window)

win_l=window('kr', n, alpha)

[win_l, cwp]=window('ch', n, par)

Fenêtrage (suite) :

La commande « window » produit un ensemble de différentes formes.

Exemple : source : ../simulation2/Scilab/sim_filtre/sim_window/fenêtres.sce

*p=7, w(i) = α + (α-1)*cos(2*%pi*i /(p-1))*, i=1, 2, …, p (longueur de la fenêtre = 7)

-->window('re',7) $\alpha = 0$, *w(i)* = 1,

ans = ! 1. 1. 1. 1. 1. 1. 1. !

-->window('tr',7)

ans = ! 0.25 0.5 0.75 1. 0.75 0.5 0.25!

-->window('hn',7) $\alpha = 0.5$, *w(i) = 0.5+0.5*cos(2*%pi*i /6)*

ans = ! 0. 0.25 0.75 1. 0.75 0.25 0.!

-->window('hm',7) *w(i) = 0.54 - 0.46*cos(2*pi*i/6)*

ans = ! 0.08 0.31 0.77 1. 0.77 0.31 0.08 !

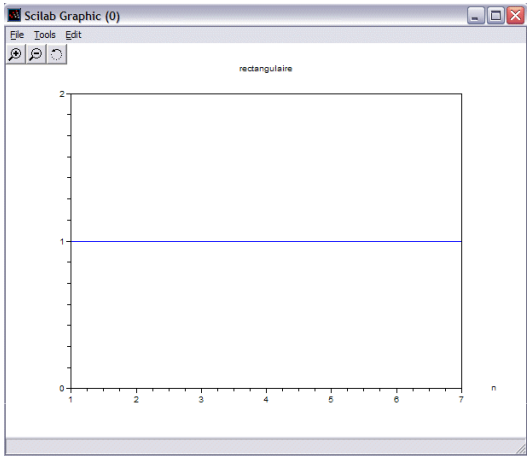
-->window('kr',7,6) $\alpha = 6$, *w(i) = α+(α-1)*cos(2*%pi*i /(p-1))*

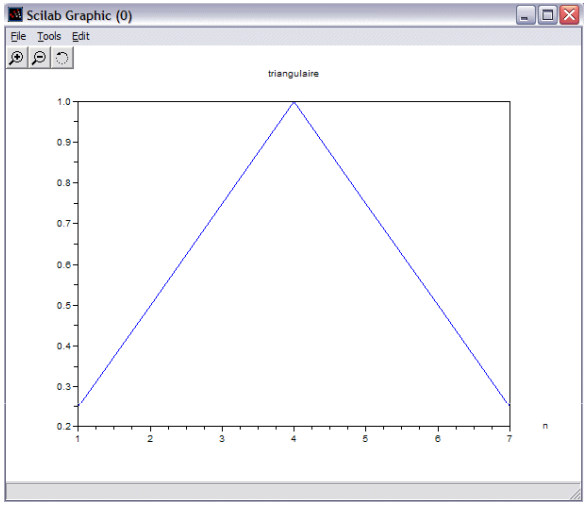
ans = ! 0.0158733 0.2537061 0.7318954 1. 0.7318954 0.2537061 0.0148733 !

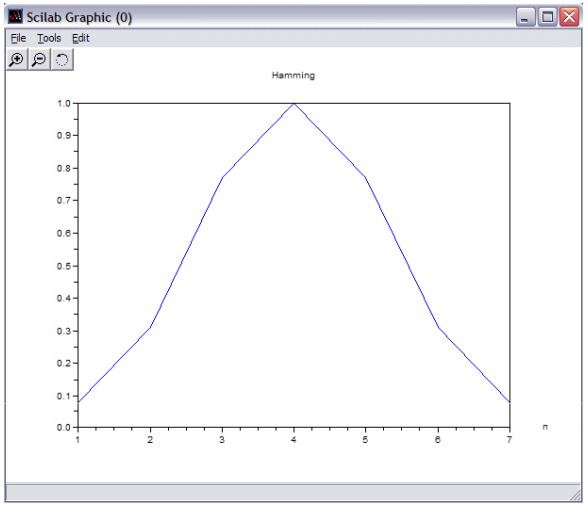
-->window('ch',7,[0.005,-1])

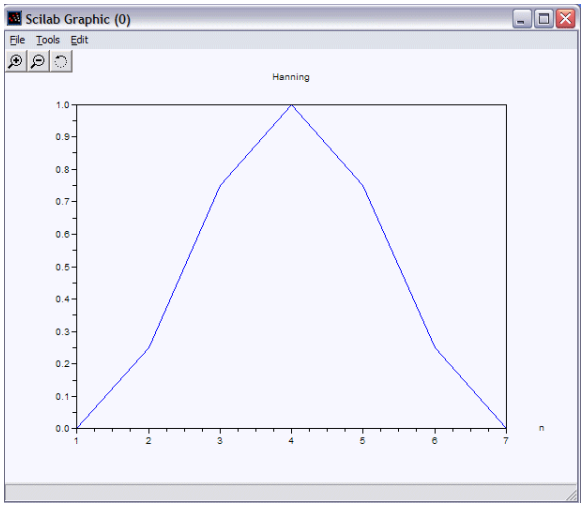
ans = !0.4402784 0.8230873 1. 1. 0.8230873 0.4402784 0.1265939!

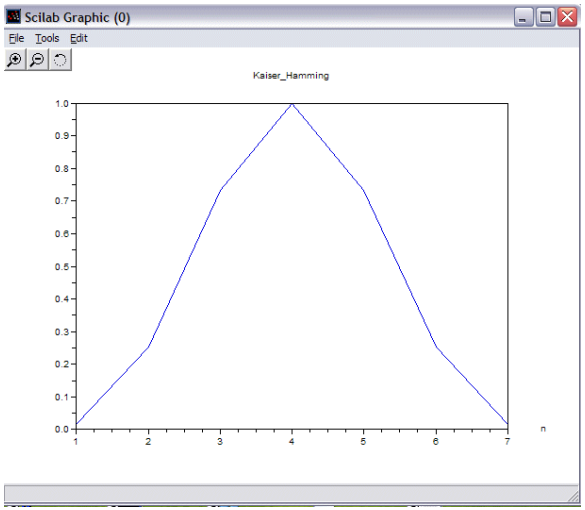
Fenêtrage (suite) : **Exemple 2.4** (suite)

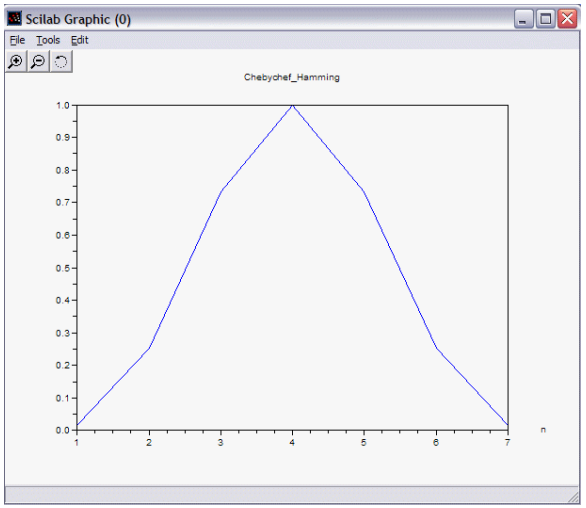












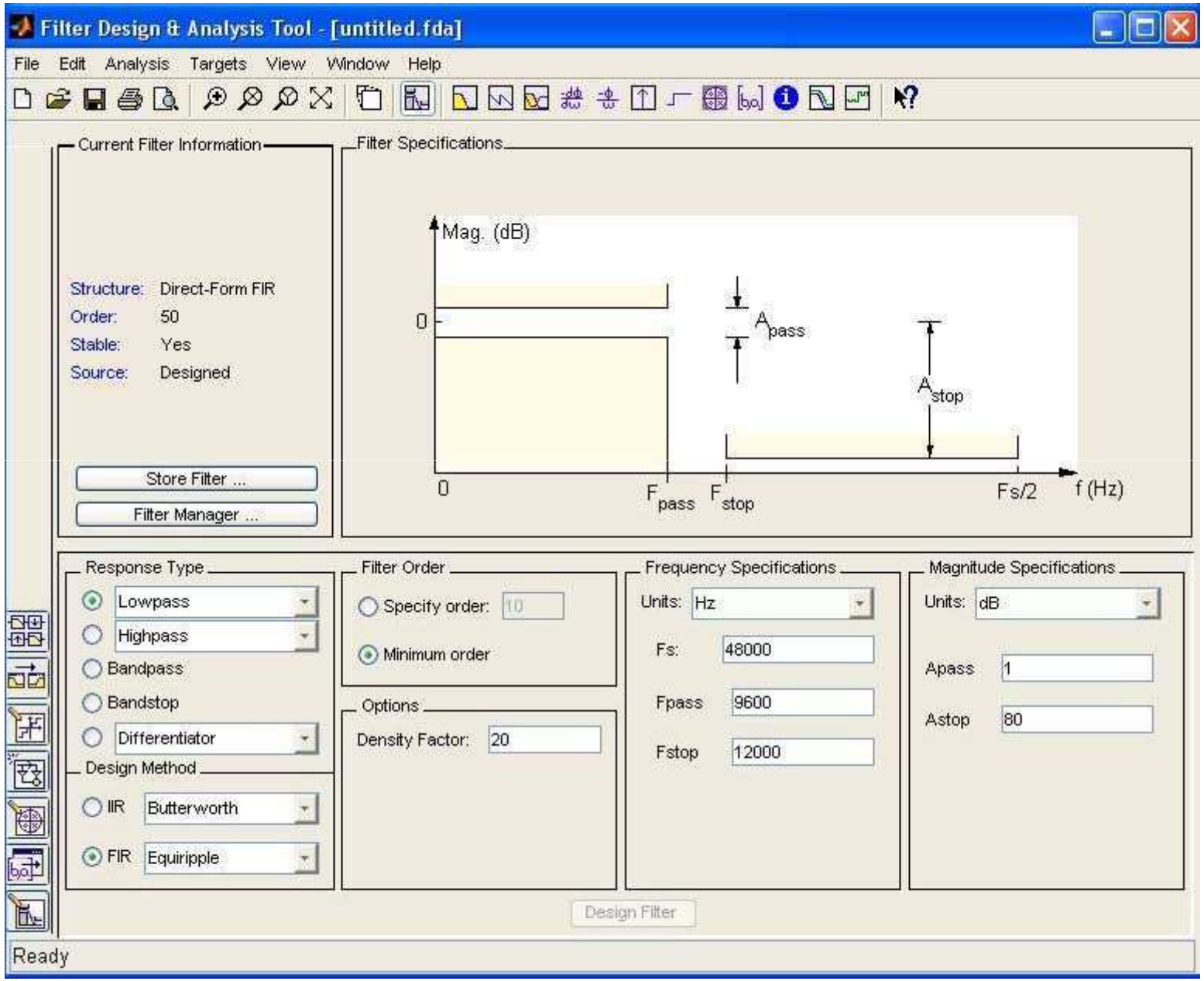
	+	-
IIR	Exige un ordre de filtre beaucoup plus faible que celui de filtre FIR afin d'atteindre un critère d'une fréquence spécifique	déphasages non-linéaires
FIR	• étant toujours stable • avoir déphasages linéaires	Moins efficace en termes de temps d'exécution et mémoire que l'*IIR* en raison de leur nature non-récurrente

- En MATLAB les filtres *IIR* et *FIR* sont implémentés dans la fonction « *filter.m* ».
- En Scilab « *iir.sci* » et « *fir.sci* » .

Voir aussi (Matlab) :

- *FIR* : *fir1*, *fir2*, *firls*
- *IIR* : *butter*, *cheby1*, *cheby2*, *ellip*, *yulewalk*
- Filter Design & Analysis GUI : *fdatool*
- Frequency Response/Filter Analysis : *freqz*, *fvtool*

>> fdatool


Filter Design & Analysis Tool - [untitled.fda]
File Edit Analysis Targets View Window Help
Current Filter Information
Structure: Direct-Form FIR
Order: 50
Stable: Yes
Source: Designed
Store Filter ...
Filter Manager ...
Filter Specifications
Mag. (dB)
0
Apass
Astop
0
Fpass
Fstop
Fs/2
f (Hz)
Response Type
Lowpass
Highpass
Bandpass
Bandstop
Differentiator
Design Method
IIR
Butterworth
FIR
Equiripple
Filter Order
Specify order: 10
Minimum order
Options
Density Factor: 20
Frequency Specifications
Units: Hz
Fs: 48000
Fpass 9600
Fstop 12000
Magnitude Specifications
Units: dB
Apass 1
Astop 80
Design Filter
Ready

Conception et application de filtres par Scilab

Si nous voulons filtrer un signal, le processus comporte deux étapes :

1. conception du filtre et
2. application du filtre au signal.

Filtrage des signaux neurophysiologiques

Avant d'aborder les signaux réels, nous allons considérer un signal simulé constitué par l'addition de deux sinusoïdes et essayer d'éliminer les composants lents.

Exemple : Filtrage des sinusoïdes

Le signal que nous emploierons est semblable à un EEG échantillonnés à 100 Hz et nous allons considérer trois secondes de signal. L'un d'entre eux semblable à un rythme alpha (un sinusoïde à 10 hertz) et l'autre semblable à une impulsion (un sinusoïde à 1 hertz avec une tension double). Nous allons concevoir un filtre pour éliminer l'impulsion et extraire le signal EEG.

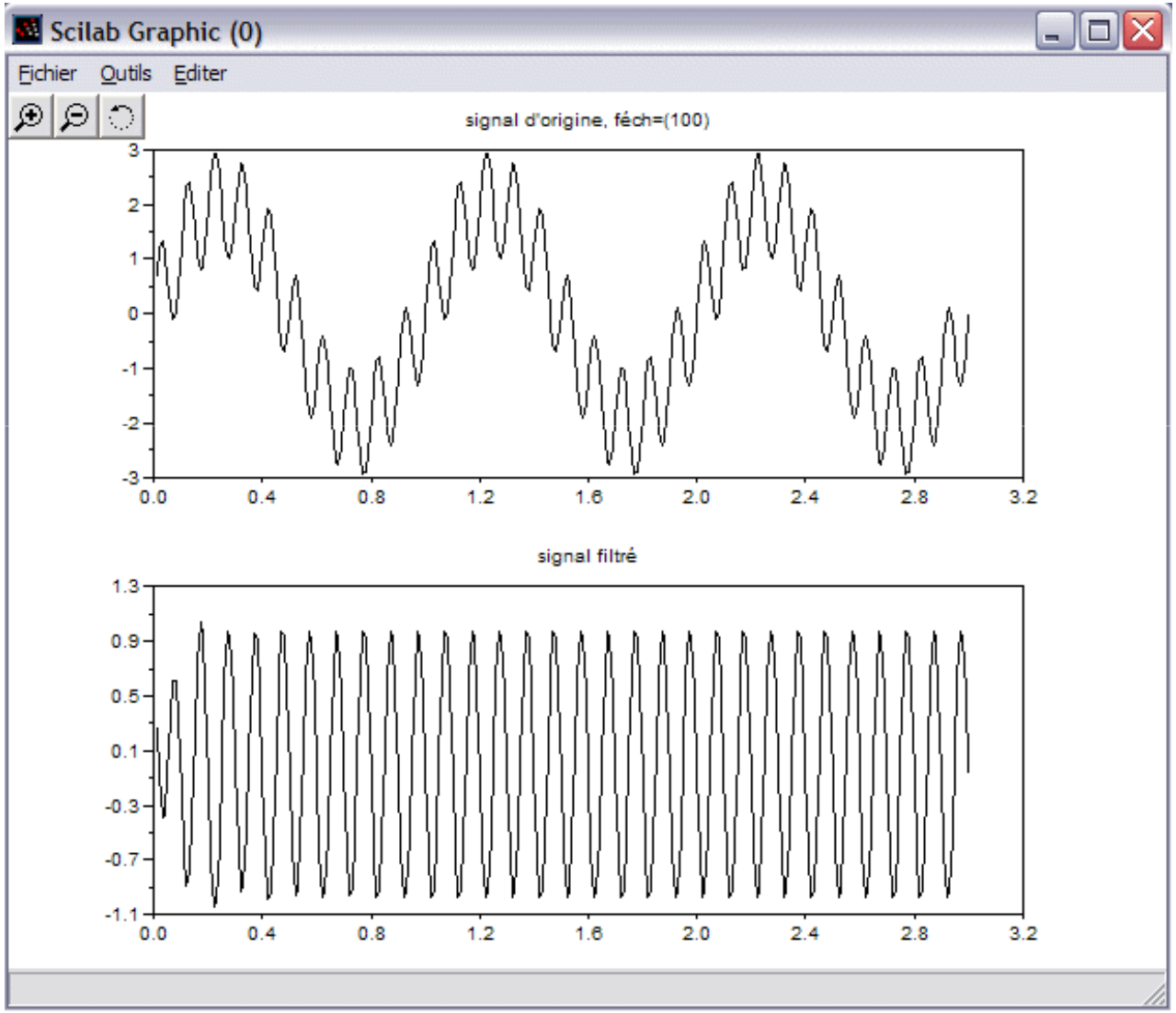
Exemple 2.2 (Scilab)

source : ../simulation2/sim_filtre/sim_iir/sim_iir.sce

```
t = [0.01:0.01:3]; // 300 échantillons (fréquence d'échantillonnage = 100 Hz)
// additionner les deux signaux
eeg = sin(2*%pi*10*t);
impulsion = sin(2*%pi*1*t);
inp = eeg + 2 * impulsion;

// Conception du filtre : utiliser un filtre avec n=10
// puisque nous allons extraire le sinusoïdale à 10 Hz et éliminer le sinusoïde à 1 Hz, dans ce
// cas nous avons un filtre passe haut ('hp' ) avec un filtre de type Betterworth ( 'butt' ).
// L'étape la plus difficile est de distinguer la fréquence de coupure. Puisque nous allons
// utiliser des signaux échantillonnés, les fréquences de coupure possibles couvrent les
// valeurs de 0 à 0.5. Un sinusoïde à une fréquence de 10 Hz avec une fréquence
// d'échantillonnage de 100 Hz correspond à une fréquence de coupure de (10/100 = 0.1), un
// sinusoïde à une fréquence de 1 Hz échantillonné à 100 Hz correspond à une fréquence
 // de coupure de (1/100 = 0.01). Ainsi, pour distinguer le signal eeg parmi ces signaux nous
// pouvons choisir une fréquence de coupure de 0.05 équivalent à un signal de fréquence 5
// Hz (100*0.05).
sys = iir(10,'hp','butt',[0.05 0],[0 0]);
out = flts(inp, sys);// réponse temporelle
```

Exemple 2.2 (suite)

```
xsetech([0,0,1,1/2]);
plot2d(t,inp);
xtitle('signal d’origine');
xsetech([0,1/2,1,1/2]);
plot2d(t,out);
xtitle('signal filtré');
```

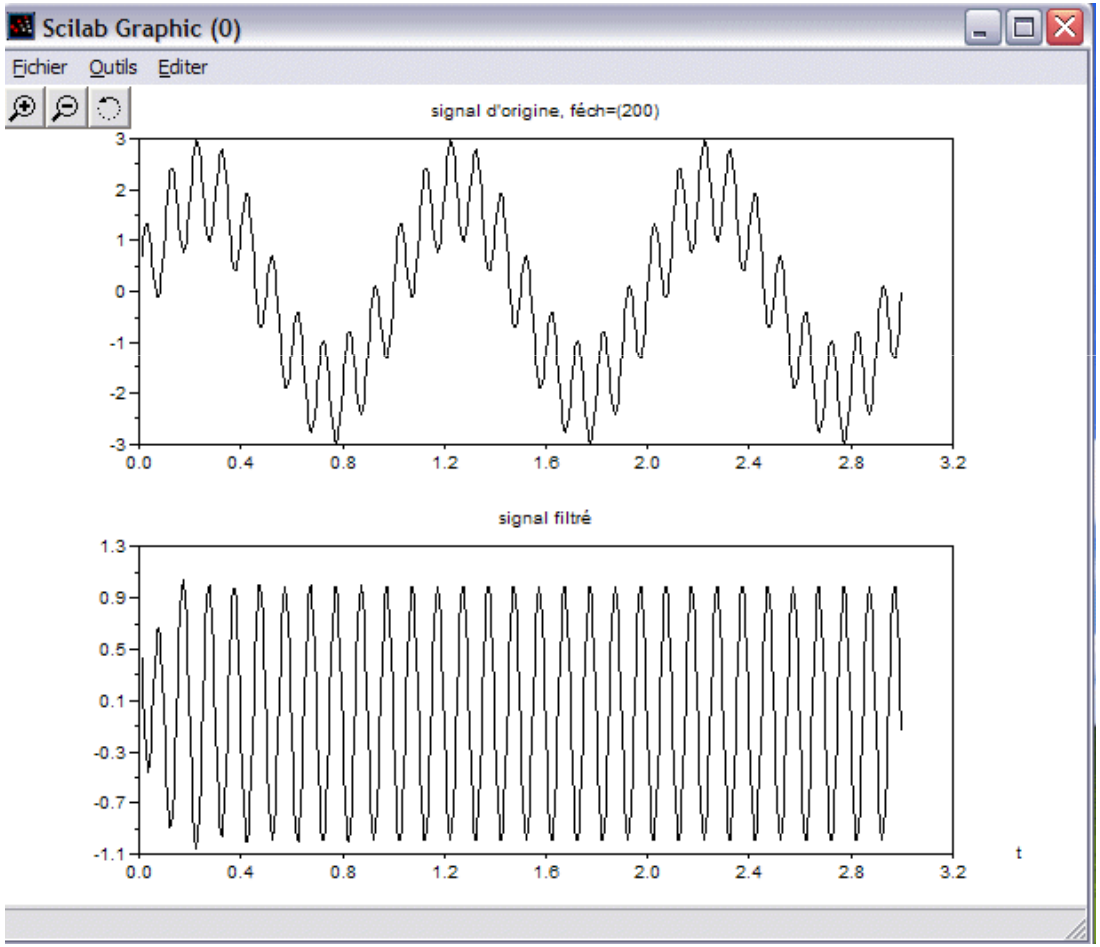


Maintenant, imaginez que vous avez échantillonné le même signal à 200 Hz (période d'échantillonnage 0.005).

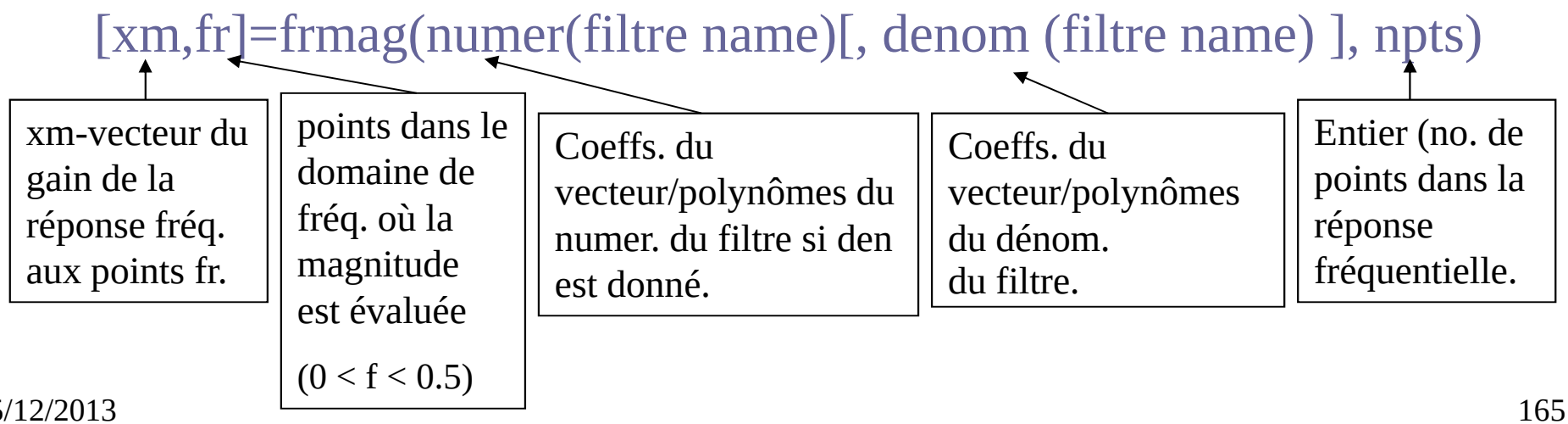
// **Conception du filtre** : utiliser un filtre avec n=10
// puisque nous allons extraire le sinusoïdale à 10 Hz et éliminer le sinusoïde à 1 Hz, dans ce
// cas nous avons un filtre passe-haut ('hp') avec un filtre de type Betterworth ('butt').
// Un sinusoïde à une fréquence de 10 Hz avec une fréquence d'échantillonnage de 200 Hz
// correspond à une fréquence de coupure de 0.05 (10/200), un sinusoïde à une fréquence de
// 1 Hz échantillonné à 200 Hz correspond à une fréquence de coupure de 0.005 (1/200).
// Ainsi, pour le distinguer parmi ces signaux nous pouvons choisir une fréquence de
// coupure de 0.025 équivalent à un signal de fréquence 5 Hz (200*0.025).

```
sys = iir(10,'hp','butt',[0.025 0],[0 0]);
out = flts(inp,sys);
```

Le résultat est la même.

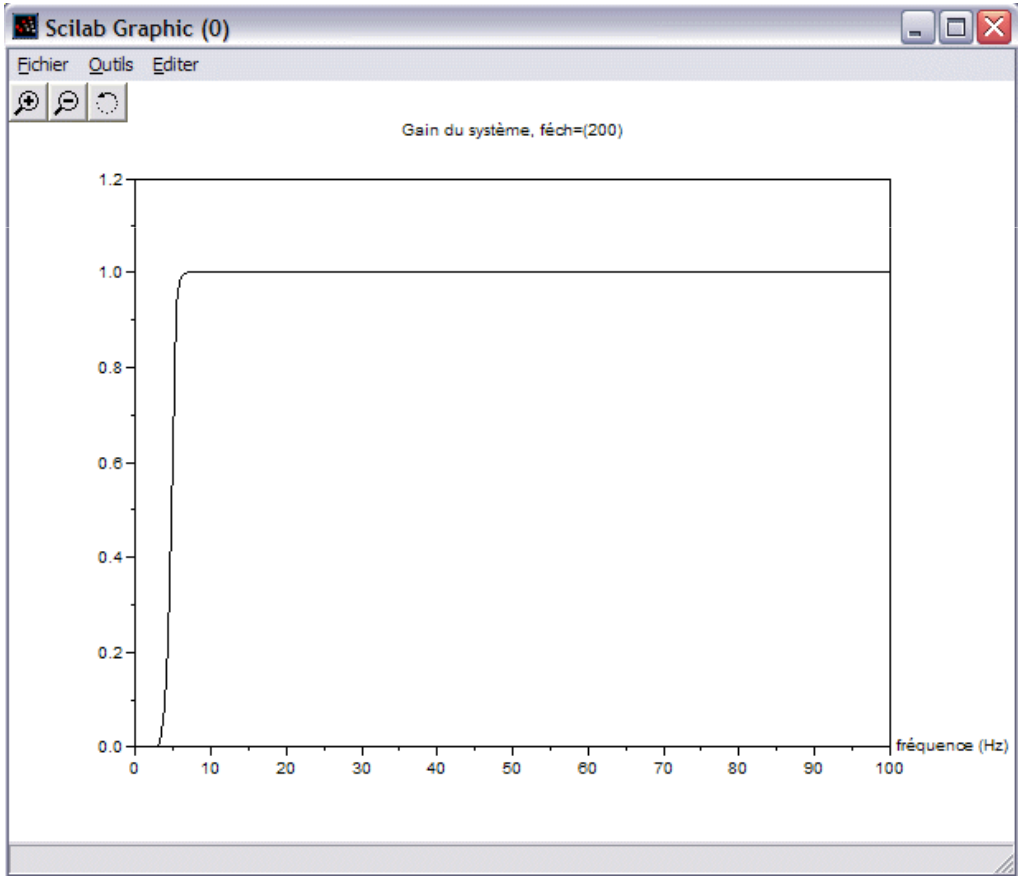


Nous avons conçu un système agissant en tant qu'un filtre passe-haut avec une fréquence de coupure de 5 Hz et nous sommes intéressés à évaluer comment le système se comporte avec différentes fréquences. Pour cela, nous utiliserons la fonction «frmag» qui calcule le gain (gain= magnitude_sortie/magnitude_entrée) des réponses fréquentielles du filtre. La description de filtre peut être un ou deux vecteurs de coefficients, un ou deux polynômes, ou un polynôme raisonnable.



```
[famp, f] = frmag(numer(sys), denom(sys), 256);
xsetech([0,0,1,1]); xbasc();
//  Pour évaluer la sortie à la fréquence d'échantillonnage employée nous pouvons
// multiplier ‘ f  ' par la fréquence d'échantillonnage utilisée, dans ce cas-ci 200 Hz.
plot2d(f*200,famp); xtitle('Magnitude du système','fréquence (Hz)');
```

- f < 2.5 Hz sont complètement éliminées
- f > 7.5 Hz sont inchangées



Propriété du filtre (exemple 2.2 (suite))

Une représentation différente de la réponse temporelle est le « diagramme de Bode ». Ce diagramme exprime l'information sur l'amplitude et la phase de la réponse en unités logarithmiques : bode(sys);

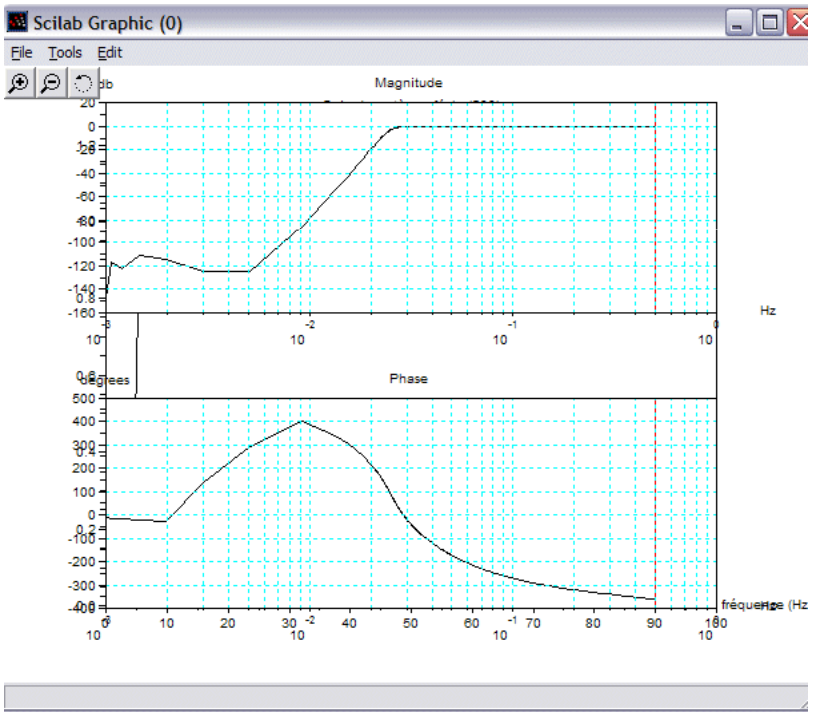
• **En haut** : échelle logarithmique : fréquences sont exprimées comme puissance de 10 et couvrent de 0.001 à 1 (l'axe-x est équivalent à fr= 0.001, 0.01, 0.1 et 1).

• f > 0.5 n'existe pas. Puisque féch=200 Hz, les valeurs sont obtenues par f =fr*féch : 0.2, 2 et 20 Hz.

• L'amplitude en décibel (db).

dbv = 20 * log10 (decv) // conversion valeurs
// déc en db
decv = 10^(dbv) // conversion valeurs db en déc.

• **En bas** : modification de phase introduite en degrée par le filtre.



Puisque la modification dans la phase peut changer la forme du signal, le filtrage peut avoir un effet profond sur certaines mesures faites sur le signal.

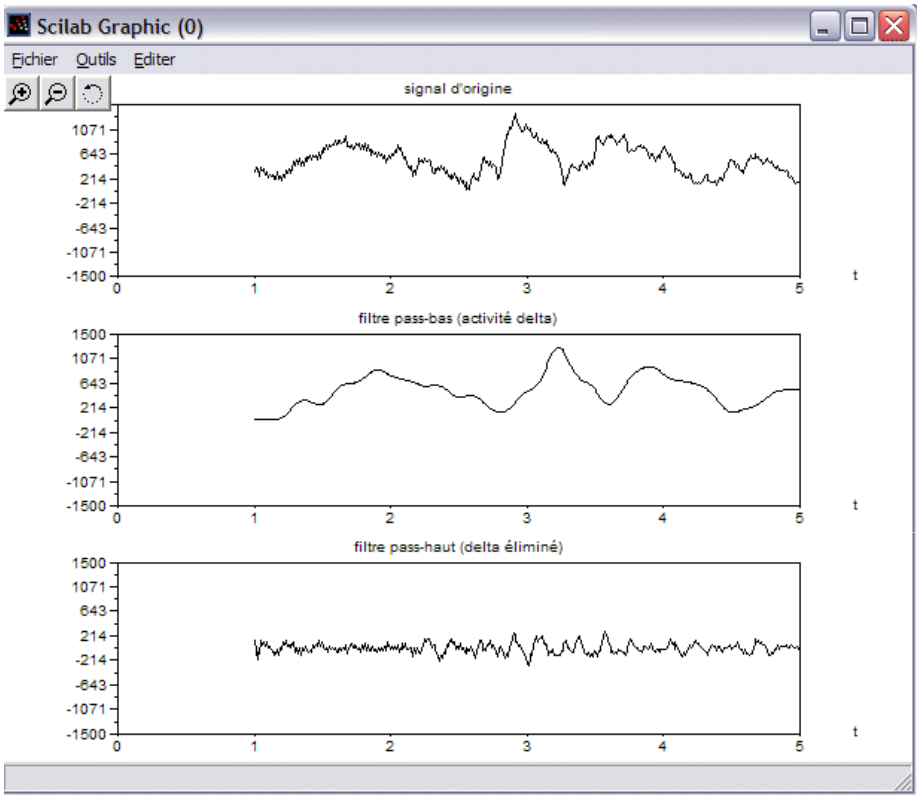
Exemple Filtrage d’un signal réel

Employer le premier signal EEG « Fpz-M2 » contenu dans « signals » (la liste Scilab que nous avons sauvée dans la première partie de ce cours au sujet du format EDF) qui est échantillonné à 100 Hz. L’objectif est d’extraire les activités liés aux ondes delta* de fréquence inférieure ou égale à 4 Hz.

* L‘onde delta est caractérisée par une fréquence inférieure ou égale à 4 Hz. Elle tend à être la plus haute en amplitude et la plus lente. Elle est tout à fait normal et elle est le rythme dominant chez les enfants âgés de moins d’un an et se trouve dans les étapes 3 et 4 de sommeil. L’onde delta est anormale chez l'adulte éveillé.

Exemple 2.3 Filtrage d'un signal réel (suite)

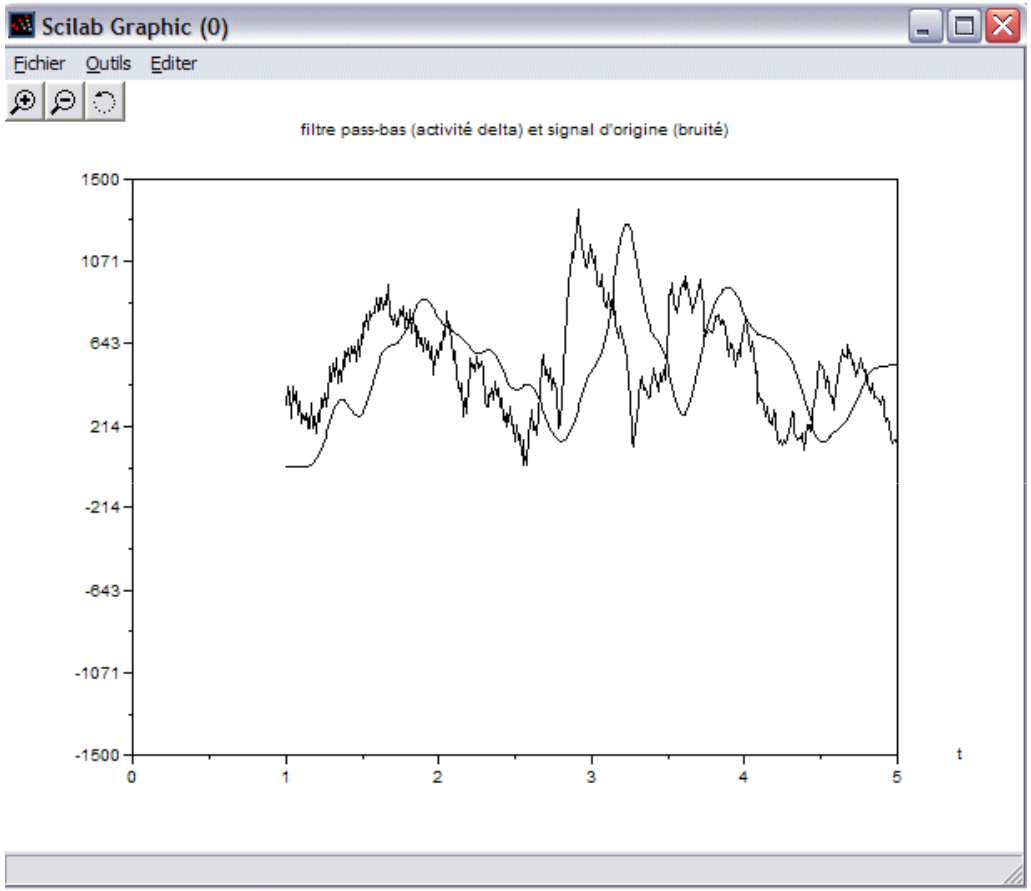
source : ../simulation2/sim_eeg_filtre/eeg_delta_filtre.sce




```
stacksize(20000000)
load(path+'signals');
eeg1 = signals(1);
t = [1:0.01:length(eeg1)];
// conception d'un filtre  'lp'
// fréq. de coupure=4/100=0.04
sys1 = iir (10,'lp','butt',[0.04 0],[0 0]);
// appliquer le filtre  'lp'
resul1 = flts(eeg1,sys1);
// conception d'un filtre  'hp'
// fréq. de coupure=4/100=0.04
sys2 = iir (10,'hp','butt',[0.04 0],[0 0]);
// appliquer le filtre 'hp'
resul2 = flts(eeg1,sys2);
xsetech([0,0,1,1/3]);
plot2d(t(1:500),eeg1(1:500),1,"011"," ",[0,-1500,5,1500]);
xtitle ('signal d'''origine'); xsetech([0,1/3,1,1/3]);
plot2d(t(1:500),resul1(1:500),1,"011"," ",[0,-1500,5,1500]);
xtitle ('filtre pass-bas');
xsetech([0,2/3,1,1/3]);
plot2d(t(1:500),resul2(1:500),1,"011"," ",[0,-1500,5,1500]);
xtitle('filtre pass-haut','t');
```

Corriger les déformations

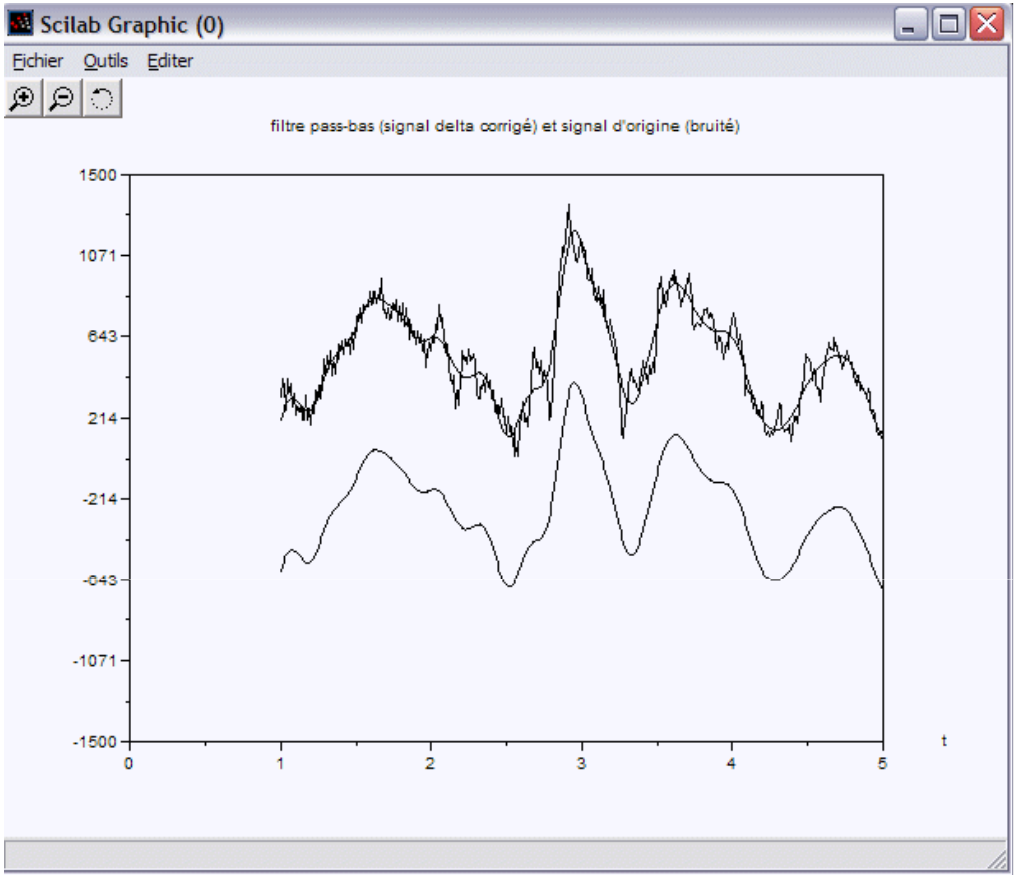
Nous sommes non seulement intéressés par les fréquences mais aussi par le maintien de la forme d'origine. Mais le filtrage peut modifier la forme du signal parce qu'il modifie la phase de différentes manières selon la fréquence.



Corriger les déformations

Puisque la déformation vient de la modification présentée dans la phase, nous pouvons l'éliminer en filtrant le résultat à partir de l'extrémité du signal jusqu'à son début. Le processus est un peu difficile

1. inversez le résultat
2. filtrez le signal avec le même filtre (les amplitudes ne seront pas très excessivement modifiées puisque le signal a été précédemment filtré, mais la modification dans la phase présentée par le filtre sera compensée).
3. inversez le signal.

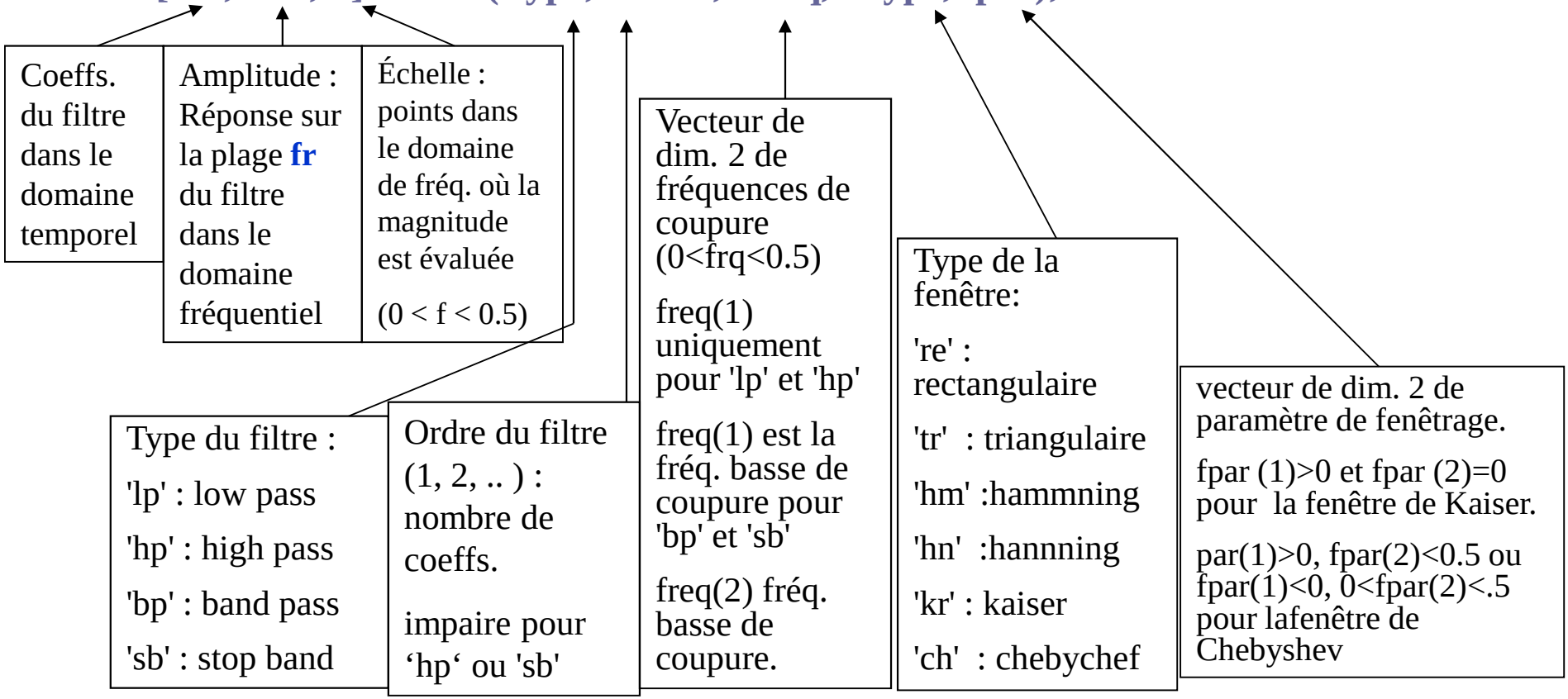


Trois étapes :

1. Concevoir du filtre Finite Impulse Response (FIR) en Scilab

// fonction pour la conception d'un filtre **wfir** en utilisant les des filtres analogiques

[wft,wfm,fr] = wfir (ftype, forder, cferq, wtype, fpar);



2. Construire la fonction de transfert en utilisant les coefficients « **wft** »

3. Appliquer le filtre « flt »

Exemple 2.5 : source : ../simulation2/sim_filtre/sim_fir/sim_fir.sce

```
// premier étape
          [wft,wfm,fr] = wfir ('lp', 4, [0.2  0], 'hm', [0  0]); // assayez wft=wfir()
wft =
! 0.0161456        0.2881307        0.2881307        0.0161456 !

// deuxième étape
 hpoly = poly (wft, 's','coeff'); //définition du polynôme du système à temps continu

hpoly =
0.0161456 + 0.2881307s + 0.2881307s^2 + 0.0161456s^3

hz = horner (hpoly, (1/%z)); // évalue le polynôme ou la fraction rationnelle (ou
                             // matrice de fractions rationnelles)  hpoly quand la
                             // variable  s  du polynôme est remplacée par  1/z

hz =
0.0161456 + 0.2881307z + 0.2881307z^2 + 0.0161456z^3
---------------------------------------------------------------------
                                  z^3
```

Exemple 2.5 (suite)

// deuxième pas (suite)

sys = syslin ('**d**', **hz**); // définition d'un système dynamique linéaire discret, création
// d'une liste

sys =

$$\frac{0.0161456 + 0.2881307z + 0.2881307z^2 + 0.0161456z^3}{z^3}$$

// **Troisième étape**

in = rand(**1:256**);

out = flts (**in**,**sys**);

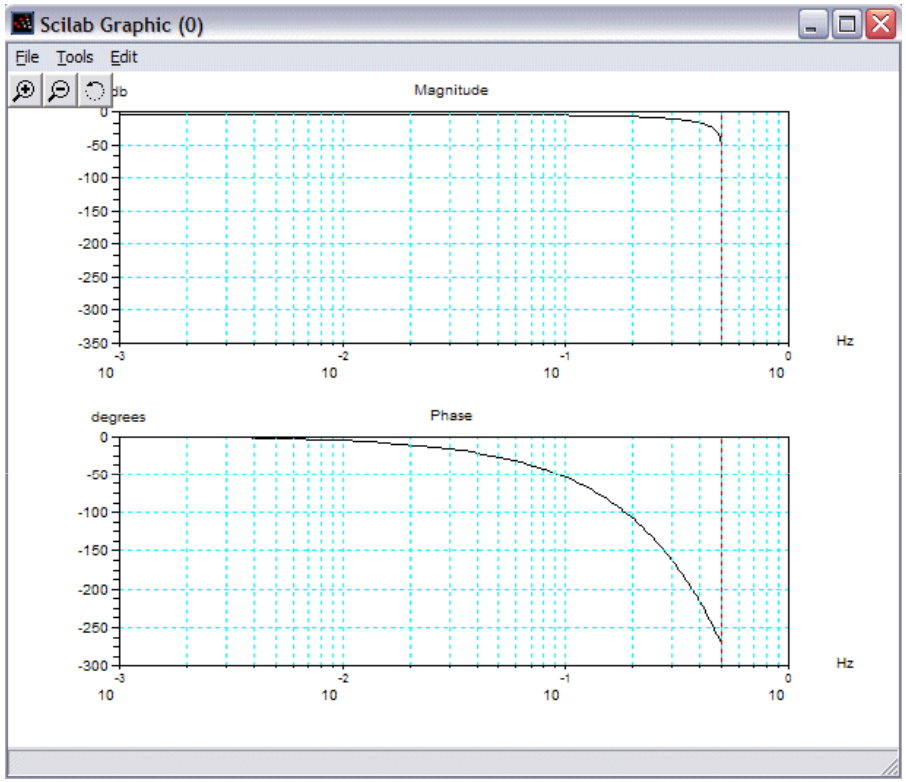
// Une fois que nous définissons le système linéaire, nous pouvons faire beaucoup
// des choses telles que ajouter ou soustraire avec d'autres systèmes linéaires. Nous
// pouvons découvrir les caractéristiques du filtre que nous avons fait :

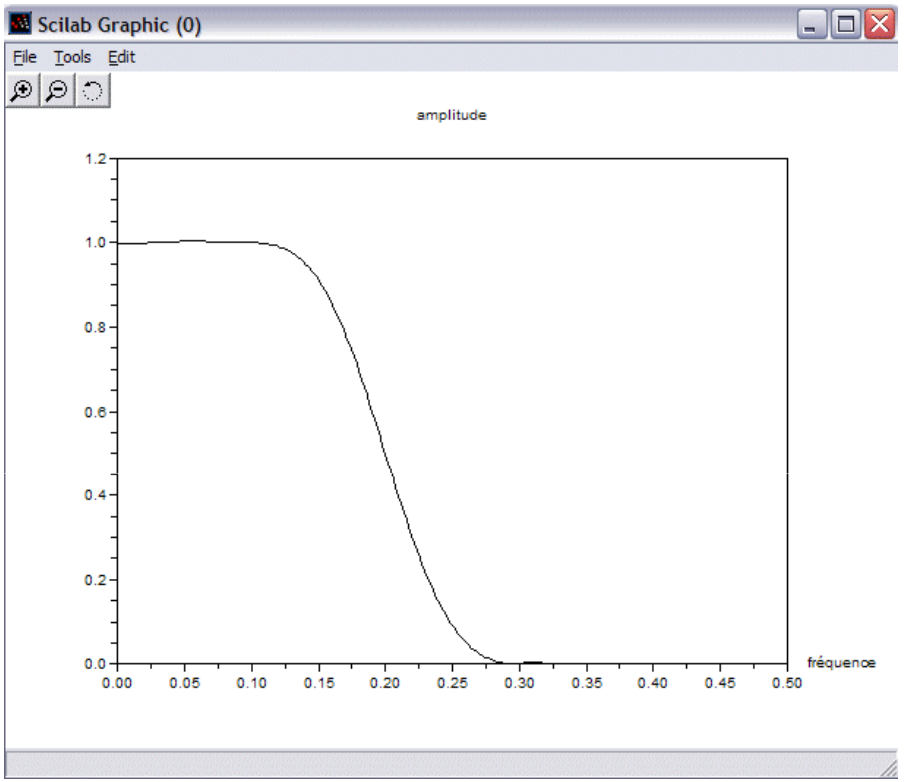
bode(**sys**);

[**wft**,**wfm**,**fr**] = wfir ('**lp**', **20**, **[.2 0]**, '**hm**', **[0 0]**);

plot2d(**fr, wfm**); **xtitle('amplitude','fréquence');**

Exemple 2.5 (suite)





Contrairement au filtre IIR, le filtre FIR a une phase linéaire, qui implique que le filtre ne change pas la forme d'origine du signal. Cette caractéristique est importante quand la forme de signal est considérée dans l'interprétation de données. En Neurophysiologie clinique, nous sommes habituellement très intéressés à maintenir la forme du signal.

4. Analyse spectrale

Pourquoi le domaine fréquentiel ?

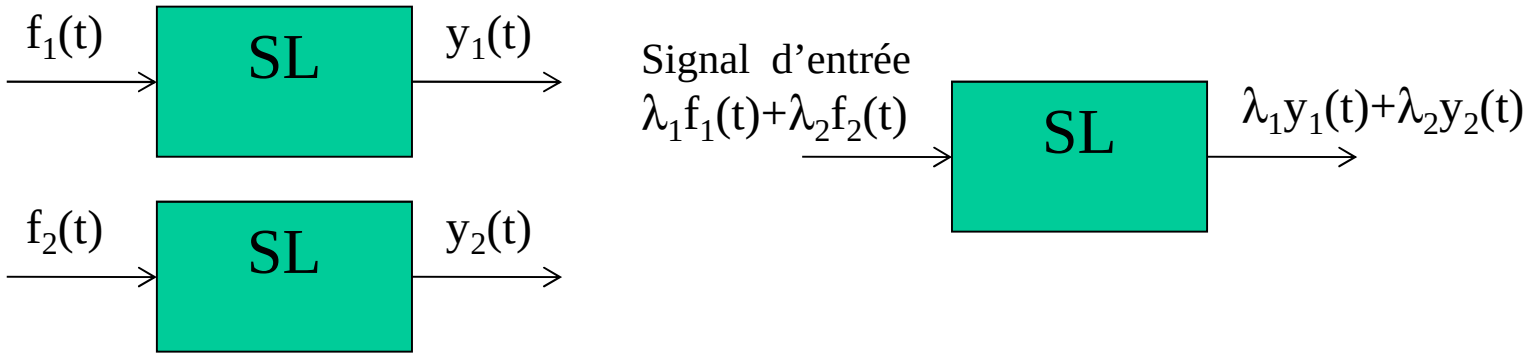
- Pour les systèmes physiologiques, l'information utile est souvent plus facilement exprimée dans le domaine fréquentiel.

- Le bruit et l'information utile sont souvent plus faciles à séparer dans le domaine fréquentiel que dans le domaine temporel.

- Certains algorithmes sont plus efficaces lorsqu'elles sont appliquées dans le domaine fréquentiel, par exemple calcul de la convolution dans le domaine fréquentiel.

Une méthode utilisée pour déterminer le spectre d'un phénomène (signal) observé.

Il existe une classe importante de systèmes, les **systèmes linéaires** (ou supposés tels) régis par le principe de **superposition**. Dans ce cas, correspondant à une **équation différentielle linéaire**, on peut essayer de décomposer le signal d'entrée en une somme de signaux simples auxquels on saurait faire correspondre des signaux de sortie également simples dont la somme donnerait le résultat cherché.



Le problème se simplifie encore plus si les caractéristiques du système restent constantes au cours du temps. On a affaire à une **équation différentielle linéaire à coefficients constants** (**système stationnaire**).

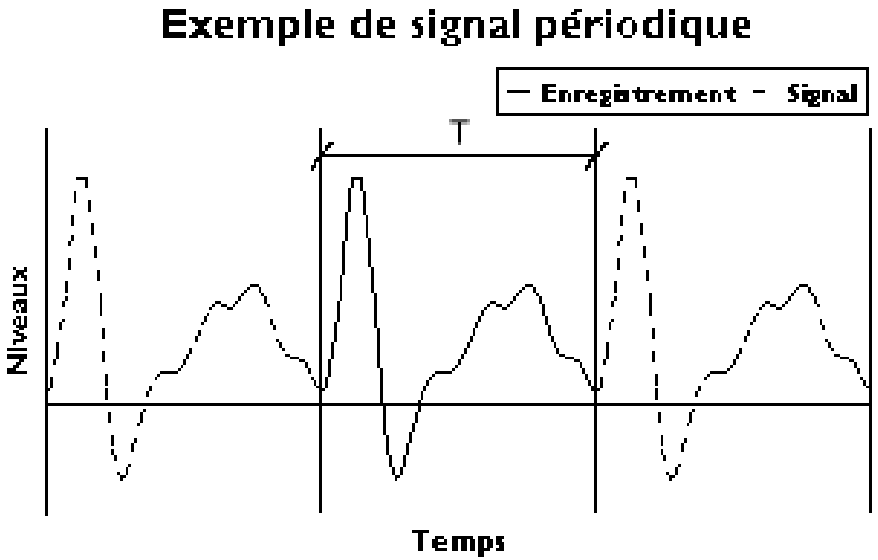
Les signaux simples sont les sinusoïdes qui subissent uniquement une **amplification** et un **déphasage**.

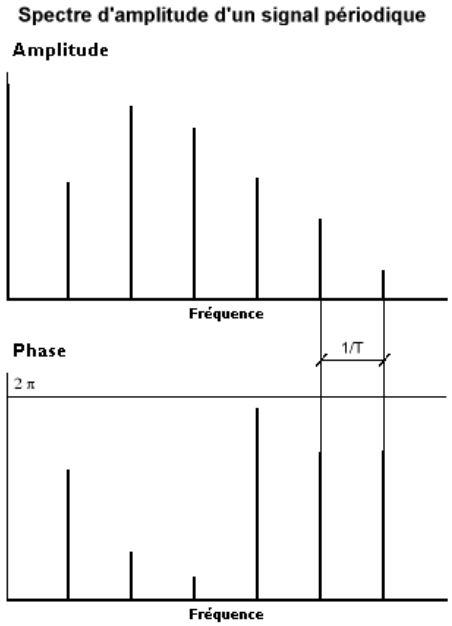
Le problème de l'**analyse spectrale** : décomposer un signal complexe en une somme de sinusoïdes.

Difficulté : cette décomposition exige que le signal soit défini sur un temps infini. Or un signal réel ne peut être connu qu'à travers un enregistrement de durée limitée : il faut donc construire un modèle du signal en faisant des hypothèses, souvent évidentes et intuitives, sur la partie non enregistrée du phénomène.

Différents modèles

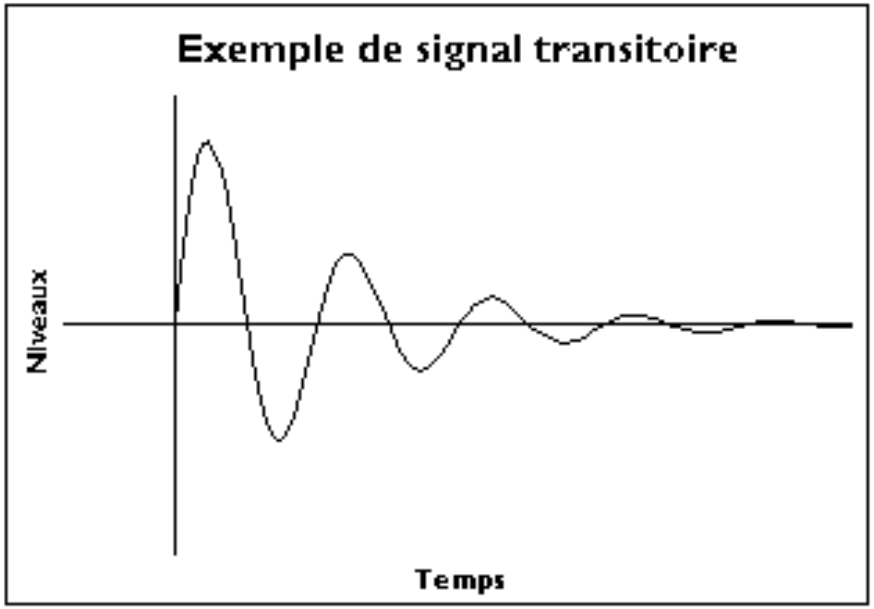
- On peut supposer, par exemple, que le signal reproduit indéfiniment le contenu de l'enregistrement : on construit alors **un modèle périodique** basé sur la **série de Fourier**. Le signal est décrit par un **spectre discret** (ensemble de fréquences en progression arithmétique).

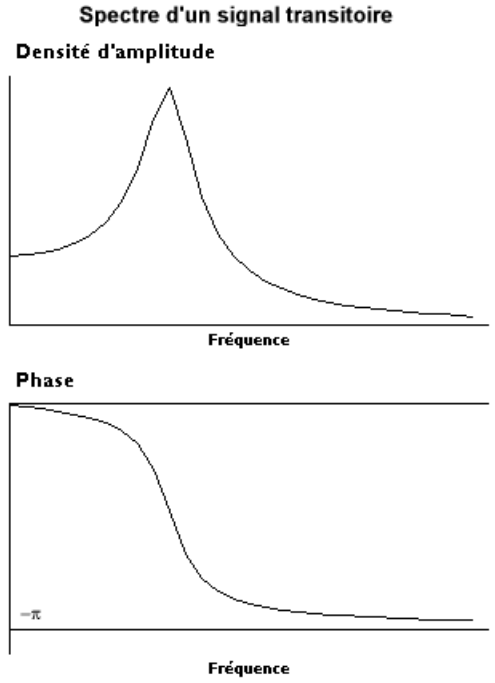




Différents modèles (suite)

- On peut aussi supposer que le niveau du signal est négligeable en dehors de l'enregistrement : on utilise dans ce cas un **modèle transitoire** basé sur la **transformation de Fourier** qui conduit en général à un **spectre continu**.





Différents modèles (suite)

- Il existe un certain nombre de phénomènes naturels pour lesquels aucune de ces deux hypothèses n'est réaliste. Par exemple, un enregistrement de signaux, qui **ne montre pas de périodicité** et **ne montre pas non plus de décroissance nette** au cours de sa durée qui peut être relativement faible.

  On peut alors utiliser une hypothèse un peu plus floue selon laquelle **la moyenne quadratique calculée sur l'enregistrement fournit une estimation raisonnable de la moyenne quadratique réelle du signal**.

Différents modèles (suite)

Ce type d'analyse conduit encore à un spectre continu. Ce spectre se définit, comme les précédents, à partir du signal mais on peut obtenir des informations supplémentaires en considérant celui-ci comme une **réalisation d'un processus aléatoire**.

Signaux périodiques

Le développement en série de Fourier d'un enregistrement de durée T associe à celui-ci des sinusoïdes d'amplitudes finies et de fréquences multiples de la fréquence du fondamental 1/T. On parle d'un **spectre d‘amplitude** qui est un spectre de raies. Dans le cas général, le résultat de l'analyse peut s'exprimer soit en **amplitudes** et **phases**, soit en **composantes cosinus** et **sinus**.

Signaux transitoires

Ici, on raisonnera d'abord sur le signal de **durée supposée infinie** avant de voir les conséquences pour un enregistrement de durée finie. Si ce signal n'est pas périodique, c-à-d n'a pas de période finie, on peut essayer de voir ce qui se passerait si on lui prêtait une période infinie. Cela entraîne les conséquences suivantes :

- Lorsque la durée d'analyse T tend vers l'infini, le pas de fréquence 1/T tend vers 0 : à la limite on passe d'un spectre discret à un spectre continu.

Signaux transitoires (suite)

Au cours de cette croissance de la durée d'analyse, lorsque celle-ci est multipliée par un nombre n quelconque, le nombre de composantes est multiplié par le même facteur. Pour que le niveau du signal ne soit pas augmenté dans les mêmes proportions il faut que les amplitudes des composantes soient, en gros, divisées par n, ce qui risque de conduire à la limite à des amplitudes nulles.

Signaux transitoires (suite)

On pare à cette difficulté en multipliant les coefficients de Fourier par la longueur d'analyse ou en les divisant par le pas de fréquence qui tend vers zéro. Ainsi, le spectre continu n'est plus un spectre d'amplitude mais un **spectre de densité d'amplitude** dont l'unité est unité physique / Hertz.

Signaux transitoires (suite)

Malgré ces précautions, la méthode peut diverger. Une condition de convergence est que le signal doit être transitoire : il doit tendre vers 0 lorsque le temps tend vers $\pm\infty$.

On obtient ainsi la transformée du signal x(t) que l'on note généralement x(f), f étant la fréquence.

Signaux transitoires (suite)

Si on retourne à un enregistrement de **durée limitée**, il y a deux possibilités :

1. Le **signal a des valeurs non nuls sur cette durée limitée** d'enregistrement : l'analyse pendant cette durée fournit, en principe, un résultat exact permettant de reconstituer le signal par inversion de la transformation.

Signaux transitoires (suite)

2. Le signal a des **valeurs non nuls sur une durée supérieure à celle de l'enregistrement** : l'imprécision du résultat croît avec la quantité d'information perdue. L'erreur ainsi commise se traduit concrètement par une dispersion des énergies correspondantes à certaines fréquences sur les énergies de fréquences voisines et mathématiquement par la notion de convolution.

4.1. Transformée de Fourier

Introduction

La **Transformée de Fourier** peut être vue comme un outil qui convertit un signal en somme de sinusoïdes de différentes fréquences, amplitudes et phases. Notre but est de comprendre ce que nous obtenons quand nous appliquons la **Transformée de Fourier** à un signal et comment relier cette information avec les caractéristiques du signal original.

Définition mathématique en temps continu

Un signal déterministe quelconque $X(t)$ non périodique peut toujours être considéré comme un signal périodique dont la période T tend vers l'infini. Sous ces hypothèses la transformée de Fourier est définit par

$$\text{Transformée de Fourier Direct} : X(f) = \int_{-\infty}^{+\infty} x(t) e^{-2\pi jft} dt$$

$$\Updownarrow$$

$$\text{Transformée de Fourier Inverse} : x(t) = \int_{-\infty}^{+\infty} X(f) e^{+2\pi jft} df$$

Propriétés de la TF

Considérons 3 signaux $x(t)$, $y(t)$ et $z(t)$

a un nombre réel quelconque > 0

b, c deux nombres complexes quelconques.

t_0, f_0 deux nombres réels quelconques

1. ***Linéarité*** : $z(t) = bx(t)+cy(t) \Rightarrow Z(f) = b \cdot X(f)+c \cdot Y(f)$
2. ***Translation*** : $z(t) = x(t - t_0) \Rightarrow Z(f) = exp(-2\pi i\, t_0\, f)X(f)$
3. ***Modulation*** : $Z(t) = exp(2\pi f_0)x(t) \Rightarrow Z(f) = X(f - f_0)$
4. ***Mise à l'échelle*** : $z(t) = x(at) \Rightarrow Z(f) = \frac{1}{|a|} X(\frac{f}{a})$

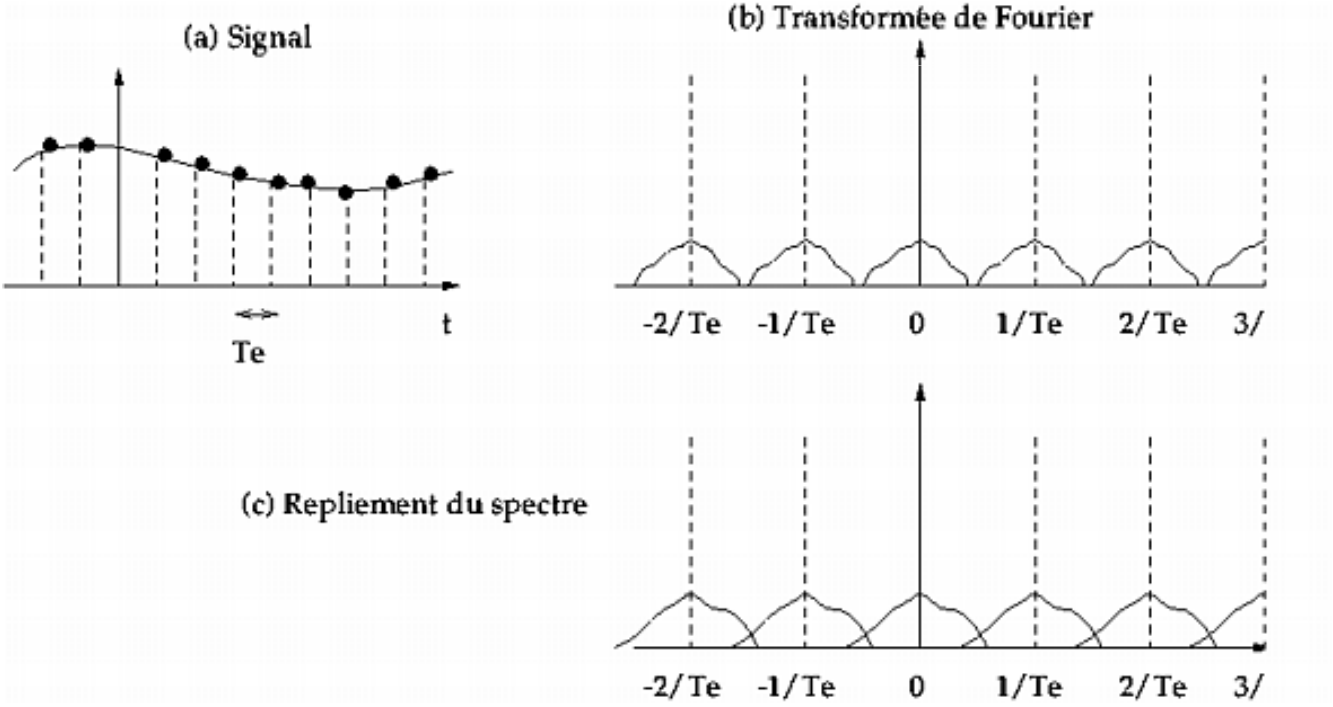
5. ***Conjugaison*** **:** $z(t) = x(t)^* \Rightarrow Z(f) = X(-f)^*$
6. **Théorème de convolution** :

$y(t) = x(t) * z(t) \Leftrightarrow Y(f) = X(f).Z(f)$

$Y(f) = X(f) * Z(f) \Leftrightarrow y(t) = x(t).z(t)$

Effet de l'échantillonnage

$x(f)$ est donc périodique de période $1/T_e$. Afin d'éviter que le spectre ne se replie pas par rapport aux axes de symétrie (voir figure), il est nécessaire que le signal ne comporte aucune fréquence supérieure à $f_e/2$ Hz.



Considérations pratiques

Transformée de Fourier Discrète à temps fini (TFD)

Soit x un signal échantillonné à une fréquence f_e Hz (échantillon/sec). La période d'échantillonnage est alors $T_e=1/f_e$.

Le processus d'échantillonnage peut être écrit

$$x[n]=x(nT_e)=x(t)\delta(t-nT_e)$$

Soit N le nombre total d'échantillons ($x=[x[1]\ x[2]\ \ldots\ x[N]$). Le $n^{ième}$ échantillon se produit au temps $t[n] = nT_e = n/f_e$. Alors la TFD peut être obtenue comme suit :

$$X(k)=\sum_{n=0}^{N-1} x[n]e^{-j\frac{2\pi kn}{N}}$$

Chaque composante de fréquence $f_k = kf_e/N$, k = 0, 1, 2, 3, …, N-1.

Transformée de Fourier Inverse Discrète (TFID)

$$x[n]=\frac{1}{N}\sum_{n=0}^{N-1} X(k)\,\mathrm{e}^{j\frac{2\pi kn}{N}}$$

Transformée de Fourier Rapide (TFR)
(Fast Fourier Transform (FFT))

La TFR est une méthode qui permet d'accélérer de façon particulièrement spectaculaire les calculs de la TFD.

Selon cette méthode, nous choisissons un nombre d'échantillons $N = 2^{\gamma}$, où γ est un entier.

Transformée de Fourier Rapide (TFR)
(Fast Fourier Transform (FFT))

- Diviser la séquence initiales de N points en 2 séquences de N/2 points,
- Faire le calcul de 2 TFD sur des séquences de N/2 points
- Continuer jusqu'à ce que les transformées de Fourier élémentaires ne porte que sur 2 points.

Deux méthodes d'arrangement des données sont utilisables :

1. La méthode d'entrelacement temporel et
2. la méthode d'entrelacement fréquentiel.

Transformée de Fourier Rapide (TFR)
(Fast Fourier Transform (FFT))

Le facteur F de réduction du volume des calculs, est donné par : $F=\gamma/N$.

Par exemple, pour un nombre d'échantillons N de 1024 ($N= 2^{10}$), nous obtenons F=0.01.

Cette méthode est implémentée dans tous les logiciels modernes de traitement du signal.

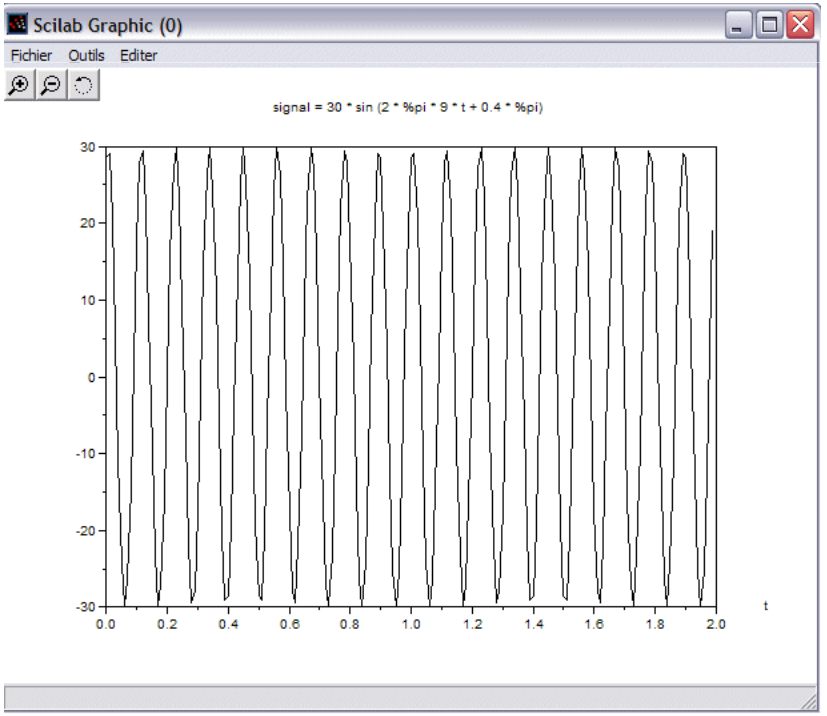
Exemple 3.1

Nous allons échantillonner un sinusoïde de fréquence 9 Hz (Ts = 1/9 s) et d'une amplitude 30 (v ou mv ou autres) à 100Hz (Tech=0.01) et nous allons considérer un segment de 2 sec (18 cycles).

Exemple 3.1 (suite)

source : ../simulation2/sim_filtre/sim_fourier/sim_fourier.sce

```
inct = 0.01;
N = 200;
t = (0:N-1) * inct;
fr = 9;
signal = 30 * sin (2 * %pi * fr * t + 0.4 * %pi);
plot2d(t,signal);
xtitle('signal = 30 * sin (2 * %pi * 9 * t + 0.4 * %pi) ', ' t ');
```



Exemple 3.1 (suite)

// Transformée Direct de Fourier (Temps --> Fréquence) de Fourier

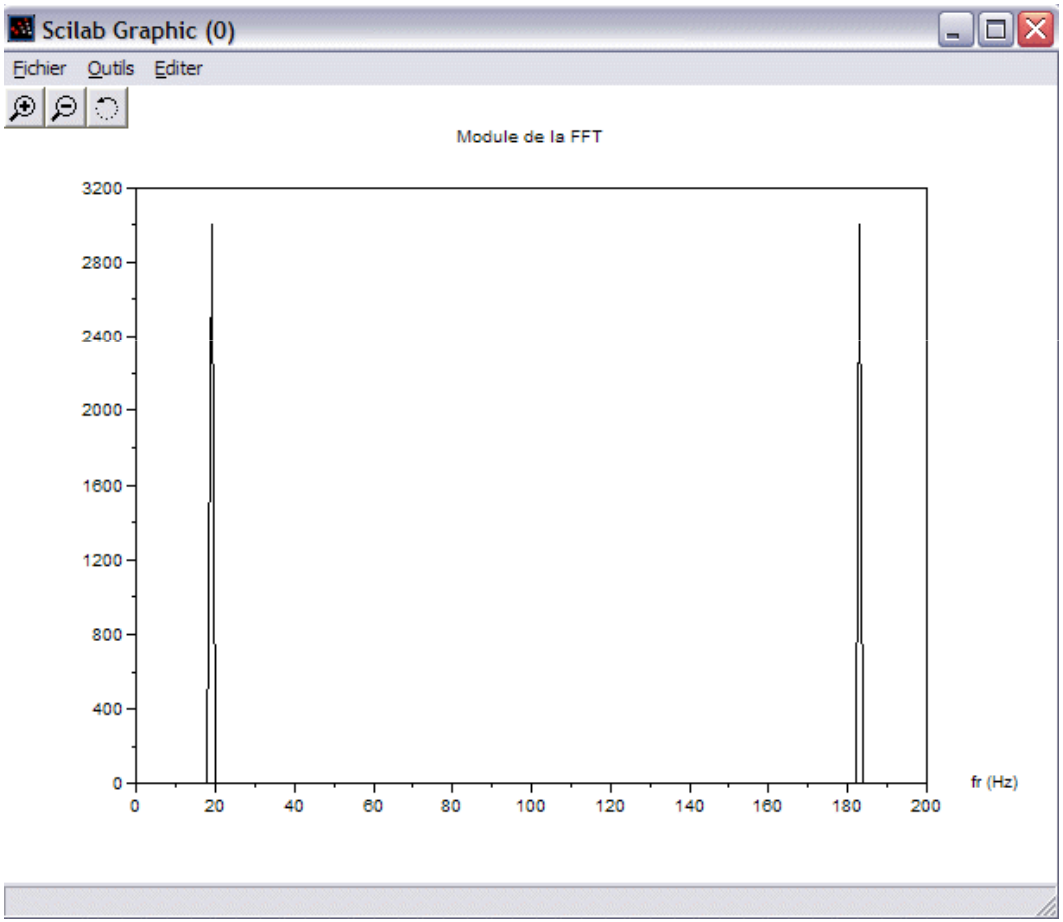
fft_signal = fft (**signal**, **-1**);

Ici nous sommes arrivés à l'étape la plus difficile. Nous avons deux crêtes avec seulement une fréquence. Que signifient-t-elles? Tout d'abord, nous devons comprendre la relation entre la période d'échantillonnage et la fréquence.

incf = 1 / (N*inct);

f = (0:N-1) * incf;

L'incrément de fréquence est l'inverse de tout le temps considéré.

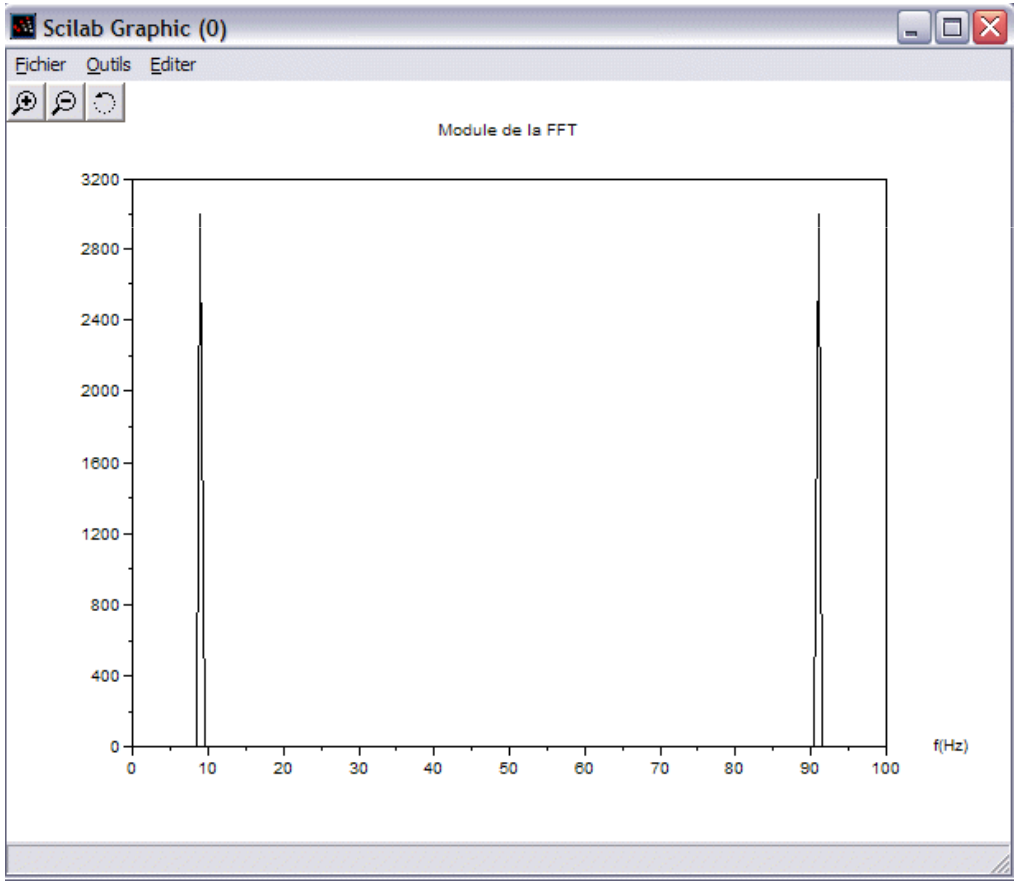


Exemple 3.1 (suite)

Si nous établissons un vecteur de fréquences et le traçons avec la transformation

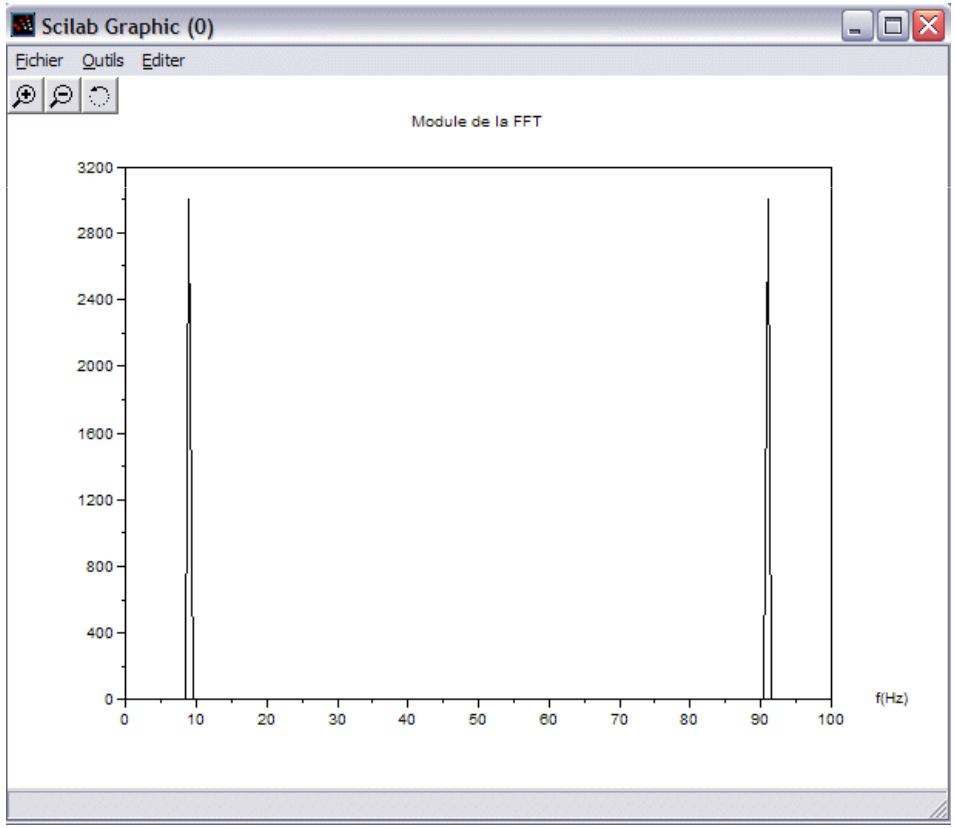
```
xbasc(); plot2d(f,abs(fft_signal));
```

Il est plus facile de comprendre cette figure. Sur l'axe des fréquences nous pouvons voir des fréquences de 0 à 100 avec une crête à 9 hertz. Ce crête représente la fréquence du sinusoïde à laquelle nous avons appliqué la transformée de Fourier. Pour expliquer la deuxième crête nous devons nous rappeler le chapitre au sujet de l'échantillonnage que nous avons traité dans la première partie.



Exemple 3.1 (suite)

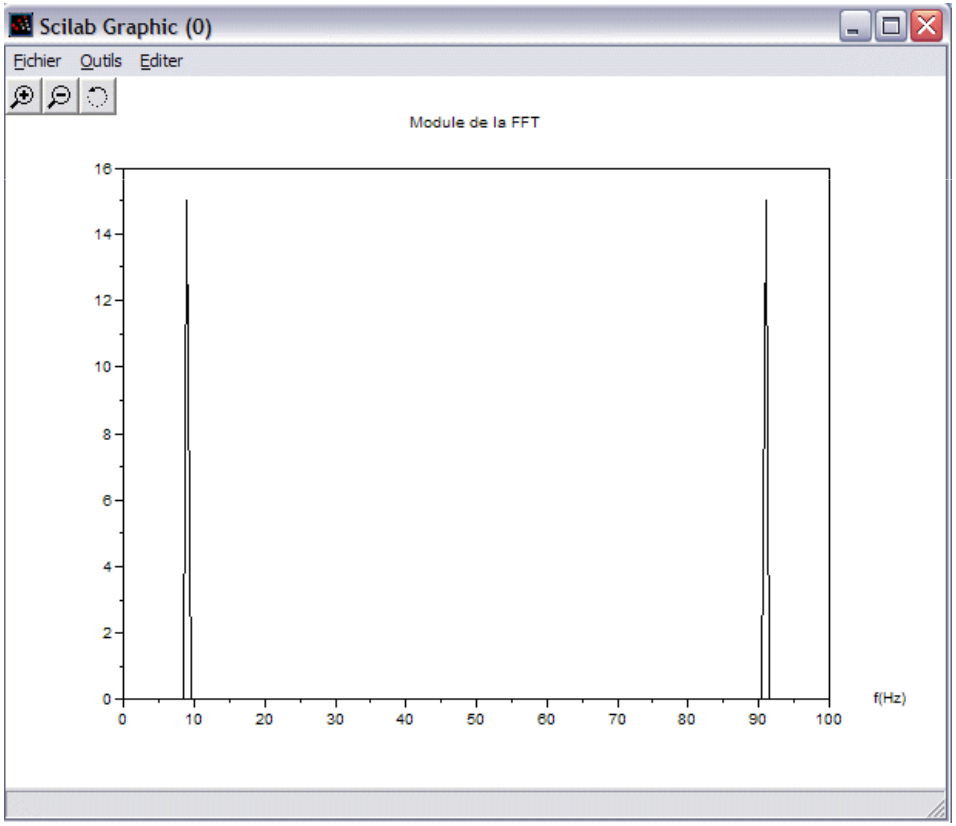
Quand nous échantillonnons un signal nous ne pouvons obtenir aucune fréquence au-dessus de la fréquence de Nyquist. Puisque nous avons échantillonné le sinusoïde à 100 hertz, il ne peut pas y avoir des fréquences au-dessus de 50 hertz. En fait, d'après la définition mathématique de la TF, les fréquences vont de –50 au 50. Les 50 premières valeurs représentent des fréquences du 0 à 49, les 50 deuxièmes valeurs représentent des fréquences de -50 à –1 (fréquences supérieurs à la fréquence de Nyquist). Ainsi la valeur que nous voyons à une fréquence du 91 correspond à une fréquence de -9.



Exemple 3.1 (suite)

La transformée de Fourier a produit deux crêtes symétriques à une fréquence du 9 et -9 Hz. La transformation d'un sinusoïde produit deux crêtes aux fréquences négatives et positives (souvent seulement des fréquences de 0 à la fréquence de Nyquist sont représentées). Maintenant nous devons établir la relation entre l'amplitude du signal et l'amplitude de la transformation. Nous avons une amplitude d'environ 3000 dans chaque crête. Si nous divisons la transformation par le nombre d'échantillons :

```
xbasc() plot2d(f,abs(fft_signal/N));
```

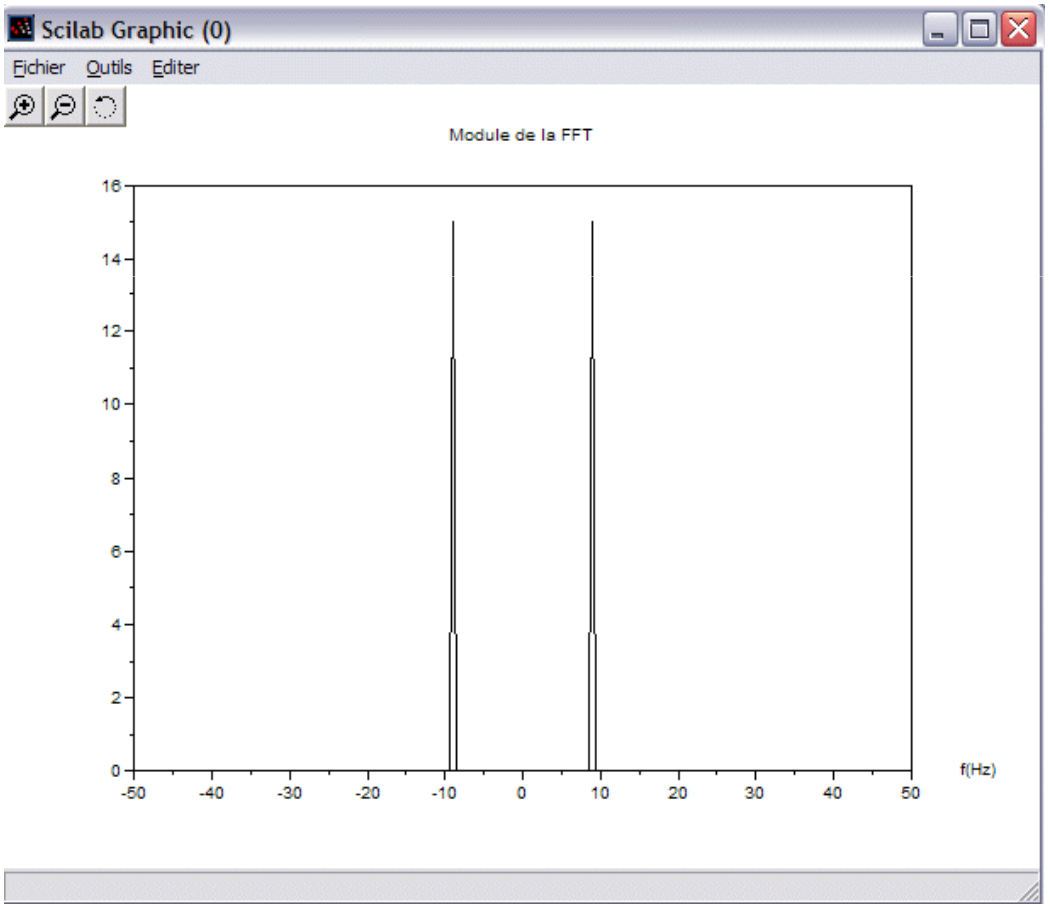


Exemple 3.1 (suite)

Nous pouvons tracer le graphique modifiant l'axe des fréquences pour montrer des fréquences positives et négatives

```
f(101: 200) = f(101: 200)-100;
plot2d(f,abs(fft_signal/N));
```

Nous avons deux crêtes avec une amplitude de 15. Si nous ajoutons les deux crêtes nous obtenons une amplitude de 30, l'amplitude d'origine du sinusoïde.



Exemple 3.1 (suite)

Transformée Inverse : Fréquence à Temps

```
// Transformée Inverse de Fourier (Fréquence --> Temps)
signal2 = fft(fft_signal,1);
max(abs(signal-signal2))
ans =
1.794E-14
```

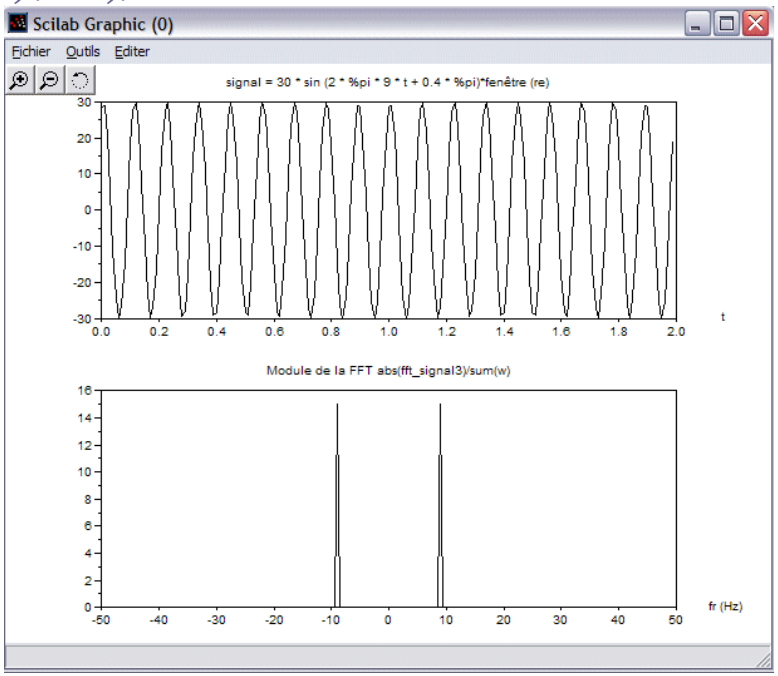
Ce résultat prouve clairement qu'il n'y a aucune différence significative entre le signal d'origine « signal » et le signal que nous obtenons après fabrication de la transformée de Fourier directe et inverse du signal « signal2 ».

Exemple 3.1 (suite)

Influence de la fenêtre rectangulaire (re) sur la Transformée de Fourier (fr=9) :

source : ../simulation2/sim_filtre/sim_fourier/sim_fourier.sce

```
fr=9;
w = window('re',length(t)); // fenêtre rectangulaire
Signal3 = 30 * sin (2* %pi * fr *t + 0.4* %pi) .*w; // multiplier le signal par la fenêtre
fft_signal3=fft (signal3, -1);
xsetech([0,0,1,0.5],[0,-40,2,30]);
plot2d(t,signal3);
xtitle('signal = 30 * sin (2 * %pi * 9 * t + 0.4 * %pi)*fenêtre (re)', ' t ');
xsetech([0,0.5,1,0.5],[0,0,100,20]);
plot2d(f,abs(fft_signal3)/sum(w));// Transformée de Fourier
// Notez que la valeur du module de la transformée de Fourier
//  est divisée par la somme des valeurs de la fenêtre
//  pour augmenter la coïncidence entre les amplitudes.
xtitle('Module de la FFT abs(fft_signal3)/sum(w)', ' f (Hz)');
```

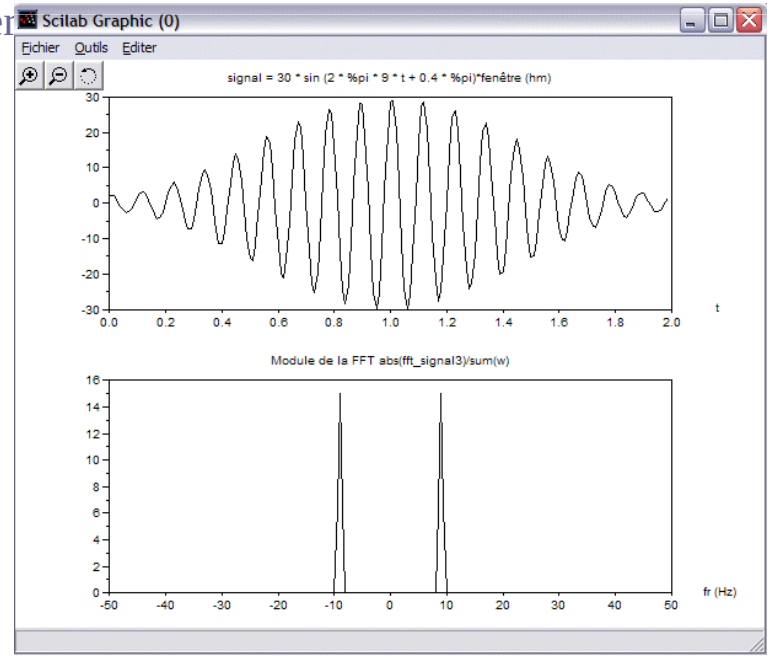


Employer une fenêtre rectangulaire est équivalent à n'employer aucune fenêtre dans le segment considéré.

Exemple 3.1 (suite)

Influence de la fenêtre Hamming (hm) sur la Transformée de Fourier

```
w = window('hm',length(t));// fenêtre Hamming
signal3  = 30 * sin (2 * %pi * fr * t + 0.4 * %pi) .* w;// multiplier le signal par la fenêtre
fft_signal3=fft (signal3, -1);
xsetech([0,0,1,0.5],[0,-40,2,30]);
plot2d(t,signal3);
xtitle('signal = 30 * sin (2 * %pi * 9 * t + 0.4 * %pi)*fenêtre (hm)', ' t ');
xsetech([0,0.5,1,0.5],[0,0,100,20]);
plot2d(f,abs(fft_signal3)/sum(w));// Transformée de Fourier
// Notez que la valeur du module de la transformée de Fourier
//  est divisée par la somme des valeurs de la fenêtre
//  pour augmenter la coïncidence entre les amplitudes.
xtitle('Module de la FFT abs(fft_signal3)/sum(w)', ' f (Hz)');
```



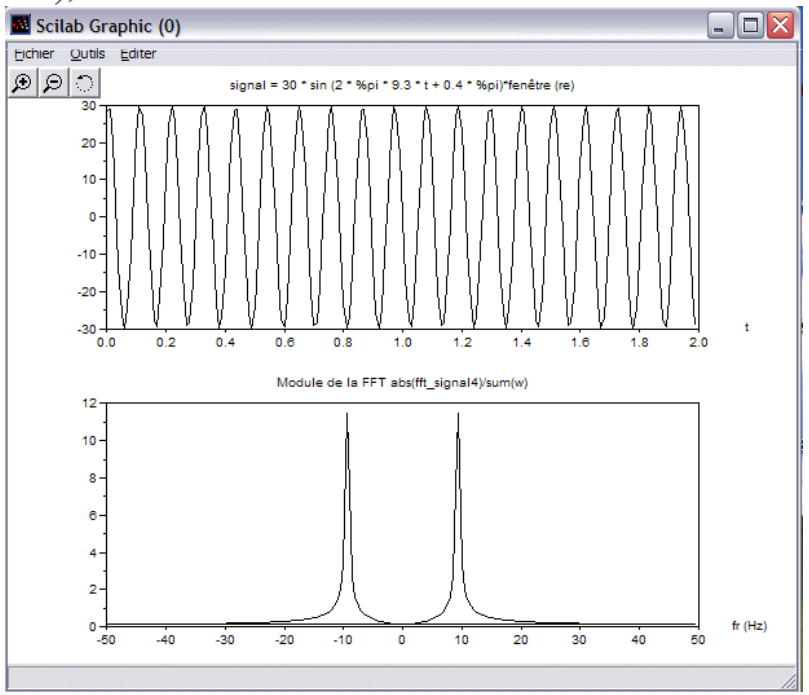
Apparemment, employer une fenêtre hamming est équivalent à n'employer aucune fenêtre dans le segment considéré. Mais y a-t-il des avantages ?!!!

Exemple 3.1 (suite)

Influence de la fenêtre rectangulaire (re) sur la Transformée de Fourier (fr=9.3) :

source : ../simulation2/sim_filtre/sim_fourier/sim_fourier.sce

```
fr=9.3;
w = window('re',length(t)); // fenêtre rectangulaire
Signal4 = 30 * sin (2* %pi * fr *t + 0.4* %pi) .*w; // multiplier le signal par la fenêtre
fft_signal4=fft (signal4, -1);
xsetech([0,0,1,0.5],[0,-40,2,30]);
plot2d(t,signal4);
xtitle('signal = 30 * sin (2 * %pi * 9.3 * t + 0.4 * %pi)*fenêtre (re)', ' t ');
xsetech([0,0.5,1,0.5],[0,0,100,20]);
plot2d(f,abs(fft_signal4)/sum(w));// Transformée de Fourier
xtitle('Module de la FFT abs(fft_signal4)/sum(w)', ' f (Hz)');
```

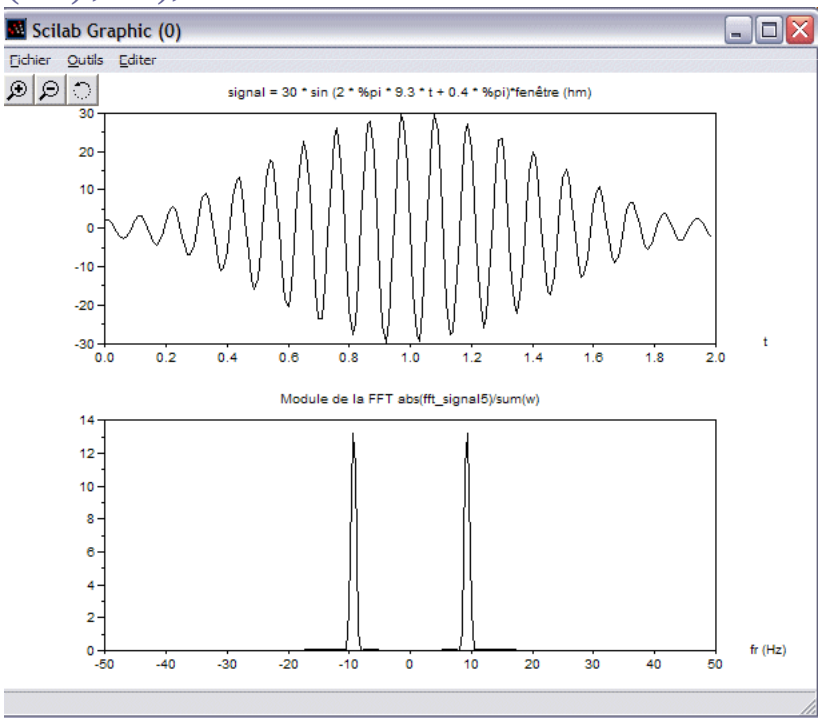


Employer une fenêtre rectangulaire provoque un phénomène de fuite « leakage » : l'activité a fui dehors « leakage out » et la crête est plus large. Bien que nous ayons seulement une fréquence, la crête couvre plusieurs fréquences. Ce phénomène peut être réduit au minimum en employant quelques fenêtres (hamming, …).

Exemple 3.1 (suite)

Influence de la fenêtre Hamming (hm) sur la Transformée de Fourier (fr=9.3)

```
fr=9.3;
w = window('hm',length(t));// fenêtre Hamming
signal5  = 30 * sin (2 * %pi * fr * t + 0.4 * %pi) .* w;// multiplier le signal par la fenêtre
fft_signal5=fft (signal5, -1);
xsetech([0,0,1,0.5],[0,-40,2,30]);
plot2d(t,signal5);
xtitle('signal = 30 * sin (2 * %pi * 9.3 * t + 0.4 * %pi)*fenêtre (hm)', ' t ');
xsetech([0,0.5,1,0.5],[0,0,100,20]);
plot2d(f,abs(fft_signal5)/sum(w));// Transformée de Fourier
xtitle('Module de la FFT abs(fft_signal5)/sum(w)', ' f (Hz)');
```



Employer une fenêtre hamming, améliore la résolution et diminue la fuite.

4.2. Analyse spectrale des signaux aléatoires stationnaires

Spectre de puissance : le théorème de Parseval

Considérons $x[t]$ comme des échantillons d'une tension à travers une résistance à R = 1 unité. Alors, la quantité $P = x[t]^2 /R$ est la puissance dissipée par cette résistance. Par analogie dans la langue de traitement du signal le carré d'une valeur absolue d'un échantillon est appelée puissance instantanée du signal.

4.2. Analyse spectrale des signaux aléatoires stationnaires

Soit $X(f)$ la *TF* de $x(t)$ et $X[k]$ la *TFD* directe de $x[n]$, le ***théorème de Parseval*** (également appelée ***identité de Rayleigh***) déclare: L'énergie totale d'un signal ne dépend pas de la représentation choisie : fréquentielle ou temporelle (***conservation d'énergie***).

- En temps continu :

$$\int_{-\infty}^{\infty} |x[t]|^2 dt = \int_{-\infty}^{\infty} |X[f]|^2 df$$

- En temps discret :

$$\sum_{n=-\infty}^{\infty} |x[n]|^2 = \frac{1}{2\pi} \int_{-\pi}^{\pi} |X(e^{j\varphi})|^2 d\varphi$$

où X est la transformée de Fourier à temps discret (DTFT) de x et φ représente la fréquence angulaire (en radians par échantillon) de x.

- Pour la transformée de Fourier discrète (DFT) :

$$\sum_{n=1}^{N} |x[n]|^2 = \frac{1}{N} \sum_{k=1}^{N} |X[k]|^2$$

Grâce au théorème de Parseval, nous pouvons penser que $|X[k]|^2/N$ est la partie de la puissance du signal transporté par la composante exponentielle complexe de fréquences indexés par k.

Si nous traitons les signaux réels, alors l'exponentielle complexe en série de Fourier vient en couples conjugué indexées par k et N-k pour $k \in 1,. . . , N/2$.
Chacune des composantes de la paire effectue la moitié de la puissance liée à l'oscillation avec une fréquence $f_k=(K/N)f_e$.

Pour récupérer la puissance totale des oscillations à la fréquence f_k nous avons besoin de faire la somme des deux parties. C'est à dire:

$$P(f_k) = \frac{1}{N}(|X[k]|^2 + |X[N-k]|^2)$$

Pour les signaux réels le spectre de puissance est souvent affiché en traçant seulement cette puissance pour les fréquences positives.

Une technique permettant la classification des signaux déterministes ou aléatoires basée sur les notions d'**énergie** et de **puissance**, est extrêmement utile pour sélectionner les meilleurs descripteurs dans les domaines temporels, fréquentiels et statistiques.

Soit $x(t)$ un signal quelconque défini sur $[-\infty, +\infty]$, un intervalle de temps T_0 et un instant quelconque t_0.

L'**énergie totale** E contenue dans x(t) :

$$E_x = \lim_{T_0 \to +\infty} \int_{-\frac{T_0}{2}}^{+\frac{T_0}{2}} x^2(t)\,dt$$

La **puissance moyenne totale** P_m :

$$P_{mx} = \lim_{T_0 \to +\infty} \frac{1}{T_0} \int_{-\frac{T_0}{2}}^{+\frac{T_0}{2}} x^2(t)\,dt$$

L’analyse détaillée des contenus énergétiques des signaux permet d’établir leur classification en plusieurs catégories (valables aussi bien pour les signaux déterministes qu’aléatoires).

• la première classification prend en compte l’énergie totale contenue dans les signaux et l’on distingue deux cas :

-les signaux à énergie fini

-les signaux à énergie infini

• la seconde classification prend en compte leur puissance moyenne totale P_m .

Nous pouvons considérer séparément les signaux à énergie finie et les signaux à énergie infinie :

• pour des signaux à énergie totale *E* finie, $P_m = 0$

• pour des signaux à énergie totale E infinie, nous pouvons distinguer deux cas distincts :

- $P_m = 0$,

- P_m = constante,

En pratique, la majorité des signaux correspondent à la deuxième catégorie (P_m = constante).

Puissances et énergies croisées contenues dans le produit de deux signaux

$$E_{x_1 x_2} = \lim_{T_0 \to +\infty} \int_{-\frac{T_0}{2}}^{+\frac{T_0}{2}} x_1(t) x_2(t)\, dt$$

La **puissance moyenne totale** P_m :

$$P_{m x_1 x_2} = \lim_{T_0 \to +\infty} \frac{1}{T_0} \int_{-\frac{T_0}{2}}^{+\frac{T_0}{2}} x_1(t) x_2(t)\, dt$$

La quantité d'énergie portée par chaque fréquence :

L'énergie entière du signal est proportionnelle à la somme des carrée des valeurs de ses amplitude.

Quelques commandes utiles pour calculer le module des valeurs complexes en Scilab

```
--> x = [5, -5, 3+4*%i, 5*%i, -5*%i]
--> abs(x)
ans = ! 5. 5. 5. 5. 5. !
--> conj(x)
ans = ! 5. - 5. 3. - 4.i - 5.i 5.i !
```

x' donne la transposé (vecteur colonne), **x**.' donne le vecteur colonne de la conjugaison

```
--> abs(x)^2 ou  x .* conj(x) // le carré des valeurs de vecteur x
ans =
! 25. 25. 25. 25. 25. !
--> x.' x
ans =
! 25. 25. - 7. + 24.i - 25. - 25. !
--> x ^2
ans =
! 25. 25. - 7. + 24.i - 25. - 25. !
--> sum(abs(x) ^2) ou x *x ' ou sum(x .* conj(x)) // somme de valeurs carrées
ans =
125
sqrt(x *x ‘) ou norm(x)
ans =
11.18034
```

Rapport signal sur bruit

Utile pour connaître le degré de qualité d'un **signal déterministe** et quantifier l'effet d'un bruit.

$$R_{s/b} = \frac{P_{md}}{P_{mb}}$$

$$R_{s/b}\ (dB) = 10\ \log\ (R_{s/b})$$

où P_{md} et P_{mb} sont les puissances moyennes totales du signal déterministe et du signal de bruit respectivement.

1. Théorème de Wiener Kintchine

Ce théorème démontre que la ***densité spectrale*** d'un signal aléatoire stationnaire $x(t)$ et sa fonction d'autocorrélation $\sigma_{xx}(\tau)$ sont liées par la transformée de Fourier :

$$P_{xx}(f) = F(\sigma_{xx}(\tau)) = \int_{-\infty}^{+\infty} \sigma_{xx}(\tau) e^{-2\pi jf\tau} d\tau$$

$$\sigma_{xx}(\tau) = F^{-1}(P_{xx}(f)) = \int_{-\infty}^{+\infty} P_{xx}(f) e^{+2\pi jf\tau} df$$

Pour un signal réel, la fonction d'autocorrélation est une fonction paire, ce qui entraîne que la densité spectrale de puissance est une fonction réelle paire.

1. Théorème de Wiener Kintchine (suite)

Cas de deux signaux

$$P_{x_1x_2}(f) = F(\sigma_{x_1x_2}(\tau)) = \int_{-\infty}^{+\infty} \sigma_{x_1x_2}(\tau) e^{-2\pi jf\tau} d\tau$$

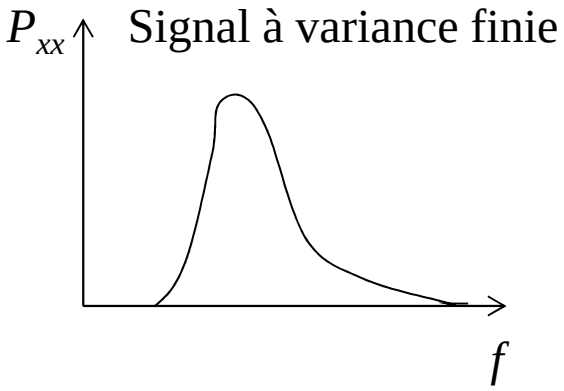
$$\sigma_{x_1x_2}(\tau) = F^{-1}(P_{x_1x_2}(f)) = \int_{-\infty}^{+\infty} P_{x_1x_2}(f) e^{+2\pi jf\tau} df$$

Propriété : $P_{x1x2} = P^*_{x1x2}$

1. Théorème de Wiener Kintchine (suite)

Propriétés

- Les composantes de l'autocorrélation étant homogènes à des carrés d'amplitude, alors la *densité spectrale* P_{xx} est non négative. Elle a une dimension (unité physique)^2 / Hertz.



Dénsités spectrales de puissance et d'énergie

Pour un signal à puissance moyenne totale fini, la puissance du signal sur un intervalle de fréquence (f_0, f_1) :

$$P_x(f_0, f_1) = \int_{f_0}^{f_1} P_{xx}(f)\, df$$

Pour un signal à énergie finie, la fonction d'autocorrélation est homogène à une énergie :

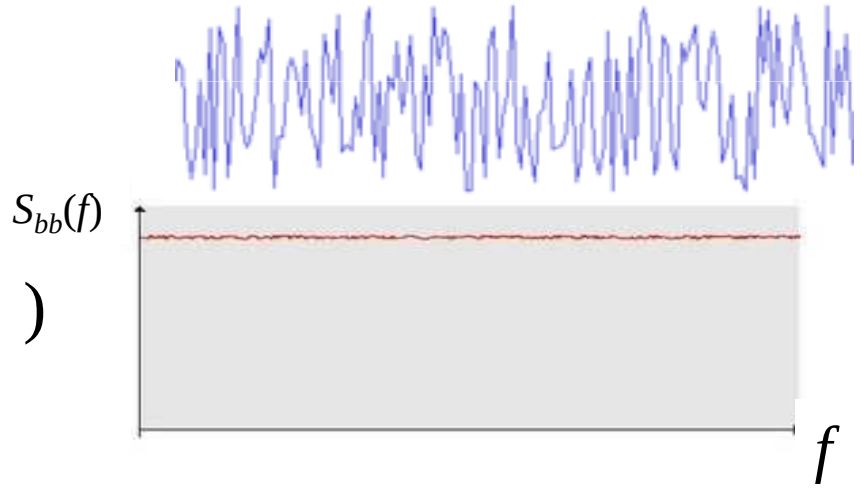
$$E_x(f_0, f_1) = \int_{f_0}^{f_1} P_{xx}(f)\, df$$

Bruit blanc

Un ***bruit blanc*** est un signal aléatoire dont la densité spectrale de puissance est constante quelle que soit sa fréquence [-∞,+ ∞]. En d'autres termes, la probabilité qu'un bruit blanc possède une certaine puissance est la même pour toutes les fréquences.

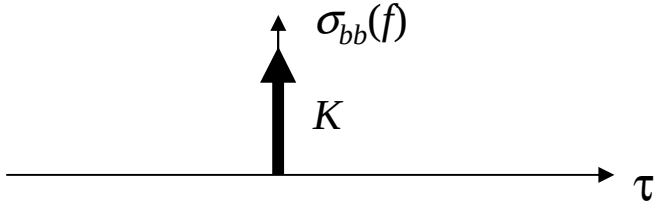
Par l'application du **théorème de Wiener Kintchine** on déduit :

$$\sigma_{bb}(\tau) = \int_{-\infty}^{+\infty} K e^{-2\pi j f \tau} d\tau = K\delta(\tau)$$



Pour $\tau = 0$, on obtient

$$\sigma_{bb}(0) = \sigma_b^2 = E[(x(t)^2] = K\delta(0)$$



Bruit blanc (suite)

- En toute rigueur un bruit blanc ne peut exister car une densité spectrale identique pour toutes les fréquences conduirait à une variance, mesurée par l'aire sous la courbe, infinie. Il n'existe donc, que des bruits blancs limités à une bande de fréquences (***bruit coloré***).

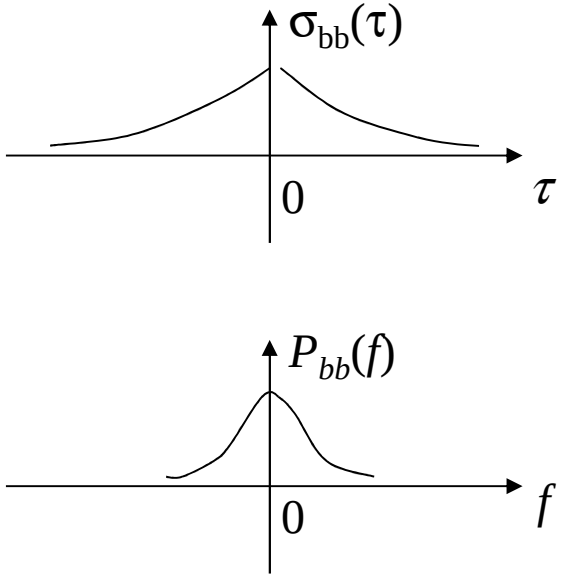
Bruits réels

Les bruits réels ont des caractéristiques spectrales qui ne possèdent pas de discontinuité car dans les plupart des cas, ils correspondent au filtrage de bruits physiques dans des systèmes linéaires invariants dans le temps et causals. Ces bruits sont à basses fréquences, à hautes fréquences, à bande étroite ou à large bande.

La fonction de corrélation et la densité spectrale d'un signal à basse fréquences peut par exemple être caractérisées par :

$$\sigma_{bb}(\tau) = e^{-\alpha|\tau|}$$

$$P_{bb}(f) = \frac{2\alpha}{\alpha^2 + 4\pi^2 f^2}$$



4.3. Spectre de puissance

Différentes applications de l'analyse de fréquence :

- Nous pouvons évaluer les différentes fréquences présentent dans le signal. Elles peuvent correspondre à différents processus impliqués dans la génération du signal.
- Nous pouvons étudier les composants d'un signal pour définir un filtre qui accentue quelques aspects du signal.
- Nous pouvons évaluer avec le spectrogramme (ou par les ondelettes ou par l'EMD) la modification des fréquences dans le temps.
- Nous pouvons détecter des fréquences non évidentes dans le domaine de temps comme dans le cas du d'un signal masqué par le bruit.
- Nous pouvons étudier avec la cohérence les fréquences qui maintiennent une relation persistante entre différents signaux.

Méthodes d'estimation du spectre de puissance

➢ Périodogramme

Estimation du spectre de puissance en utilisant la *FFT*

$$P_{XX}(\tau) = \frac{\left|X(f_k)\right|^2}{N.f_e}$$

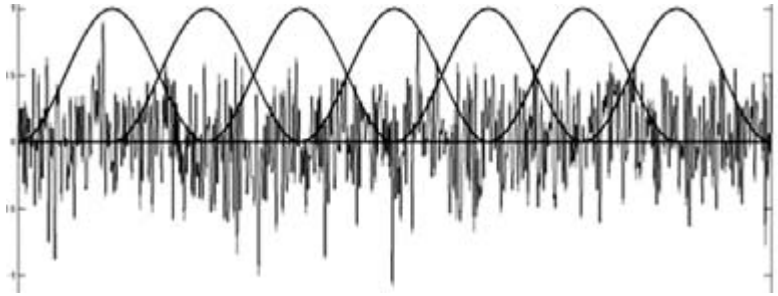
où $X(f_k)$ est la TFD sur $x(n)$ sur $f_k = k.\frac{f_e}{N}, k = 0, 1, \cdots, N-1$

En général, la longueur N de la *FFT* et les valeurs de l'entrée $x(n)$ déterminent la longueur de P_{xx} et la gamme des fréquences correspondantes.

Méthodes d'estimation du spectre de puissance (suite)

➢ Méthode de welch

- Diviser les données en plusieurs segments (en Matlab ***window***), qui se recoupent avec un certain nombre de chevauchements (en Matlab ***noverlap***),
- Effectuer une FFT sur chaque segment avec un fenêtre donnée (en Matlab ***nfft)***,
- Calculer le spectre de puissance (amplitude élevée au carré),
- Calculer la moyenne de ces spectres



Spectre de puissance d'un signal de bruit blanc

Le bruit blanc est un signal dont le spectre de puissance est homogènement distribué à travers les fréquences. En ce chapitre nous allons produire un long signal aléatoire en employant le « random » avec l'option « normal ». Puis nous calculerons son spectre de puissance.

Dès qu'un spectre de puissance plat issu d'un bruit Gaussien est produit, nous pouvons éliminer certaines fréquences par filtrage pour obtenir de bruit coloré. Nous pouvons vérifier le comportement d'un système en comparant les deux spectres.

Spectre de puissance d'un signal de bruit blanc

Exemple

a. Concevoir un filtre passe-bas à fréquence de coupure de 0.25 Hz et appliquer le filtre à un bruit blanc.

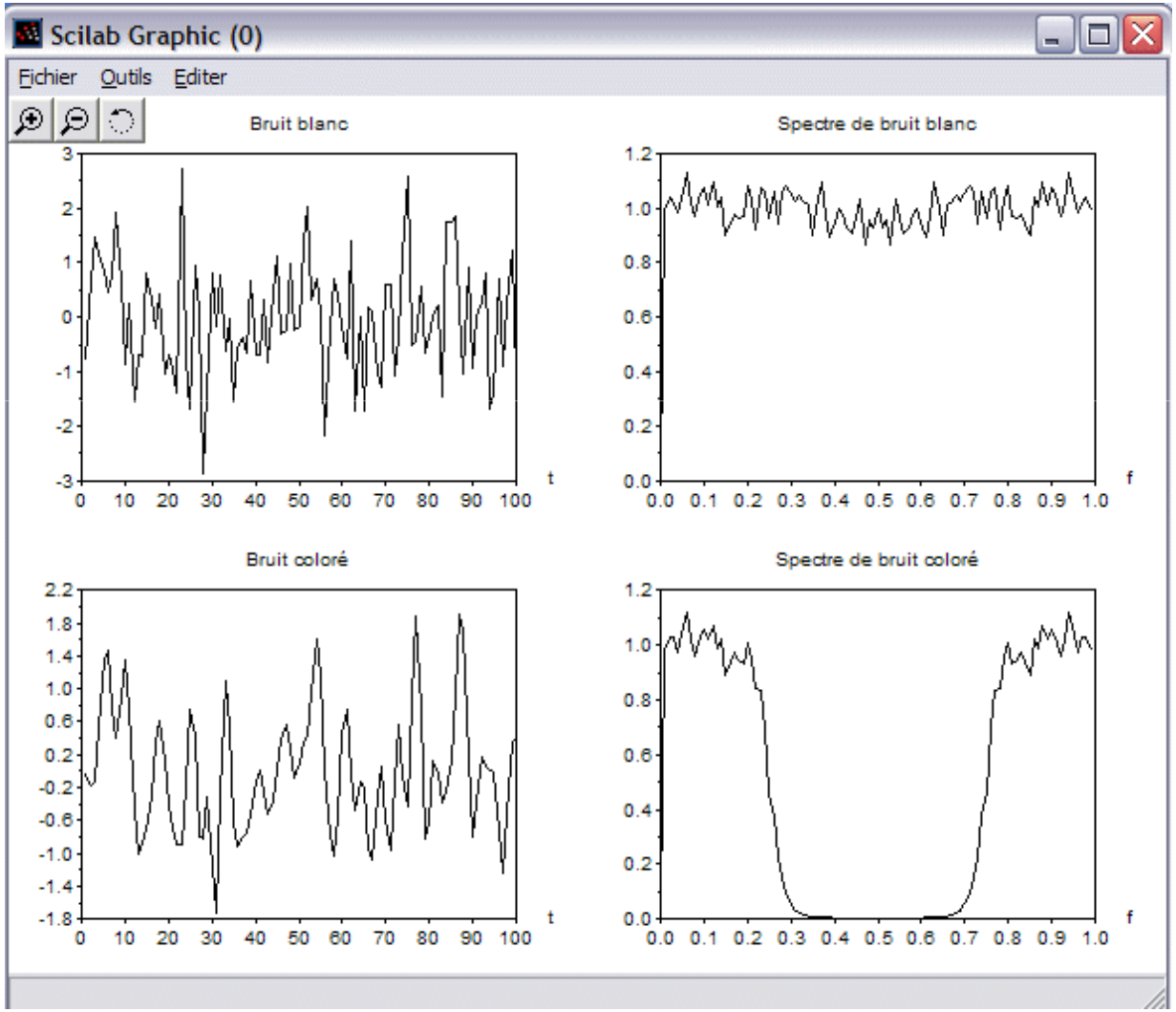
source : ../simulation2/ densité_spectrale/Scilab/bruitb_spec.sce

```
rand('normal');
rand('seed',0);
bb = rand(1,20000);
sys = iir(5,'lp','butt',[0.25 0],[0 0]);
bc = flts(bb,sys);
N = 100;
inct = 1;
incf = 1 / (N * inct);
f = [0:(N-1)] * incf;
xbasc()
xsetech([0,0,1/2,1/2])
plot (bb(1:100)); xtitle('Bruit blanc','t');
xsetech([1/2,0,1/2,1/2])
plot2d(f,pspect(50,100,'re',bb)); xtitle('Spectre de bruit blanc','f');
xsetech([0,1/2,1/2,1/2])
plot(bc(1:100)); xtitle('Bruit coloré','t');
xsetech([1/2,1/2,1/2,1/2])
plot2d(f,pspect(50,100,'re',bc)); xtitle('Spectre de bruit coloré','f');
```

Spectre de puissance d'un signal de bruit blanc

Exemple (suite)

Notez que les fréquences au-dessus de 0.5 sont une forme qui représente des fréquences négatives et étant donné que le filtre passe-bas a éliminé des hautes fréquences, les composants au-dessus de 0.25 hertz ont disparu.



Spectre de puissance d'un signal de bruit blanc Exemple (suite)

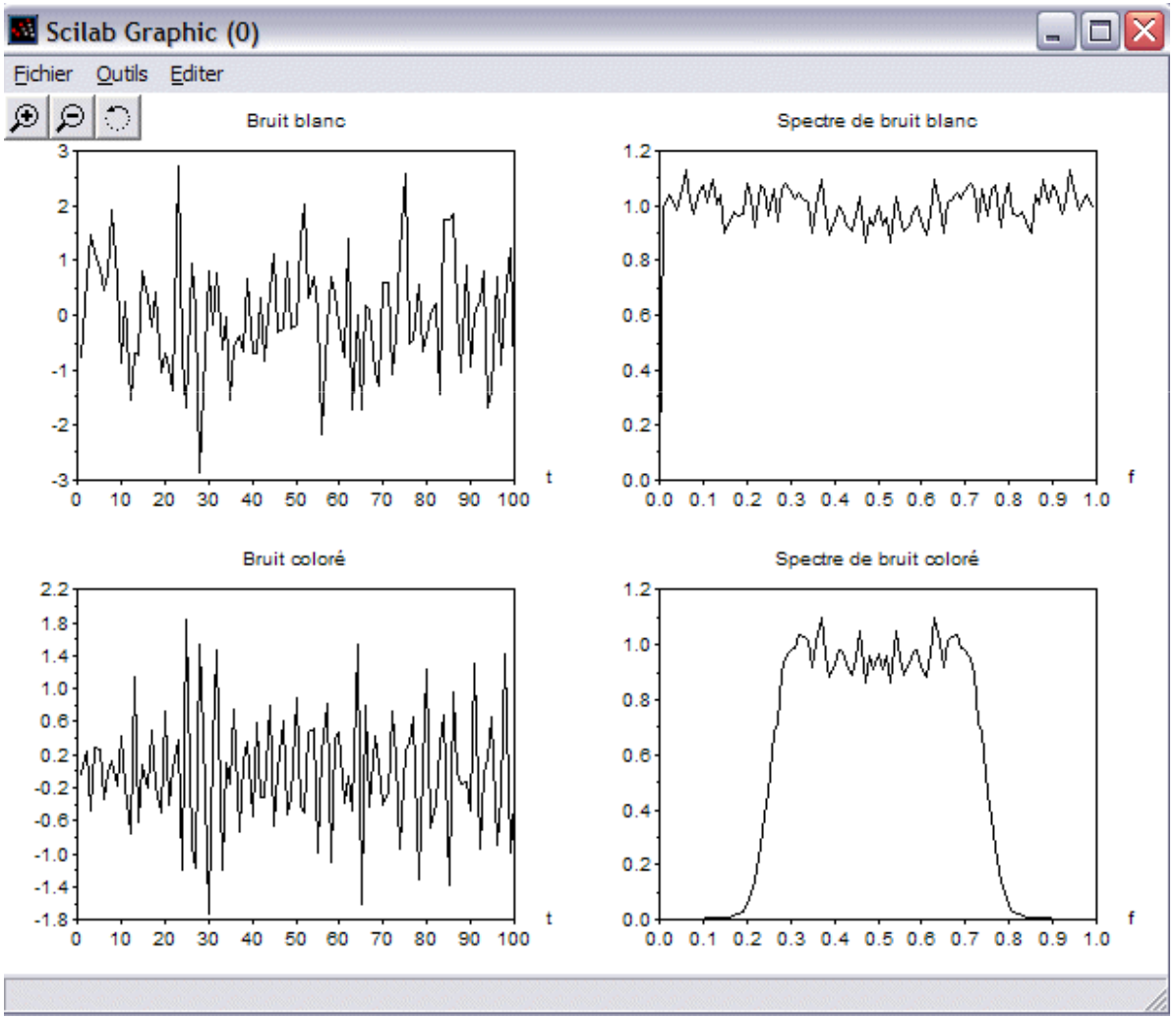
b. Concevoir un filtre passe-haut à fréquence de coupure de 0.25 Hz et appliquer le filtre à un bruit blanc.

source : ../simulation2/sim_filtre/sim_spect/ Scilab/bruitb_spec.sce

```
rand('normal');
rand('seed',0);
bb = rand(1,20000);
sys = iir(5,'hp','butt',[0.25 0],[0 0]);
bc = flts(bb,sys);
N = 100;
inct = 1;
incf = 1 / (N * inct);
f = [0:(N-1)] * incf;
xbasc()
xsetech([0,0,1/2,1/2])
plot (bb(1:100)); xtitle('Bruit blanc','t');
xsetech([1/2,0,1/2,1/2])
plot2d(f,pspect(50,100,'re',bb)); xtitle('Spectre de bruit blanc','f');
xsetech([0,1/2,1/2,1/2])
plot(bc(1:100)); xtitle('Bruit coloré','t');
xsetech([1/2,1/2,1/2,1/2])
plot2d(f,pspect(50,100,'re',bc)); xtitle('Spectre de bruit coloré','f');
```

Spectre de puissance d'un signal de bruit blanc Exemple (suite)

Notez que les fréquences au-dessus de 0.5 ont une forme qui représente des fréquences négatives et puisque le filtre passe-haut a éliminé des bases fréquences, les composants au-dessous de 0.25 hertz ont disparu.



Spectre de puissance d’un signal de bruit blanc Exemple (suite)

Nous pouvons voir que nous commençons par le bruit blanc dont le spectre est distribué de façon homogène nous éliminons certaines fréquences par le filtrage. Dans les parties gauches des figures nous pouvons voir l'effet dans le domaine temporel. La disparition de certaines fréquences est également évidente sans analyse spectrale.

Nous pourrions généraliser cette idée et considérer un signal (par exemple une trace d'EEG) comme un résultat de l'application d’un filtre au bruit blanc. Nous pourrions étudier les paramètres de ce filtre comme une manière de simplifier la description du signal ou d'analyser ses modifications dans le temps (par exemple pour détecter la transition entre l’éveil et le sommeil).

Voir aussi le même exemple en Matlab :
..\simulation2\densité_spectrale\Matlab\BruitBlac\..

Spectre de puissance d’un signal sinusoïde

Exemple

Nous pouvons définir l'énergie d'un signal comme la somme du signal carré. Nous commençons avec 20 sec d’un sinusoïde à 9.3 Hz échantillonné à 100 Hz.

source : ../simulation2/densite_spectrale\Scilab/sim_spec_sinus.sce

```
inct = 0.01;
N = 2000; // nombre de points pour 20 sec.
t = (0:N-1) * inct;
fr = 9.3;
x = sin(2*%pi*fr*t);
```

Exemple 4.1 (suite)

Spectre de puissance d'un signal sinusoïde (suite)

Nous allons calculer la somme des carrées de ses valeurs mais au lieu de calculer le vecteur entier nous allons calculer par des groupes de 100 échantillons (1 s). puis, nous calculons la transformée de Fourier et ajoutons les valeurs carrées de ses coefficients. Les deux valeurs sont à peu près les mêmes quand nous divisons la somme des coefficients carrés de la transformée de Fourier par le nombre d'échantillons.

```
--> x1 = x(1:100);
--> ssumx1 = x1 * x1'
ssumx1 =
49.769926

--> ffx1 = fft(x1,-1);
--> ssumffx1 = sum(abs(ffx1)^2) / 100
ssumffx1 =
49.769926
```

Exemple 4.1 (suite)

Spectre de puissance d'un signal sinusoïde (suite)

Si nous répétons le processus avec les valeurs de la prochain seconde, nous obtenons (bien que non identique) un résultat très semblable

```
--> x2 = x(101:200);
--> ssumx2 = x2 * x2'
ssumx2 =
49.697833

--> ffx2 = fft(x2,-1);
--> ssumffx2 = sum(abs(ffx2)^2) / 100
ssumffx2 =
49.697833
```

Si l'amplitude du signal est mesurée en V, une énergie sera mesurée en V^2 ; si l'amplitude du signal est mesurée en mV, l'énergie sera en mV^2.

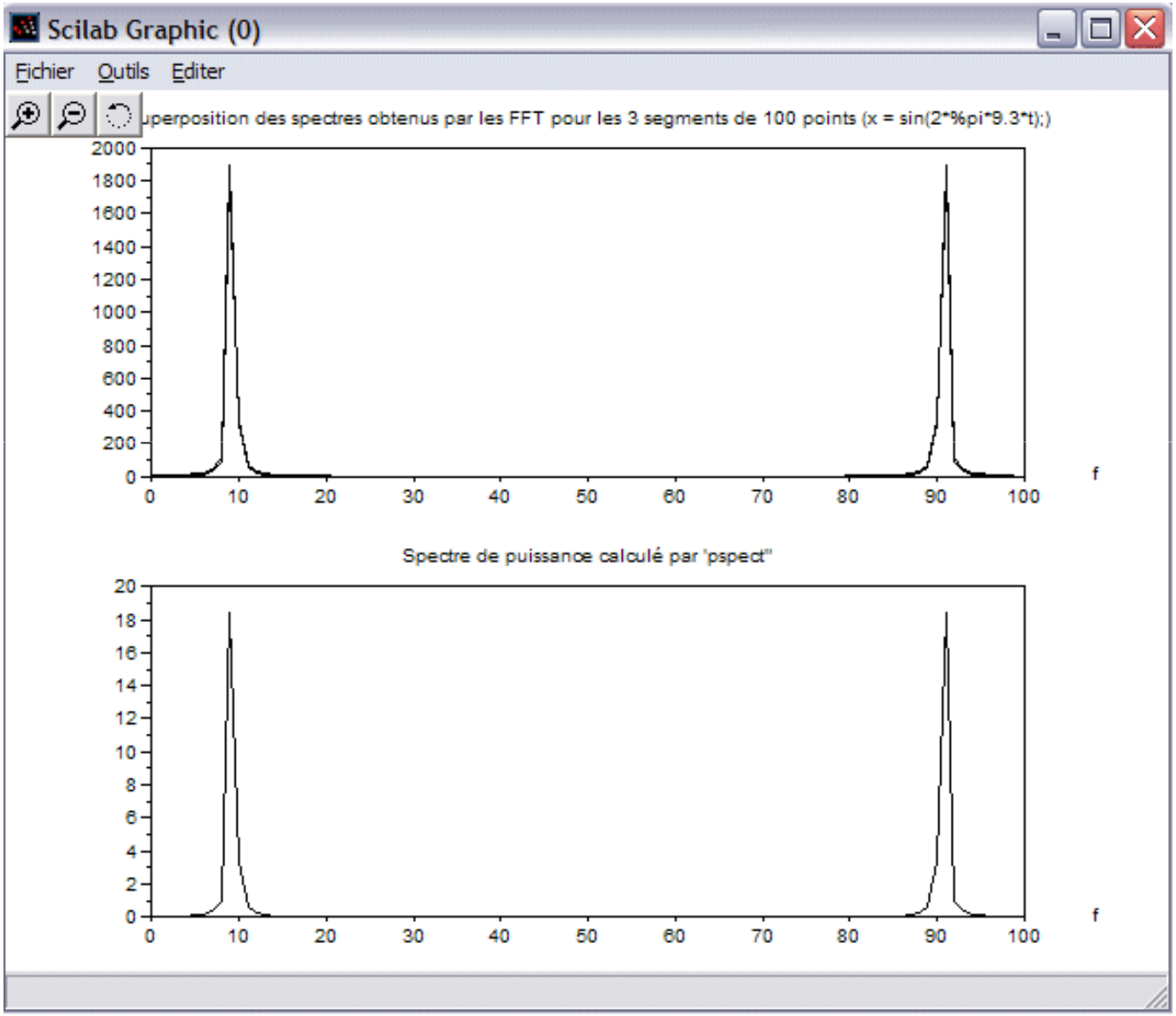
Exemple 4.1 (suite)

Spectre de puissance d'un signal sinusoïde (suite)

```
// Calculer la moyenne par "pspect"
//Avec cette commande, nous disons que  "xnoise" a été divisé dans les
// fenêtres de 100 valeurs et que le spectre de puissance soit calculé.
// Enfin les différents spectres sont ramenés à une moyenne. Nous pouvons
// appliquer différentes fenêtres (rectangulaires, hamming...).
Pxx = pspect(50,100,'re',x);
xsetech([0,0,1,1/2]);
f = [0:99]*(1/ (100*inct));
plot2d(f,abs(fft_x1)^2,1);
plot2d(f,abs(fft_x2)^2,1,'001');
plot2d(f,abs(fft_x3)^2,1,'001');
xtitle('Superposition des transformées de Fourier pour les 3 segments de 100 points (x = sin(2*%pi*9.3*t);)', ' f ');
xsetech([0,1/2,1,1/2]);
plot2d(f,Pxx);
xtitle('Spectre de puissance calculé par '"pspect"", ' f ');
```

Exemple 4.1 (suite)

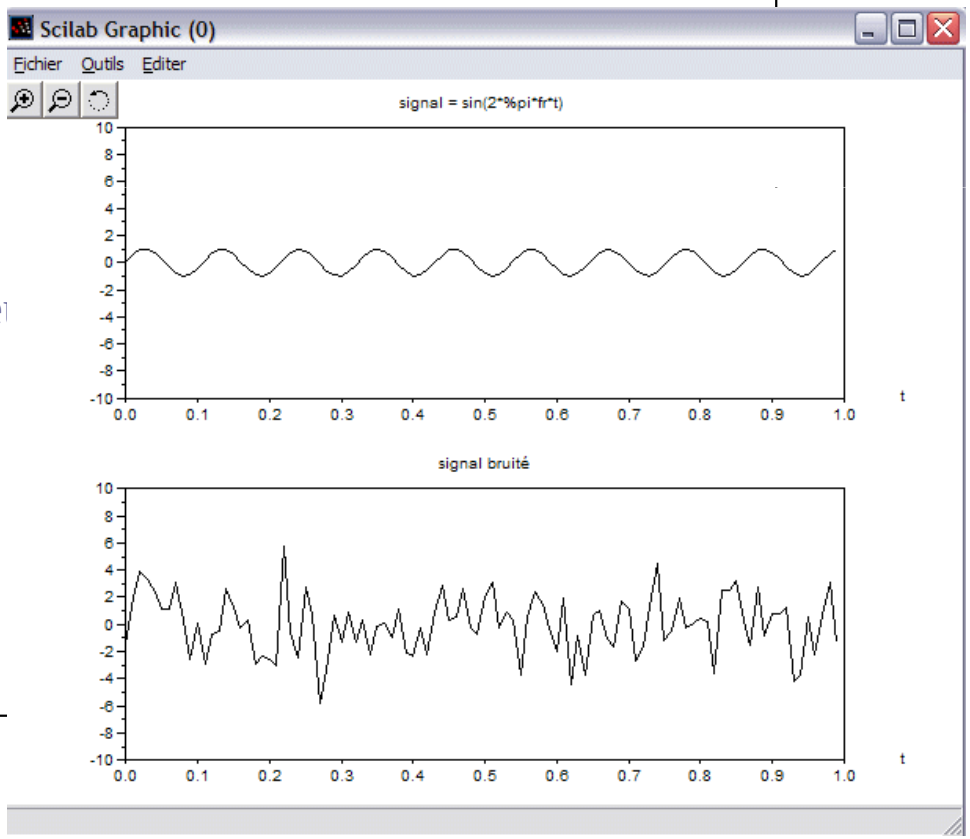
Spectre de puissance d'un signal sinusoïde (suite)



Exemple 4.1 (suite)

Spectre de puissance d'un signal sinusoïde bruité

```
rand('normal')
rand('seed',0)
xnoised = x + 2*rand(x); // ajouter du bruit au signale
xsetech([0,0,1,1/2],[0,-10,1,10]);
plot2d([0:99]*inct, x(1:100),1,'001'); // représenter les premières 100 valeurs
xtitle('signal = sin(2*%pi*fr*t) ', ' t ');
xsetech([0,1/2,1,1/2],[0,-10,1,10]);
plot2d([0:99]*inct, xnoised(1:100),1,'001');
xtitle('signal bruité', ' t ');
// Calculer la Transformée de Fourier
//pour trois segments successifs de 100 vale
x1 = xnoised(1:100);
fft_x1 = fft(x1,-1);
x2 = xnoised(101:200);
fft_x2 = fft(x2,-1);
x3 = xnoised(201:300);
fft_x3 = fft(x3,-1);
```



Exemple 4.1 (suite)

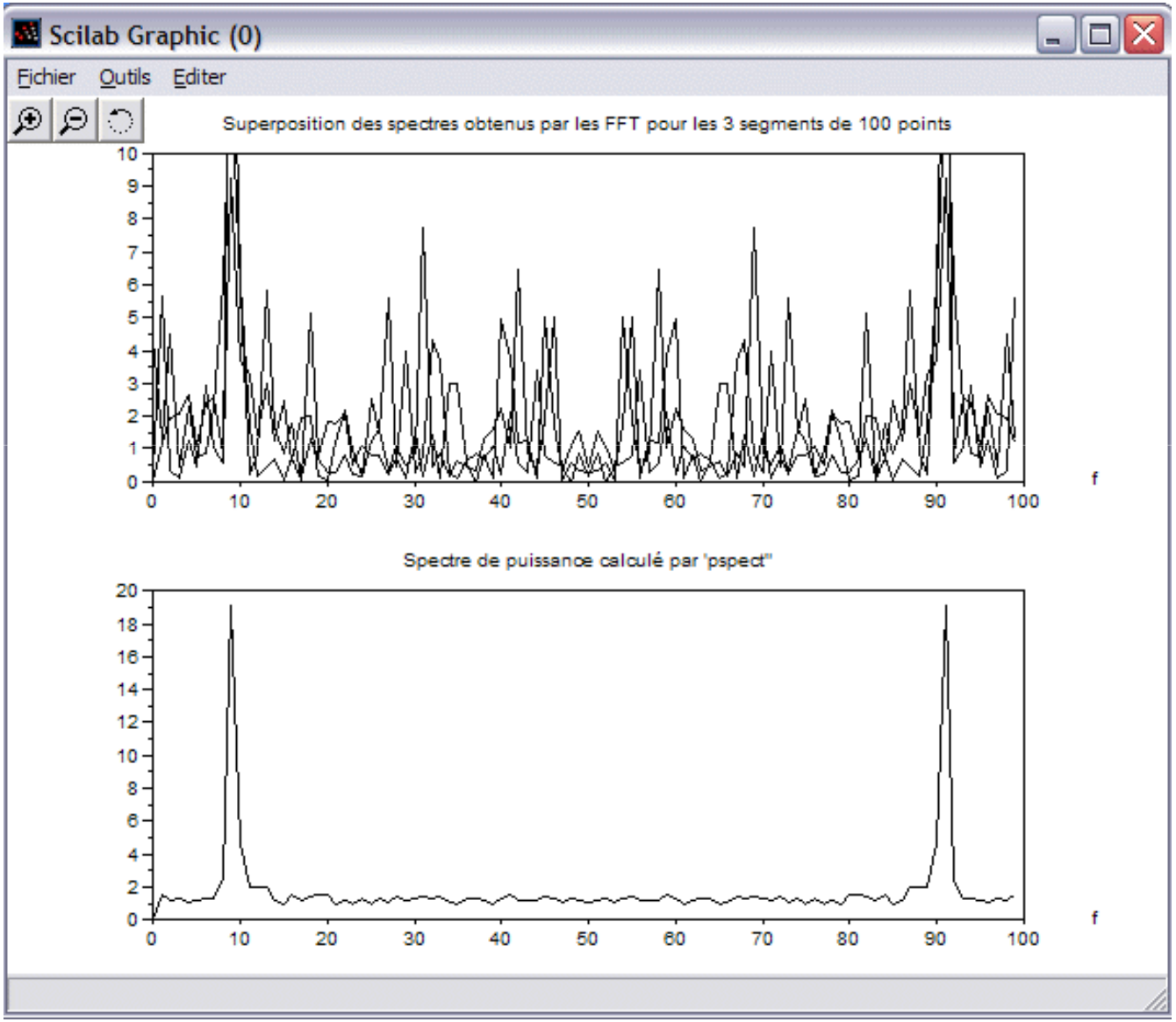
Spectre de puissance d'un signal sinusoïde bruité (suite)

source : ../simulation2/densite_spectrale\Scilab/sim_spec_sinus.sce

```
// Calculer la moyenne par "pspect"
//Avec cette commande, "xnoised" a été divisé dans les
// fenêtres de 100 valeurs et que le spectre de puissance soit calculé.
// Enfin les différents spectres sont ramenés à une moyenne. Nous pouvons
// appliquer différentes fenêtres (rectangulaires, hamming...).
Pxx = pspect(50,100,'re',xnoised);
xbasc();
xsetech([0,0,1,1/2]);
f = [0:99]*(1/ (100*inct));
plot2d(f,(abs(fft_x1)^2 /100),1);
plot2d(f,(abs(fft_x2)^2 /100),1,'001');
plot2d(f,(abs(fft_x3)^2 /100),1,'001');
xtitle('Superposition des transformées de Fourier pour les 3 segments de 100 points', ' f ');
xsetech([0,1/2,1,1/2]);
plot2d(f,Pxx);
xtitle('Spectre de puissance calculé par '"pspect"', ' f ');
```

Exemple 4.1 (suite)

Spectre de puissance d'un signal sinusoïde bruité (suite)



Spectre de puissance d'un signal ECG

Exemple

source : ../simulation2/densite_spectrale/Scilab/ecg_spec.sce

Utiliser la liste scilab « signals » (la variable que nous avons sauvée dans la première partie de ce cours au sujet du format EDF).

1. Extraire le ECG (numéro 12 dans la liste ecg=signals(12)).
2. Exécuter le programme « read_edf » dans le répertoire « ..\simulation2\sim_spectre\ » puis charger le fichier « \simulation1\sim_binaire_edf\osas\osas.edf »
3. Extraire la durée du signale en tapant dans la fenêtre Scilab : « duration »
4. Extraire le nombre d'échantillons contenues dans le signal en tapant dans la fenêtre Scilab : « nsamples(12) »
5. Calculer la période d'échantillonnage « inct ».

Spectre de puissance d'un signal ECG

Exercice (suite)

6. Extraire une information pour étalonner le signal en tapant dans la fenêtre Scilab : « physmin(12) », « physmax(12) », « digmin(12) », « digmax(12) » et « physdim(12) ». Ces valeurs permettent d'effectuer une correction de l'amplitude du signale. Où

- « physmin(12) » : valeur réelle minimale de l'amplitude, mesurée en unité « physdim(12) », correspondant à la valeur numérique « digmin(12) »
- « physmax(12) » : valeur réelle maximale de l'amplitude, mesurée en unité « physdim(12) », correspondant à la valeur numérique « digmax(12) »

7. Calculer le nombre d'échantillons « N » correspondant à 1 sec.
8. Construire un vecteur de temps « t » sur une seconde en utilisant la période d'échantillonnage comme incrémentation (t = [1:N]*inct).
9. Calculer l'incrémentation en fréquence « incf » (incf = 1/(1:N*inct).

Spectre de puissance d'un signal ECG

Exemple (suite)

10. Construire un vecteur de fréquence « f » (f= [0:N-1]*incf).
11. Considérer 3 segments successifs de N points :
    - seg1=1:N,
    - seg2=N+1:2*N et
    - seg3=2*N+1:3*N.
12. Tracer les trois segments et leurs spectres en utilisant leurs transformées directes de Fourier comme suit:

```
x1= ecg(seg1);
xsetech([0,0,1/2,1/3]);
plot2d(t,x1);
xtitle('Segment 1','t');
xsetech([1/2,0,1/2,1/3]);
fft_x1 = fft(x1,-1);
ffx1 = (abs(fft_x1)^2) / N;
plot2d(f,abs(ffx1)^2 /100);
xtitle('Spectre 1', 'f');
```

```
x2= ecg(seg2);
xsetech([0,1/3,1/2,1/3]);
plot2d(t,x2);
xtitle('Segment 2','t');
xsetech([1/2,1/3,1/2,1/3]);
fft_x2 = fft(x2,-1);
ffx2 = (abs(fft_x2)^2) / N;
plot2d(f,abs(ffx2)^2 /100);
xtitle('Spectre 2', 'f');
```

```
x3= ecg(seg3);
xsetech([0,2/3,1/2,1/3]);
plot2d(t,x3);
xtitle('Segment 3','t');
xsetech([1/2,2/3,1/2,1/3]);
fft_x3 = fft(x3,-1);
ffx3 = (abs(fft_x3)^2) / N;
plot2d(f,abs(ffx3)^2 /100);
xtitle('Spectre 3', 'f');
```

Spectre de puissance d'un signal ECG

Exercice (suite)

13. Introduire les lignes suivantes :

```
printf('Prochain tracé : Spectre de signal ecg'"\n');
printf('PAUSE, Tapez "'resume"' pour continuer\n');
pause;
```

14. Utiliser une fenêtre rectangulaire 're' de largeur N et un offset de pas N/2 pour tracer le spectre en même temps que le tracé de FFT de différents segments comme suit :

```
Pxx = pspect(N/2,N,'re',ecg); // largeur de la fenêtre=N, offset=N/2
xbasc()
xsetech([0,0,1,1/2]);
plot2d(f,ffx1,1);
plot2d(f,ffx2,1,'001');
plot2d(f,ffx3,1,'001');
xtitle('Superposition des spectres obtenus par les FFT pour les 3 segments de '+string(N)+' points', ' f ');
xsetech([0,1/2,1,1/2]);
plot2d(f,Pxx);
xtitle('Spectre de puissance calculé par "'pspect"', ' f ');
```

15. Étudier, en utilisant la FFT, si l'utilisation d'un plus long segment (e.g. N=200, 500, 1000) aurait produit le même résultat. Conclure.

16. Étudier, en utilisant « pspect », si l'utilisation d'un plus long segment (e.g. N=200, 500, 1000) aurait produit le même résultat. Conclure.

4.4. Analyse Temps-Fréquence

Principe d’incertitude

L’analyse de Fourier ne représente les signaux soit dans le domaine temporel soit dans le domaine fréquentiel. Pour des signaux stationnaires, il suffit de connaître l'une des deux représentations. Les signaux réels sont souvent un mélange d'éléments stationnaires et non-stationnaires, par exemple, les fuseaux alpha, les fuseaux du sommeil, les K-complexes ou d'autres éléments avec une activité de fond de EEG continu.

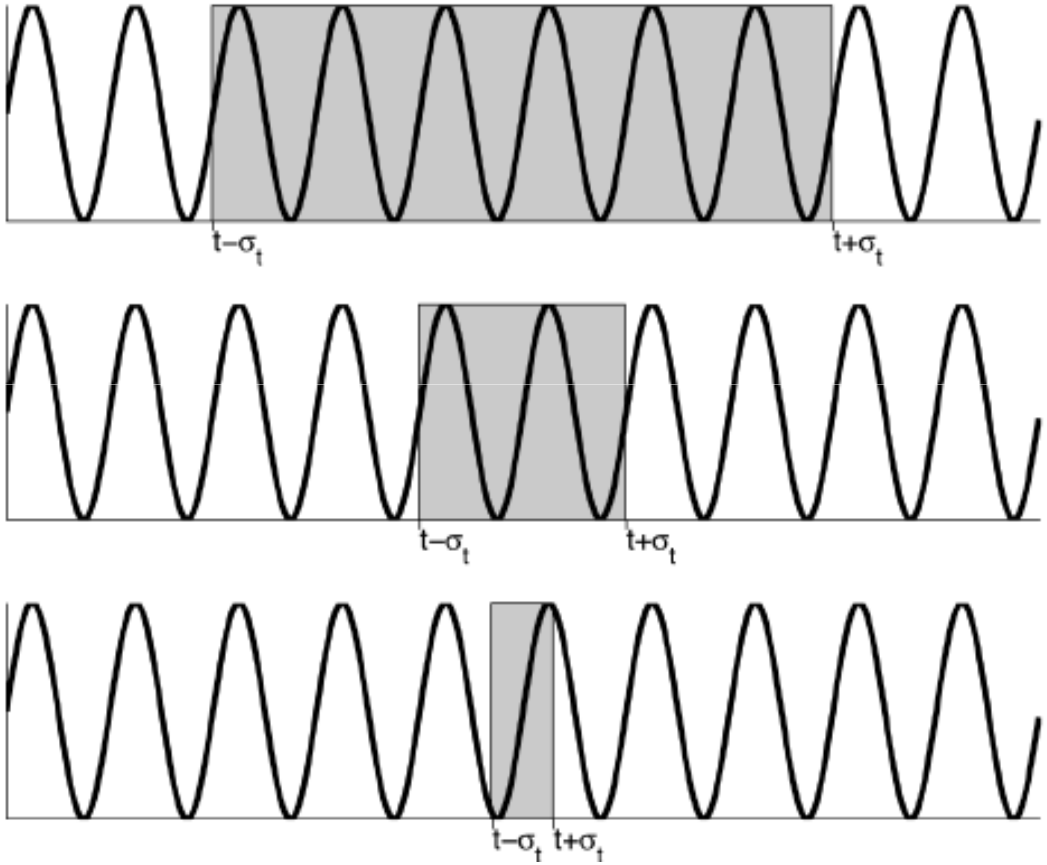
Principe d'incertitude

Nous sommes souvent intéressés par la caractérisation des transitoires non-stationnaires. Il est naturel de penser à leur fréquences et leur localisations dans le temps. Il est également évident qu'une telle transitoire a une certaine durée dans le temps.

L'étendue du temps peut être caractérisée par σ_t. La localisation de la transitoire dans le domaine fréquentiel a aussi une résolution finie caractérisée par une étendue de fréquence σ_f. Ces deux plages sont délimitées par le ***principe d'incertitude***.

Suposons qu'un fragment d'une sinusoïde observé au fil du temps $T = (t - \sigma_t, t + \sigma_t)$,

Nous pouvons calculer sa fréquence en divisant le nombre de périodes observées pendant le temps *T* par la longueur de *T*. Comme nous rétrécissons le temps *T* la localisation dans le temps du fragment de la sinusoïde devient plus précis, mais inversement l'estimation du nombre de cycles (et donc la fréquence) devient de moins en moins précise.



Adapté de (K J Blinowska., J Zygierewicz 2012)

Formellement, nous traitons la représentation de l'énergie du signal dans le temps $|x(t)|^2$ et dans la fréquence $|X(f)|^2$ comme des distributions de probabilité avec un ***facteur de normalisation***

$$E_x = \int_{-\infty}^{+\infty} \left|\mathrm{x}(t)\right|^2 dt$$

Ensuite nous définissons le ***temps moyen***

$$t_m = \frac{1}{E_x} \int_{-\infty}^{+\infty} t \left|\mathrm{x}(t)\right|^2 dt$$

La ***fréquence moyenne***

$$f_m = \frac{1}{E_x} \int_{-\infty}^{+\infty} f \left|\mathrm{x}(f)\right|^2 df$$

L'***étendue de temps***

$$\sigma_t^2 = \frac{1}{E_x}\int_{-\infty}^{+\infty}(t-t_m)^2\left|\mathrm{x}(t)\right|^2 dt$$

L'***étendue de fréquence***

$$\sigma_f^2 = \frac{1}{E_x}\int_{-\infty}^{+\infty}(f-f_m)^2\left|\mathrm{X}(f)\right|^2 df$$

Il s'avère que le produit de l'étendu du temps et l'étendue de la fréquence est borné [Folland and Sitaram, 1997] :

$$\sigma_t^2\sigma_f^2 \geq \frac{1}{16\pi^2}$$

En d'autre terme, le produit de la résolution fréquentielle (exprimé en bande passante Δf) et la résolution temporelle Δt est

$$\Delta f \bullet \Delta t \geq \frac{1}{4\pi}$$

[Folland and Sitaram, 1997] Folland, G. and Sitaram, A. (1997). The uncertainty principle: A mathematical survey. Journal of Fourier Analysis and Applications, 3(3):207–238.

L'égalité est atteinte par des ***fonctions de Gabor*** (enveloppe gaussienne modulée par cosinus). Il est important de vérifier l'inégalité précédente quand on travaille avec des représentations time-fréquence d'un signal. Différentes méthodes de représentations temps-fréquence font divers compromis entre les étendues du temps et de fréquence, mais dans chacune d'elles l'inégalité précédente est vérifiée.

Limites de l'analyse de Fourier

➢L’analyse de Fourier suppose la stationnarité du signal, par exemple, ses propriétés statistiques ne changent pas dans le temps.

Nombreux signaux - en particulier ceux d'origine biologique - ne sont pas stationnaires.

➢ L’analyse de Fourier est unidirectionnelle (temps →fréquence ou fréquence →temps) mais jamais une représentation temps-fréquence.

Limites de l'analyse de Fourier (suite)

Malgré son immense succès, cette technique a plusieurs défauts, en particulier

• son manque évident de **localisation temporelle**. En effet, l'analyse de Fourier permet de connaître les différentes fréquences excitées dans un signal, c'est-à-dire son spectre, mais ne permet pas de savoir à quels instants ces fréquences ont été émises.

Cette analyse donne une **information globale** et non locale, car les fonctions d'analyse utilisées sont des sinusoïdes qui oscillent indéfiniment sans s'amortir. Cette perte de localité n'est pas un inconvénient pour analyser des signaux dont la structure n'évolue pas ou peu (statistiquement stationnaires), mais devient un problème pour l'étude de signaux non stationnaires.

Un grand nombre d'approches ont été développées pour tenter d'extraire à la fois l'information temps-fréquence à partir d'un signal non stationnaire. Fondamentalement, elles peuvent être divisées en deux groupes:

- Analyse ***temps-fréquence*** :

➢***Transformée de Fourier à court terme*** ou ***Spectrogramme*** (en anglais *Short-term Fourier Transform* (*STFT*) ou *spectrogram*).

Transformée de Fourier à court terme ou ***Spectrogramme*** est une méthode où il est difficile de trouver un compromis temps-fréquence avec une bonne résolution. C'est pour cette raison, il existe d'autres approches comme la ***distribution Wigner-Ville***.

En dépit de ces limitations, la STFT a été utilisée avec succès dans une grande variété de problèmes, en particulier ceux où seules les composantes hautes fréquences sont d'un intérêt et quand la résolution en fréquence n'est pas critique.

▪ Analyses ***temps-échelle*** (***transformée en ondelettes*** (en anglais ***wavelet transform (WT)***).

En 1982, Morlet ouvre la voie conduisant à la représentation temps-fréquence en construisant la **transformation en ondelettes (wavelet)** qui consiste à décomposer un signal sur une base de fonctions comme le ferait une décomposition en séries de Fourier. Cette approche nécessite que la base de fonctions soit donnée a priori.

La *STFT* donne une résolution fixe pour tous les instants de temps alors que la *WT* donne une résolution variable.

• Stéphane Mallat, "A wavelet tour of signal processing" 2nd Edition, Academic Press, 1999, ISBN 0-12-466606-X

• Ingrid Daubechies, *Ten Lectures on Wavelets*, Society for Industrial and Applied Mathematics, 1992, ISBN 0-89871-274-2

Limites de l'analyse de Fourier (suite)

▪ La décomposition en modes empiriques (***empirical mode decomposition*** (***EMD***)) consiste à décomposer un signal sur une base de fonctions comme le ferait une décomposition en séries de Fourier ou une décomposition en ondelettes. La particularité de l'***EMD*** réside dans le fait que la base de fonctions n'est pas donnée a priori mais est construite à partir des propriétés du signal. Il a été observé que ces fonctions sont, avec une bonne précision, orthogonales entre elles et de somme égale au signal d'origine.

- Huang NE, Shen Z, Long SR, et al. The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis. Proc R Soc London 1998; 454:419 903-95.
- Huang NE. Introduction to Hilbert-Huang transform and its related mathematical problems. In: Huang NE, Shen SSP, editors. Hilbert-Huang transform and its applications, 2005, pp. 1-26.

Limites de l'analyse de Fourier (suite)

La combinaison de l'algorithme d'EMD et de la ***Transformée de Hilbert*** (***TH***) pour l'estimation de l'***Amplitude Instantanée*** (***AI***) et de la ***Fréquence Instantanée*** (***FI***) est appelée ***Transformation de Hilbert - Huang*** (***THH***).

- Huang NE, Shen Z, Long SR, et al. The empirical mode decomposition and the Hilbert spectrum for nonlinear and non-stationary time series analysis. Proc R Soc London 1998; 454:419 903-95.
- Huang NE. Introduction to Hilbert-Huang transform and its related mathematical problems. In: Huang NE, Shen SSP, editors. Hilbert-Huang transform and its applications, 2005, pp. 1-26.

Dans cette partie du cours, nous présentons uniquement la ***Transformée de Fourier à court terme*** ou ***Spectrogramme***.

Transformée de Fourier à court terme ou ***Spectrogramme***

Nous sommes intéressés par la description de la modification des différentes fréquences en fonction du temps.

Une approche simple est d'évaluer le spectre de puissance dans une fenêtre temporelle glissante appliquée sur le signal et de calculer la *TF* et représenter le spectre de puissance en fonction du temps. Ce genre de représentation s'appelle un ***spectrogramme*** et peut être stocké dans une matrice. Ses caractéristiques sont déterminées par la fenêtre que nous appliquons au signal.

Transformée de Fourier à court terme ou ***Spectrogramme*** **(suite)**

- Effectuer une transformation de Fourier (TF) sur une fenêtre *w* de données le long de l'axe de temps → une fonction temps-fréquence qui décrit la distribution de fréquence proche de temps τ

$$STFT(f,\tau) = TF(x(t)\cdot w(t-\tau))$$

- Le spectrogramme est donnée par l'amplitude de la fonction *STFT*

$$Spectrogram\ \{x(t)\} = STFT(f,\tau)^2$$

Spectrogramme

Exemple

Nous allons étudier l'évolution des fréquences présentent dans 4 signaux cosinus : le 1er 0 =< t < 5s, le 2ème 5 =< t < 10s, le 3ème 10 =< t < 15s et le 4ème 15 =< t < 20s

```
x=[cos(2*pi*10*t1) cos(2*pi*25*t1) cos(2*pi*50*t1)
   cos(2*pi*100*t1)];

Te=0.0025; %Période d'échantillonnage
```

source :

../simulation2/ spectrogramme/Matlab /spectrogram_sim.m

Spectrogramme

Exemple

Nous allons étudier l'évolution des fréquences présentent dans le signal ECG à partir de « ecg » (la variable que nous avons sauvée dans la première partie de ce cours au sujet du format EDF).

source :

../simulation2/ spectrogramme/Matlab /spectrogram_ecg.m

Puisque nous sommes intéressés par les changements de la fréquence à l'intérieur de chaque battement, nous devons employer une fenêtre courte. Nous allons employer une fenêtre de longueur 40 qui correspond à 0.4 s.

Spectrogramme

Modification de la fonction « pspect » pour générer un spectrogramme

La fonction « pspect.sci » qui se trouve dans `macros/signal/' applique une fenêtre à un ou deux signaux et calcule son spectre de puissance. Enfin tous les gros morceaux considérés sont ramenés à une moyenne.

Ses options sont l'offset de chaque fenêtre, la longueur de la fenêtre et le type de fenêtrage appliqué. Il agit sur deux signaux. Le rendement de cette fonction est un vecteur contenant la moyenne du spectre de puissance de tous les gros morceaux. Nous allons modifier seulement deux lignes de la définition de la fonction et du rendement de la fonction.

Spectrogramme

Modification de la fonction « pspect » pour générer un spectrogramme

Nous allons modifier seulement deux lignes de la définition de la fonction et du rendement de la fonction. Nous voulons que la nouvelle fonction renvoie tous les spectres de puissance intermédiaires au lieu de leur moyenne. Ainsi nous allons éditer seulement deux lignes:

Avant : fonction pspect.sci

```
function [sm,cwp]=pspect (sec_step,sec_leng,wtype,x,y,wpar) // Ligne 1
 ...
sm=sm+real(fx.*conj(fx)); // Ligne 94
 ...
```

Après :fonction spectrogram.sci

```
function [sm,cwp]=spectrgram (sec_step,sec_leng,wtype,x,y,wpar) // Nouvelle ligne 1
...
sm=[sm ; real(fx.*conj(fx))]; // Nouvelle ligne 94
...
```

Spectrogramme

Exemple

Nous allons étudier l'évolution des fréquences présentent dans le signal ECG à partir de « ecg » (la variable que nous avons sauvée dans la première partie de ce cours au sujet du format EDF).

source : ../simulation2/spectrogramme/Scilab/spectrogram.sci

source : ../simulation2/spectrogramme/Scilab/ecg_spgm.sce

```
stacksize(20000000);
load (signals);
ecg = signals(12)(1:300);
getf('spectrogram.sci');
```

Puisque nous sommes intéressés par les changements de la fréquence à l'intérieur de chaque battement, nous devons employer une fenêtre courte. Nous pouvons définir le degré de recouvrement en employant le paramètre d'entrée « sec_step ». Nous avons décidé d'employer une étape d'un échantillon (sec_step=1). Nous allons employer une fenêtre de longueur 40 qui correspond à 0.4 s.

Spectrogramme Exemple (suite)

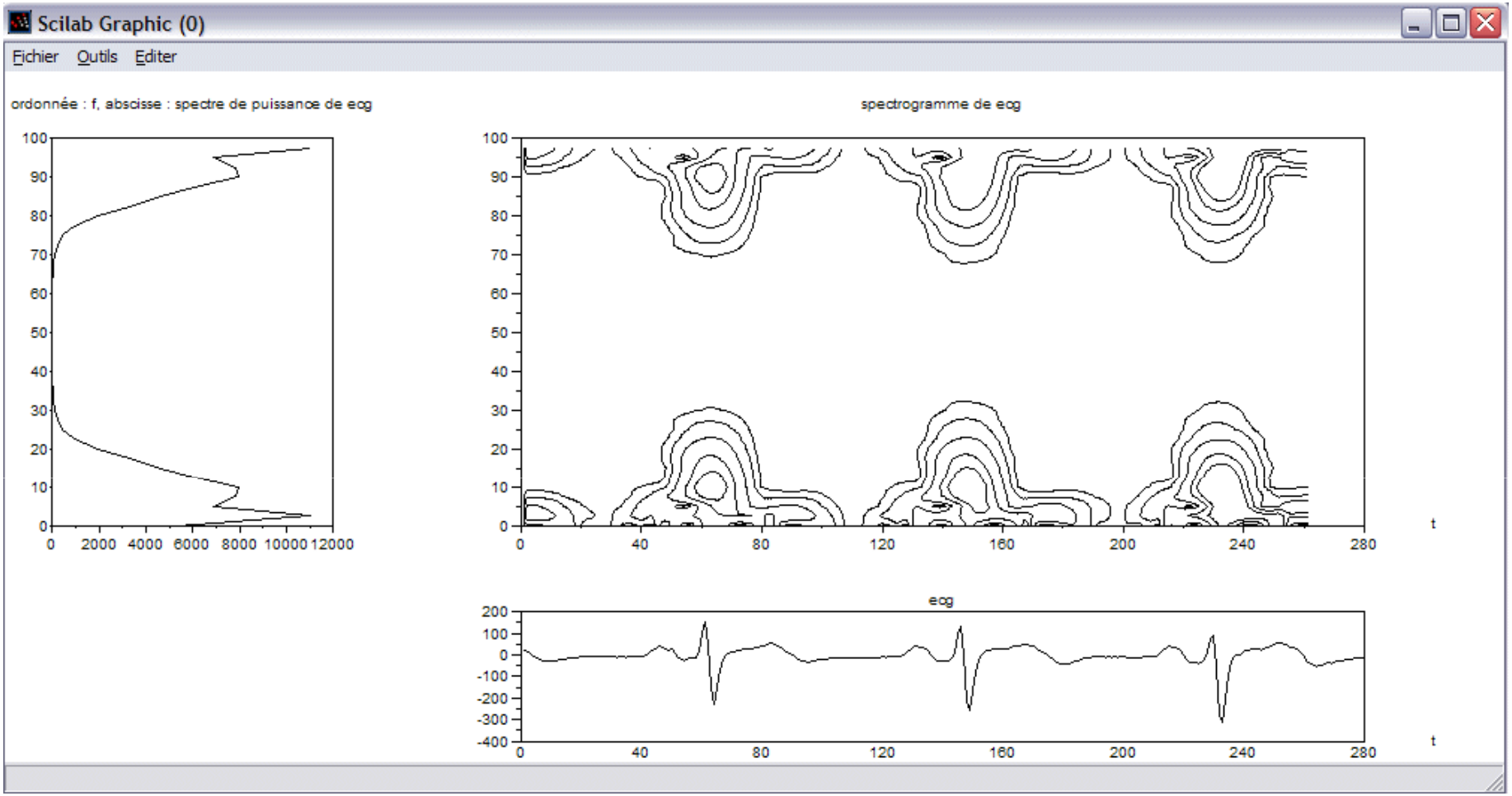
```
N = 40;
    f = (0:N-1)*(1/(N*0.01));
    ecg_spgrm = spectrogram(1,N,'hn',ecg);
    Pxx = pspect(1,N,'hn',ecg);
```

Avec ce code nous avons calculé le vecteur ligne de fréquence (pour tracer l'axe des abscisses dans les graphiques) avec une longueur de 40 échantillons et à fréquence d'échantillonnage de 0.01s. Ensuite, nous utiliserons « spectrogram » pour calculer le spectrogramme et « pspect » pour calculer le spectre de puissance de signal ecg avec les mêmes options de la fenêtre (step = 0.01 ms, length = 0.4 ms, 'hm'.

Pour représenter les valeurs de la matrice, nous pouvons utiliser différentes représentations graphiques. Pour cet exemple nous utilisons 'contour ' :

```
xsetech([0,0,1/4,3/4])
plot2d(Pxx ,f )
xtitle('spectrogramme de ecg', 'f');
xsetech([1/4,3/4,3/4,1/4],[1,-400,275,200])
h= ecg(N/2:$);
plot2d(1:length(h),h,1,'002');
xtitle('ecg', 't');
xsetech([1/4,0,3/4,3/4])
xset('fpf',' ');
levels=[1,5,20,60,100]; contour2d(1:size(ecg_spgrm,1),f,ecg_spgrm,levels,ones(1,5)) ;
xtitle('spectrogramme de ecg', 't');
```

Spectrogramme Exemple (suite)



Le spectrogramme peut être utilisé pour déterminer où les différentes fréquences sont localisées.

Spectrogramme Exemple (suite)

Voici une représentation plus claire en utilisant la FFT sur trois points représentatifs : t= 53, 63 et 93. La fenêtre de Hamming (N=40) est appliquée à chaque point, ensuite la FFT est calculée.

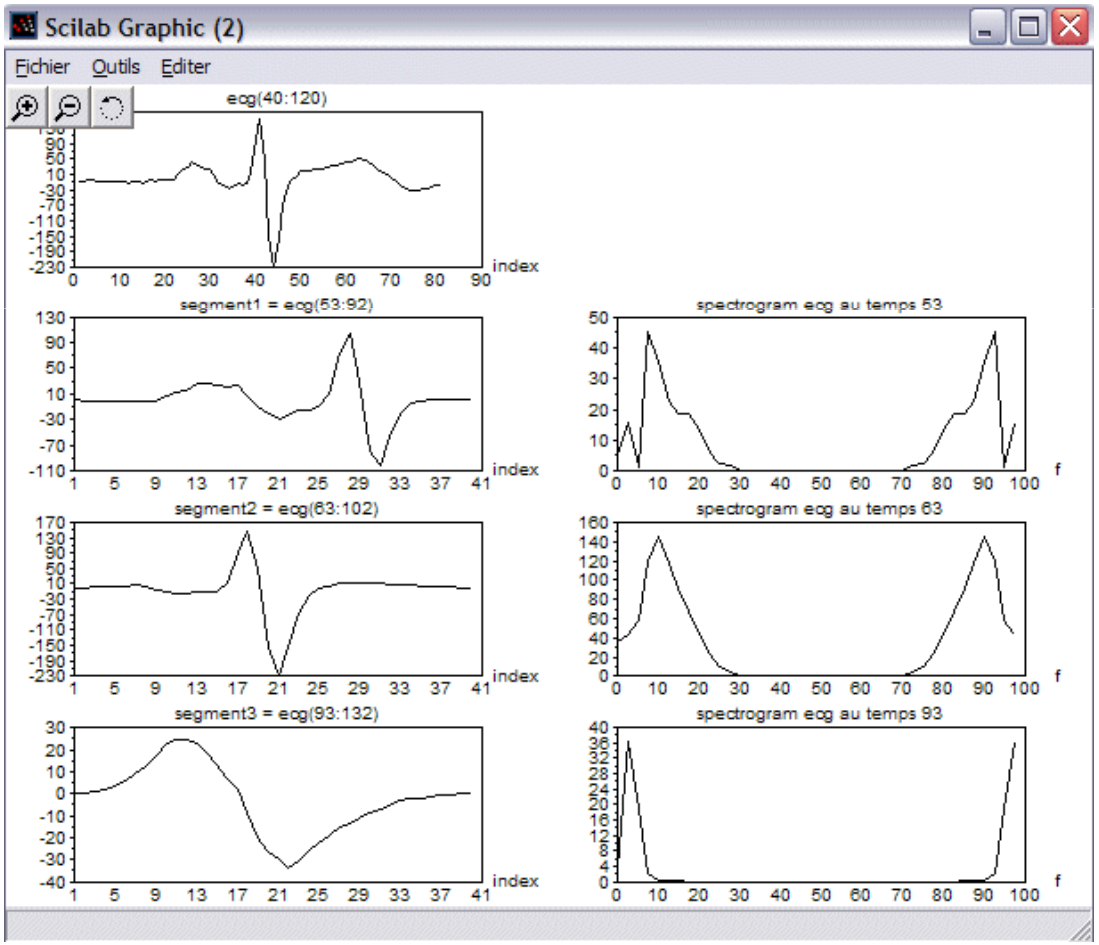
```
fenêtre = window('hn',N);
pos1 = 53;
pos2 = 63;
pos3 = 93;
ind11=string(pos1);ind12=string(pos1+N-1);
ind21=string(pos2);ind22=string(pos2+N-1);
ind31=string(pos3);ind32=string(pos3+N-1);
segment1 = ecg(pos1:pos1+N-1) .* fenêtre;
segment2 = ecg(pos2:pos2+N-1) .* fenêtre;
segment3 = ecg(pos3:pos3+N-1) .* fenêtre;
xset('window',2);xsetech([0,0,1/2,1/4]);plot(ecg(40:120));xtitle('ecg(40:120)', 't');
xsetech([0,1/4,1/2,1/4]);plot(segment1);
xtitle('segment1 = ecg('+ind11+':'+string(ind12+') ', 'index');
xsetech([1/2,1/4,1/2,1/4]);plot2d(f,ecgspgrm(pos1,1:$));xtitle('spectrogram ecg de '+string(pos1), 'f');
xsetech([0,2/4,1/2,1/4]);plot(segment2);
xtitle('segment2 = ecg('+ind21+':'+string(ind22+') ', 'index');
xsetech([1/2,2/4,1/2,1/4]);plot2d(f,ecgspgrm(pos2,1:$));xtitle('spectrogram ecg de '+string(pos2), 'f');
xsetech([0,3/4,1/2,1/4]);plot(segment3);
xtitle('segment3 = ecg('+ind31+':'+string(ind32+') ', 'index');
xsetech([1/2,3/4,1/2,1/4]);plot2d(f,ecgspgrm(pos3,1:$));xtitle('spectrogram ecg de '+string(pos3), 'f');
```

Spectrogramme Exemple (suite)

Cette figure essaye de montrer l'effet du fenêtrage sur un signal. Nous pouvons voir que le fenêtrage permet de concentrer l'analyse sur quelques segments spécifiques de la trace et, par conséquent, les fréquences contenues dans cet espace particulier peuvent être soumises à une contrainte.

Le trame supérieure à gauche montre le signal d'origine. La deuxième rangée des trames souligne l'onde P, le troisième souligne le QRS complexe et le quatrième souligne le composant T. Notez que le spectre change en conséquence.

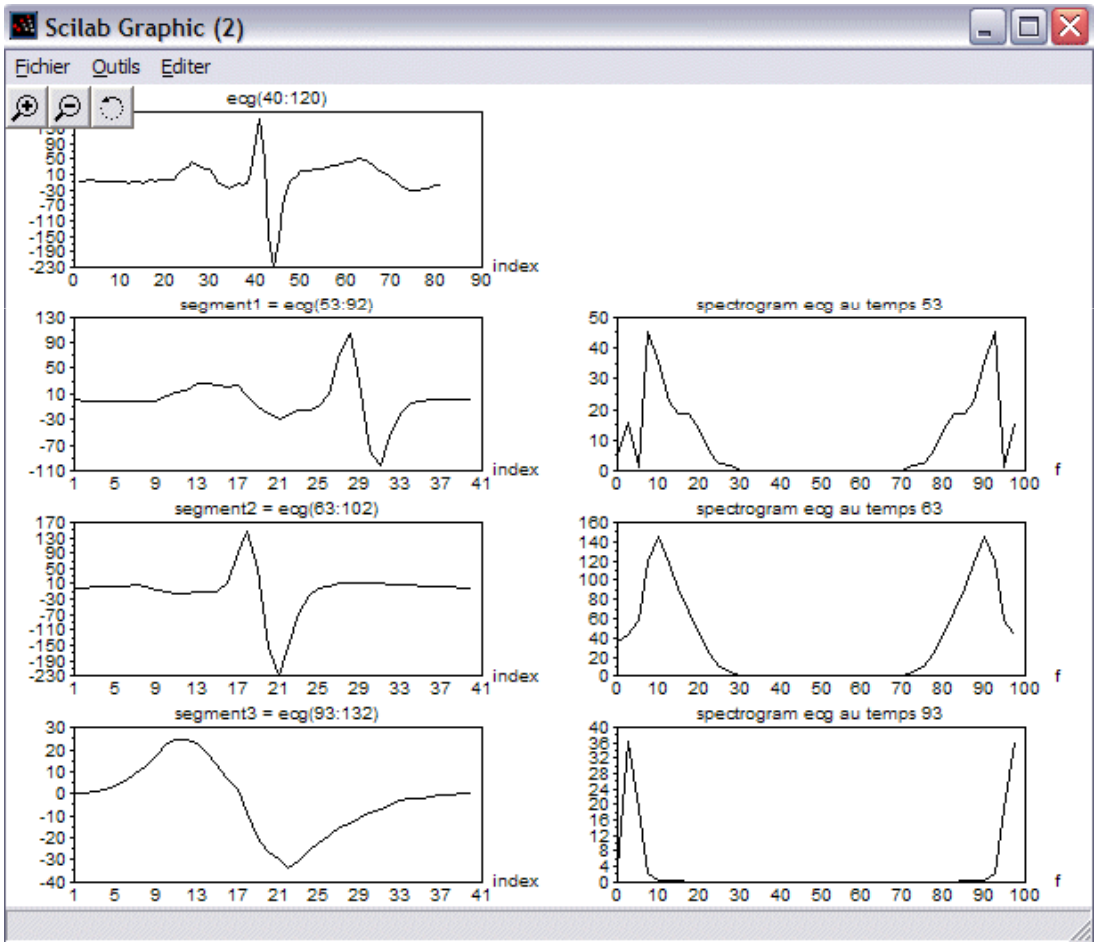
Les vecteurs contenant le spectre sont certaines des colonnes de la matrice représentée dans le tracé du contour de la figure précédente.



Spectrogramme Exemple (suite)

Le résultat prouve que la fenêtre focalise la transformée de Fourier à quelques composants. Nous pouvons voir que l'énergie des ondes P et T est la plupart du temps concentrée au-dessous de 5 Hz tandis que l'énergie des complexes QRS de 5 à 30 hertz.

Pour obtenir ce résultat, le choix de la longueur appropriée de la fenêtre est critique puisqu'une fenêtre plus large pourrait mélanger les fréquences des différentes ondes tandis qu'une fenêtre plus courte pourrait manquer la résolution de fréquence pour montrer les changements.



4.5. Cohérence

Nous avons vu que le spectre de puissance représente la puissance portée par chaque fréquence. Quand nous avons deux signaux, nous pouvons **vérifier le degré de corrélation croisée dans le domaine fréquentiel**. Pour évaluer ce degré nous employons la **cohérence** (coherence).

La cohérence spectrale est une statistique qui peut être utilisé pour examiner la relation entre deux signaux ou ensembles de données.

Cohérence

La cohérence entre deux signaux *x* et *y* est une mesure normalisée, la densité spectrale croisée de puissance est divisé par la racine carrée du produit des densités autospectrales de signaux. La ***Magnitude Carrée de la Cohérence*** (***Magnitude Squared Coherence*** (***MSC***)) est fréquemment employée.

MSC = le carré de spectre de puissance croisé divisé par le produit des spectres de puissance de deux signaux.

$$C_{xy}(\omega) = \frac{\left|P_{xy}(\omega)\right|^2}{P_{xx}(\omega)P_{yy}(\omega)}, 0 \leq C_{xy} \leq 1$$

où P_{xy} est la densité spectrale croisée de puissance entre *x* et *y*, et P_{xx} et P_{yy} les densités autospectrales de *x* et *y* respectivement.

Une valeur 0 pour une fréquence donnée signifie que les deux signaux sont complètement non-corrélés dans cette fréquence, alors qu'une valeur de 1 signifie que, dans cette fréquence, les signaux sont complètement corrélés.

Cohérence

Un signal corrélé avec un autre ne signifie pas seulement qu'une certaine énergie dans une fréquence est présente dans les deux signaux mais que la fréquence actuelle dans les deux signaux est liée par la phase. La cohérence du bruit blanc tend vers 0.

En pratique, pour calculer la cohérence de deux signaux nous calculons le spectre de puissance croisé (cross-power spectrum) de deux signaux avec « pspect ».

Cohérence

Exemple

Nous allons considérer certains signaux issus de fichier « signals » qui présentent certaine cohérence. Ce fichier contient un enregistrement de sommeil avec canaux d'ECG. Nous allons calculer la cohérence entre trois paires de segments de signaux. Nous choisissons les segments contenus dans les canaux 8, 9, 11 et 12, qui contiennent EOG (canaux 8 et 9), EMG de jambes liées (11) et ECG (12).

source Matlab:

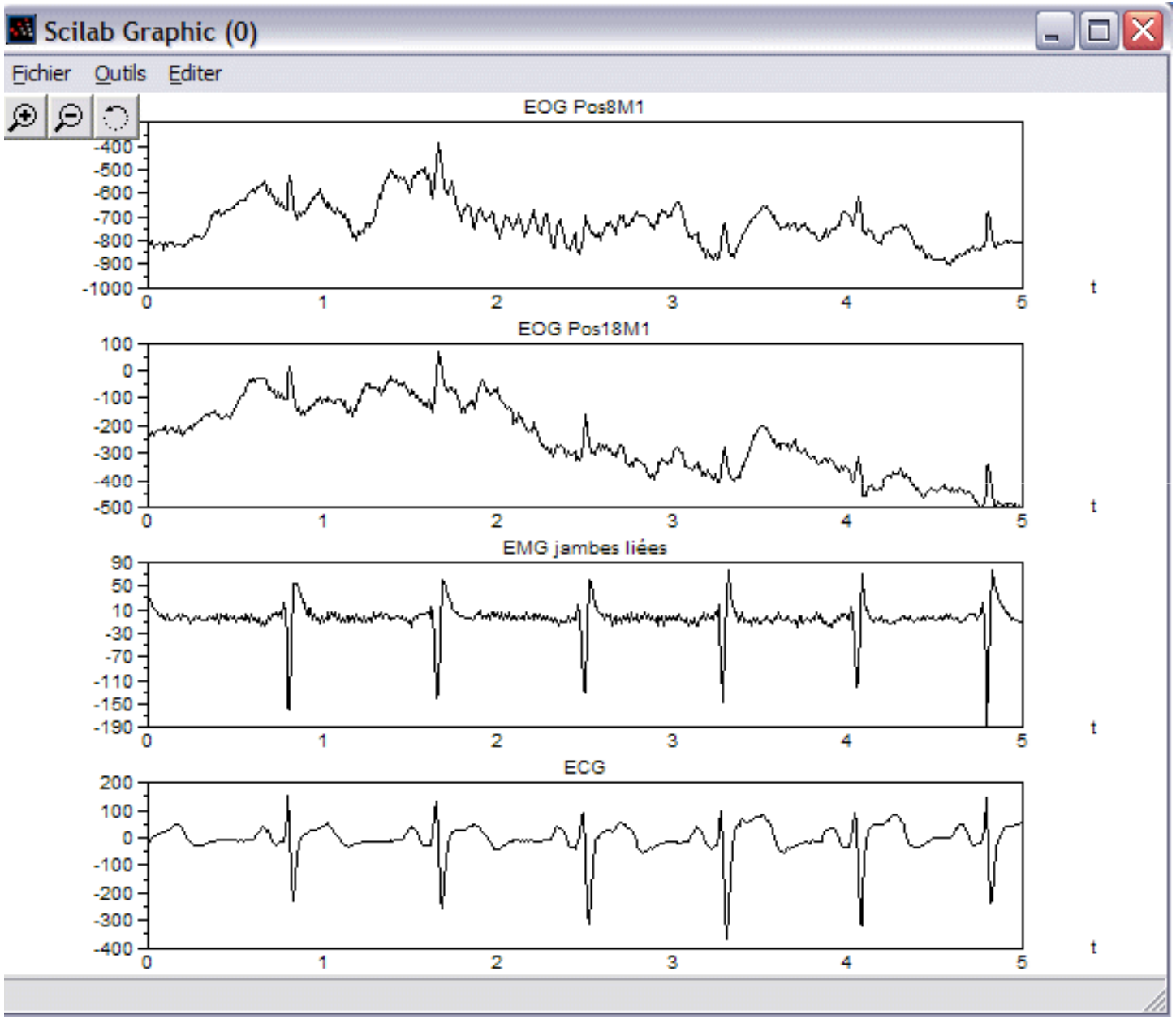
../simulation2/filtrage_densité_spectrale_cohérence/Matlab/coherence_4signaux_réels/
coherence_4signaux_réels.m

Cohérence Exemple (suite)

source : ../simulation2/filterage_densité_spectrale_cohérence/Scilab/coherence_4signaux_reels.sce

```
stacksize(20000000)
my_dir=get_absolute_file_path('coherence.sce');
path=string(my_dir);
stacksize(20000000)
my_dir=get_absolute_file_path('coherence.sce');
path=string(my_dir);
// Charger les signaux
load(path+'signals');
getf(path+'spectrogram.sci');
x1 = signals(8);    // EOG Pos8M1
x2 = signals(9);    // EOG Pos18M1
y1 = signals(11);   // EMG linked legs
y2 = signals(12);   // ECG
inct = 0.01;
incf = 1 / (100 * inct) ;
segment = [1:500];
tscale = segment * inct;
xsetech([0,0,1,1/4]); plot2d(tscale ,x1(segment)); xtitle('EOG Pos8M1','t');
xsetech([0,1/4,1,1/4]) plot2d(tscale,x2(segment)); xtitle('EOG Pos18M1','t');
xsetech([0,2/4,1,1/4]) plot2d(tscale,y1(segment)); xtitle('EMG jambes liées','t');
xsetech([0,3/4,1,1/4]) plot2d(tscale,y2(segment));xtitle('ECG','t');
```

Cohérence Exemple (suite)



Cohérence Exemple (suite)

Maintenant nous allons calculer la cohérence entre les canaux d'EOG et la cohérence entre l'EMG et les signaux d'ECG.

```
psx1 = pspect(50,100,'hn',x1); // spectre de EOG Pos8M1
psx2 = pspect(50,100,'hn',x2); // spectre de EOG Pos18M1
psx12 = pspect(50,100,'hn',x1,x2); // spectre croisé de EOG Pos8M1 et EOG Pos18M1
psy1 = pspect(50,100,'hn',y1); // spectre de EMG jambes liées
psy2 = pspect(50,100,'hn',y2); // spectre de ECG
psy12 = pspect(50,100,'hn',y1,y2); // spectre croisé de y1 et y2
psxy = pspect(50,100,'hn',x1,y2); // spectre croisé de x1 et y2
cohx = ( (abs(psx12) ^ 2) ./ (psx1 .* psx2) );// coherence de EOG Pos8M1 et EOG Pos18M1
cohy = ( (abs(psy12) ^ 2) ./ (psy1 .* psy2) );// coherence de EMG et ECG
cohxy = ( (abs(psxy) ^ 2) ./ (psx1 .* psy2) );// coherence de EOG Pos8M1 et ECG
f = [0:49] * incf;
```

```
// Spectre des signaux EOG Pos8M1 et EOG Pos18M1 d'origine et leur cohérence
xbasc(); xsetech([0,0,1/3,1/3]); plot2d (f,20*log10(psx1(1:50))) ;
xtitle('spectre de EOG Pos8M1','f'); xsetech([1/3,0,1/3,1/3]);
plot2d (f, cohx(1:50),1,'011',' ', [0,0,50,1] ) ;
xtitle('cohérence de EOG Pos8M1 et EOG Pos18M1','f');
xsetech([2/3,0,1/3,1/3]); plot2d (f,20* log10( psx2(1:50))) ;
xtitle('spectre de EOG Pos18M1','f');
```

Cohérence Exemple (suite)

```
// Spectre et cohérence des signaux EMG des jambes et ECG
xsetech([0,1/3,1/3,1/3]); plot2d (f,20*log10( psy1(1:50))) ;
xtitle('spectre de EMG jambes','f');
xsetech([1/3,1/3,1/3,1/3]); plot2d (f, cohy(1:50),1,'011',' ', [0,0,50,1] ) ;
xtitle('cohérence de EMG jambes et ECG','f');
xsetech([2/3,1/3,1/3,1/3]); plot2d (f,20* log10( psy2(1:50))) ;
xtitle('spectre de ECG','f');
```

```
// Spectre et cohérence des signaux coherence de EOG Pos8M1 et ECG
xsetech([0,2/3,1/3,1/3]); plot2d (f,20*log10( psx1(1:50))) ;
xtitle('spectre de EOG Pos8M1','f');
xsetech([1/3,2/3,1/3,1/3]); plot2d (f, cohxy(1:50),1,'011',' ', [0,0,50,1] ) ;
xtitle('cohérence de EOG Pos8M1 et ECG','f');
xsetech([2/3,2/3,1/3,1/3]); plot2d (f,20* log10( psy2(1:50))) ;
xtitle('spectre de ECG','f');
```

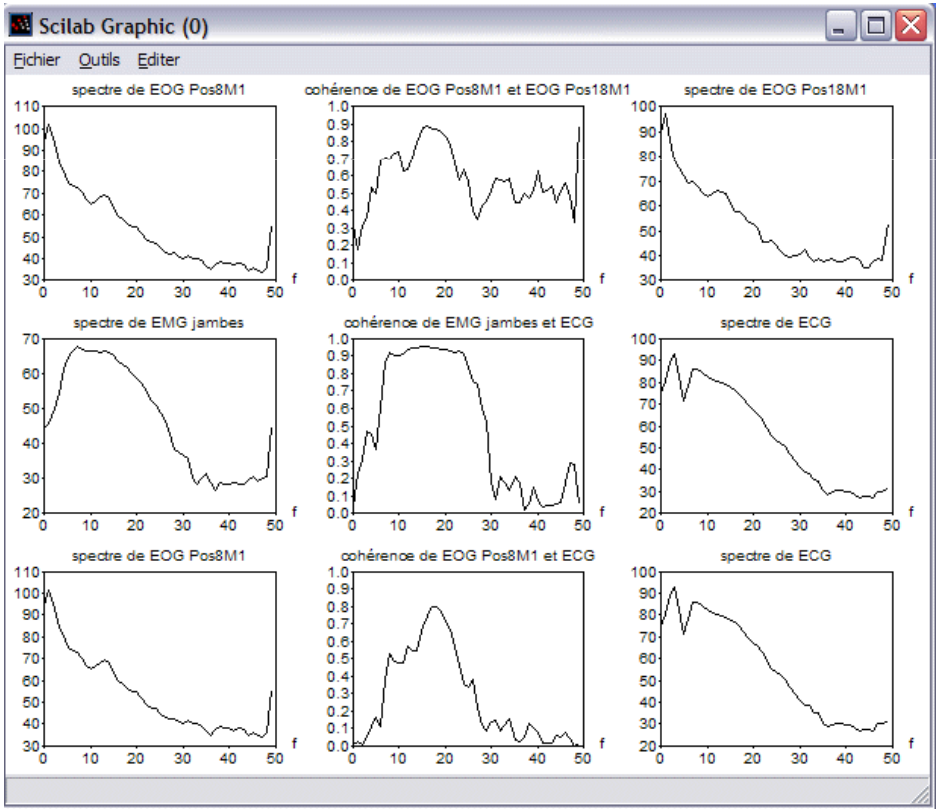
Cohérence Exemple (suite)

Pour chaque rangée, les spectres de deux signaux sont représentés dans l'échelle logarithmique à droite et à gauche de la cohérence, qui est représentée au milieu.

Nous pouvons voir que la cohérence est très haute quand nous considérons les deux canaux EOG. Quand nous considérons la cohérence entre l'EMG et l'ECG nous pouvons voir qu'il est très haut entre 5 et 25 Hz.

Comme nous pourrions voir dans le paragraphe précédent, l'activité du complexe de QRS est située principalement dans cette bande.

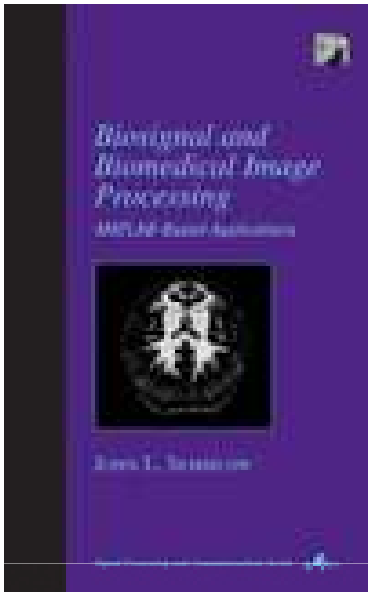
Une cohérence semblable est présente entre EOG et ECG bien que la présence des complexes de QRS dans la trace d'EOG soit moins évidente.



Références

1. Scilab : http://www.scilab.org/
2. EDF : http://www.hsr.nl/edf/
3. Jesus Olivan Palacios, “ interface Scilab/EDF ” : ftp://ftp.neurotraces.com/pub/edf/
4. Some general concepts of signal treatment applied to Clinical Neurophysiology, Jesus Olivan Palacios
5. Physionet : http://physionet.org/
6. Cours Traitement et analyse des signaux bio-médicaux – partie 1 : http://www.esiee.fr/~alanit
7. Gilles ZWINGELSTEIN. Diagnostic des défaillances, théorie et pratique pour les systèmes industriels. Ed. HERMES, 1995.
8. http://www.techno-science.net/
9. Signal processing Course, W. Penny, http://www.fil.ion.ucl.ac.uk/~wpenny/course/course.html
10. K J Blinowska., J Zygierewicz,. Practical Biomedical Signal Analysis Using Matlab. 2012 by Taylor & Francis Group, LLC.

Biosignal and Biomedical Image Processing
MATLAB-Based Applications
John L . Semmlow
CRC Press 2004
Print ISBN: 978-0-8247-4803-6
eBook ISBN: 978-0-203-02405-8



Signal Processing for Neuroscientists
Introduction to the Analysis of Physiological Signals
Wim van Drongelen
Elsevier Inc.
ISBN: 978-0-12-370867-0