

الجمهورية الجزائرية الديمقراطية الشعبية  
République Algérienne démocratique et populaire

وزارة التعليم العالي والبحث العلمي  
Ministère de l'enseignement supérieur et de la recherche scientifique

جامعة سعد دحلب البليدة  
Université SAAD DAHLAB de BLIDA

كلية التكنولوجيا  
Faculté de Technologie

قسم الإلكترونيك  
Département d'Électronique



## Mémoire de Master

Filière Électronique

Spécialité Traitement de l'Information et Systèmes Électroniques

présenté par

TOUAIBIA Moufida

&

AKKACHE Ouardia

---

# Implémentation de la méthode de séparation de source aveugle FAST ICA sous XSG

---

Proposé par : BOUGHERIRA Nadia & KAOULA Ikram

Année Universitaire 2017-2018

## Remerciements

---

Nous remercions tout d'abord ALLAH le tout puissant de nous avoir permis de rendre nos parents fiers et de nous avoir aidées à réaliser notre travail.

Au terme de ce travail nous tenons à témoigner notre profonde reconnaissance et nos vifs remerciements à notre Encadreur **Madame N. BOUGHERIRA** et madame KAOULA de nous avoir encadrée et conseillée tout au long de notre travail sur notre projet de fin d'études.

Nous tenons aussi à exprimer notre profonde gratitude à Mr Chikhi, Mme H. Bougherira, Mr Rellam, Mr Ferdjouni et Mr Mamoune pour tous leurs aides et leurs conseils.

Enfin, nous remercions aussi tous nos amis et camarades de promotion ainsi que tous ceux qui ont contribué de près ou de loin à la réalisation de ce travail.

Nous remercions chaleureusement les membres du jury pour l'honneur qu'ils nous ont fait en acceptant d'évaluer notre projet.

## *Dédicace*

*Je remercie tout d'abord ALLAH de m'avoir aidée à achever ce  
modeste travail.*

*J'ai le grand plaisir de dédier les résultats de mes années  
d'études aux êtres les plus chers que DIEU me les  
protège.*

*A ma très chère mère, autant de phrases aussi expressives  
soientelles ne sauraient montrer le degré d'amour que j'éprouve  
pour  
toi. Tu as toujours été présente à mes côtés, tu n'as cessé de me  
soutenir et de m'encourager, ainsi à mn cher père, je vous  
remercie pour tout le soutien et l'amour que vous me portez. Que  
ce modeste travail soit l'exaucement de vos vœux tant formulés,  
le fruit de vos innombrables sacrifices, bien que je ne vous en  
acquitte jamais assez.*

*A mes chères précieuses sœurs Amina, Khadidja qui m'ont  
toujours soutenu, les adorables jumelles Safa et Maroua et à mon  
cher frère AMINE, pour toute ambiance dont vous m'avez  
entouré, pour toute spontanéité et complexité qui nous unissent  
je*

*vous dédie ce travail.*

*A mon cher beau-frère Aksill Houcine*

*A mes chers Ahlem, Nassima et Khaled*

*A tous mes chers amis Ouardia, Nadjet, Karima, Mus, Chakib,  
Siradj, Abdellah, Zohir, Ali, Chemsedine et Othmane*

*Moufida...*

*Je remercie tout d'abord ALLAH de m'avoir aidée à achever ce  
modeste travail.*

*Je dédie ce travail à tous ceux qui m'aiment*

*A mes chers parents, je vous remercie pour tout le soutien et  
l'amour que vous me portez. Vos conseils ont toujours guidé mes  
pas vers la réussite. Votre patience sans fin, votre  
compréhension  
et encouragement sont pour moi le soutien indispensable que  
vous avez toujours su m'apporter. Que dieu le tout puissant  
vous préserve.*

*A la mémoire de ma chère grande mère paternel Malika*

*A ma chère grande mère maternel Ghnima*

*A mes frères Sofiane, Farid et Sidahmed*

*A mes deux chères petites sœurs Fairouz et Meriem*

*A tous mes chers amis, Moufida, Imen, Yasmine*

*Ouardia...*

---

**ملخص:** تحليل المكونات المستقلة (ICA) هو تقنية معالجة إشارات إحصائية تنبثق من مجالات التطبيق العملية الجديدة ، مثل فصل الإشارة المكفوف ، مثل الأصوات أو الصور المختلطة ، في مشروعنا سنقوم بذلك دراسة هذه الطريقة ، ثم نفذت تحت Matlab و Matlab Simulink كخطوة تمهيدية من أجل تطبيقه في النهاية مولد نظام xilinx

**كلمات المفاتيح:** التحليل المستقل للمكونات , فصل الإشارة , Matlab و Matlab Simulink

---

**Résumé :** L'analyse de composants indépendants (ICA) est une technique de traitement de signal statistique émergents de nouveaux domaines d'application pratiques, tels que la séparation des signaux à l'aveugle, comme les voix mixtes ou images, dans notre projet on va faire l'étude de cette méthode, ensuite l'implémenté sous Matlab et Matlab Simulink comme étape préparatives afin de pouvoir l'implémenter finalement xilinx system generator.

**Mots clés :** L'analyse de composants indépendants ; séparation de sources ; Matlab ; Simulink ; xilinx system generator.

---

**Abstract:** Independent Component Analysis (ICA) is a statistical signal processing technique having emerging new practical application areas, such as blind signal separation such as mixed voices or images, in our project we will do the study of this method, then implemented under MATLAB and MATLAB Simulink as preparatory step in order to finally implement it in xilinx system generator.

**Keywords:** Independent component analysis; separate the independent sources; Matlab; Simulink ; xilinx system generator.

---

## Listes des acronymes et abréviations

**ICA** Analyse des composants indépendants. **FAST ICA**  
Analyse des composants indépendants.

**PCA** Analyse principal des composants.

**BSS** Blind source separation (séparation de source aveugle).

**MEG** Magnétoencéphalographie.

**ECG** Electrocardiogramme.

**EEG** Electroencéphalogramme.

**FHR** Fréquence cardiaque fœtal.

**FECG** Analyse morphologique de Electrocardiogramme fœtal.

**ADS** Signaux abdominaux.

**SNR** rapport signal sur bruit.

**CDMA** Code division multiple access.

**FPGA** Field programmable gate array.

**XSG** xilinx system generator.

# Table des matières

Introduction générale .....	1
Chapitre 1 ANALYSE INDEPENDANTE DES COMPOSANTS .....	3
1.1 Introduction : .....	3
1.2 Indépendance statistique .....	5
1.3 Théorème de la limite central .....	5
1.4 Paramètres statistiques généraux : non gaussianité et kurtosis .....	6
1.5 Prétraitement des données pour ICA (Analyse en composantes principales PCA).....	7
1.6 Application de la Fast ICA pour un composant indépendant .....	9
1.7 Domaines d'applications de l'ICA .....	11
1.8 Conclusion .....	13
Chapitre 2 GENERALITES SUR L'ELECTROCARDIOGRAPHIE ET PLATFOME D'IMPLEMENTATION .....	14
2.1 Introduction .....	14
2.2 L'électrocardiographie .....	15
2.2.1 Introduction .....	15
2.2.2 L'électrocardiographie .....	15
2.2.3 Principe de fonctionnement .....	15
2.2.4 Appareillage .....	16
2.2.5 Chaine d'acquisition des données de l'ECG .....	17
2.2.6 Structure de la peau .....	18
2.2.7 Les électrodes .....	19
2.2.8 Types de bruits dans le signal ECG .....	20
2.2.9 ECG foetal .....	21
2.3 plate forme d'implémentation.....	21
2.3.1 Introduction .....	21
2.3.2 Matlab Simulink .....	22
2.3.3 Field programmable gate array (FPGA) .....	22
2.3.4 Constitution interne .....	24
2.3.5 Le circuit XC3SD1800A-4FG676C .....	24
2.3.6 SYSTEM GENERATOR .....	25
2.3.7 Installation .....	26

2.4 Conclusion .....	29
Chapitre 3 IMPLEMENTATION DE L'ALGORITHME FAST ICA RESULTATS ET DISCUSSIONS .....	30
3.1 Introduction .....	30
3.2 Implémentation du centrage .....	31
3.3 Implémentation du blanchiment .....	31
3.4 Calcule de la norme .....	34
3.5 Implémentation de la FAST ICA .....	34
3.6 Implémentation sous Matlab .....	36
3.6.1 Le blanchiment .....	38
3.6.2 La méthode FAST ICA .....	41
3.7 Implémentation sous Matlab Simulink .....	42
3.7.1 Le blanchiment .....	44
3.7.2 La méthode FAST ICA .....	47
3.8 Simulation sous system generator .....	50
3.8.1 Implémentation hard .....	51
3.9 Conclusion .....	56
Conclusion générale .....	57
Annexes .....	58
Bibliographie .....	65



## Liste des figures

<b>Figure 1.</b> L'exemple de séparation de sources de mélanges instantanés.....	2
<b>Figure 1.1</b> séparation de source aveugle.....	4
<b>Figure 1.2</b> Organigramme de l'algorithme FAST ICA.....	11
<b>Figure 2.1</b> électrographe analogique.....	16
<b>Figure 2.2</b> électrographe numérique.....	17
<b>Figure 2.3</b> structures de la peau.....	19
<b>Figure 2.4</b> Les électrodes autocollantes.....	20
<b>Figure 2.5</b> schémas d'une carte FPGA.....	22
<b>Figure 2.6</b> bloc logique configurable (CLB).....	24
<b>Figure 2.7</b> un circuit FPGA.....	25
<b>Figure 2.8</b> lancements de l'installation.....	26
<b>Figure 2.9</b> configurations de Matlab avec system generator.....	27
<b>Figure 2.10</b> insertions de licence.....	27
<b>Figure 2.11</b> bibliothèques de system generator.....	28
<b>Figure 3.1</b> schéma fonctionnel de la mise en œuvre du centrage.....	32
<b>Figure 3.2</b> schéma fonctionnel de la mise en œuvre du blanchiment.....	34
<b>Figure 3.3</b> schéma fonctionnel du calcul de la norme.....	35
<b>Figure 3.4</b> schéma fonctionnel de l'implémentation de $W$ .....	36
<b>Figure 3.5a</b> signal source sinusoïdale.....	40
<b>Figure 3.5b</b> signal source triangulaire.....	41
<b>Figure 3.6a</b> signal mélange 1 sous Matlab.....	41
<b>Figure 3.6b</b> signal mélange 2 sous Matlab.....	42

<b>Figure 3.7</b> Algorithme de la BSS.....	38
<b>Figure 3.8a</b> signal mélange 1 centré sous Matlab.....	42
<b>Figure 3.8b</b> signal mélange 2 centré sous Matlab.....	43
<b>Figure 3.9a</b> signal mélange 1 blanchis sous Matlab.....	40
<b>Figure 3.9b</b> signal mélange 2 blanchis sous Matlab.....	40
<b>Figure 3.10a</b> Signal mélange 1 séparé sous Matlab.....	42
<b>Figure 3.10b</b> Signal mélange 2 séparé sous Matlab.....	42
<b>Figure 3.11</b> signaux source sous Simulink.....	42
<b>Figure 3.12</b> signaux mélange sous Simulink.....	43
<b>Figure 3.13</b> Schéma bloc du centrage sous Simulink.....	43
<b>Figure 3.14</b> signaux mélange centré sous Simulink.....	44
<b>Figure 3.15</b> Schéma bloc du calcul des éléments de la matrice de covariance.....	44
<b>Figure 3.16</b> Schéma bloc de la détermination de la matrice E.....	45
<b>Figure 3.17</b> Schéma bloc du blanchiment.....	46
<b>Figure 3.18</b> signal mélange blanchie.....	47
<b>Figure 3.19</b> Schéma bloc génération des $W$ .....	47
<b>Figure 3.20</b> Schéma bloc génération des $W_k$ .....	48
<b>Figure 3.21</b> Schéma bloc du calcul des pré-w .....	48
<b>Figure 3.22</b> Schéma bloc de la méthode.....	49
<b>Figure 3.23</b> Signal mélange séparé sur Simulink .....	50
<b>Figure 3.24</b> représentation du point fixe.....	51
<b>Figure 3.25</b> blanchiment sous system generator.....	52
<b>Figure 3.26</b> centrage sous system generator.....	53
<b>Figure 3.27</b> calcule des calcule des éléments de la matrice de covariance $c_x$ sous system generato.....	53
<b>Figure 3.28</b> calcule des éléments de la matrice de blanchiment system generator.....	54
<b>Figure. A.</b> Générer un signal mixte avec programme Matlab.....	58

<b>Figure B.</b> schéma bloc Simulink du sous system du centrage.....	62
<b>Figure C.</b> schéma bloc Simulink du compteur.....	62
<b>Figure D.</b> schéma bloc system generator du calcul des valeurs et vecteurs propres ....	63
<b>Figure E.</b> schéma bloc system generator du calcul des éléments de E.....	63
<b>Figure F.</b> schéma bloc system generator du calcul des éléments de D.....	64

<b>Liste des tableaux</b>	
<b>Tableau 3.1</b> programme Matlab de la partie blanchiment.....	<b>Error! Bookmark not defined.</b>
<b>Tableau 3.2</b> programme Matlab de la partie FAST ICA.....	41
<b>Tableau 3.3</b> comparaison des résultats du system generator et Matlab Si .....	56
<b>Tableau. 1</b> blocs Simulink utilisés.....	61
<b>Tableau. 2</b> blocs system generator utilisés .....	62

# Introduction générale

---

Ce mémoire aborde le problème de la séparation des signaux à l'aveugle (Bss) à l'aide de l'analyse rapide des composants indépendants (Fast ica). Dans la séparation aveugle des signaux, les signaux provenant de multiples sources arrivent simultanément sur le réseau de récepteurs, de sorte que chaque sortie du réseau de récepteurs contienne un mélange de signaux source. Les ensembles de sorties du récepteur sont traités pour récupérer les signaux sources ou pour identifier le système de mélange. Le terme aveugle signifie qu'aucune connaissance explicite des signaux sources ou système mélange est disponible. L'approche de l'analyse des composants indépendantes utilise des statistiques d'indépendance des signaux sources pour résoudre les problèmes de séparation aveugle des signaux. Les domaines d'application comprennent la communication, biomédical, la parole et l'image.

Si nous considérons la situation d'assister à une fête, nos oreilles captent de nombreux sons : la voix d'un ami, la voix des autres, la musique de fond, les téléphones qui sonnent, et bien d'autres « figure 1 ». Si on se concentre, on peut entendre ce que dit une personne et on va filtrer n'importe quel autre son. On peut aussi changer son centre d'attention. Par exemple, on peut d'abord faire attention à la parole de votre ami et mettre l'accent sur la musique s'il joue une chanson que vous aimez. La capacité de se concentrer et de reconnaître une source spécifique appelée l'effet cocktail party. Si nous devions enregistrer ces sources en plaçant des micros dans de nombreux endroits à l'intérieur de la pièce, la lecture serait mélangée mélange de sons. On pourrait être en mesure de choisir quelques mots ici et là, mais il n'y a aucun moyen d'entendre les détails de la conversation. S'il y avait autant de microphones dans la pièce que de personnes, il est possible d'extraire et de séparer chaque conversation en utilisant la séparation de sources aveugle [20]. Cela nous permettrait d'entendre tout dans la pièce.

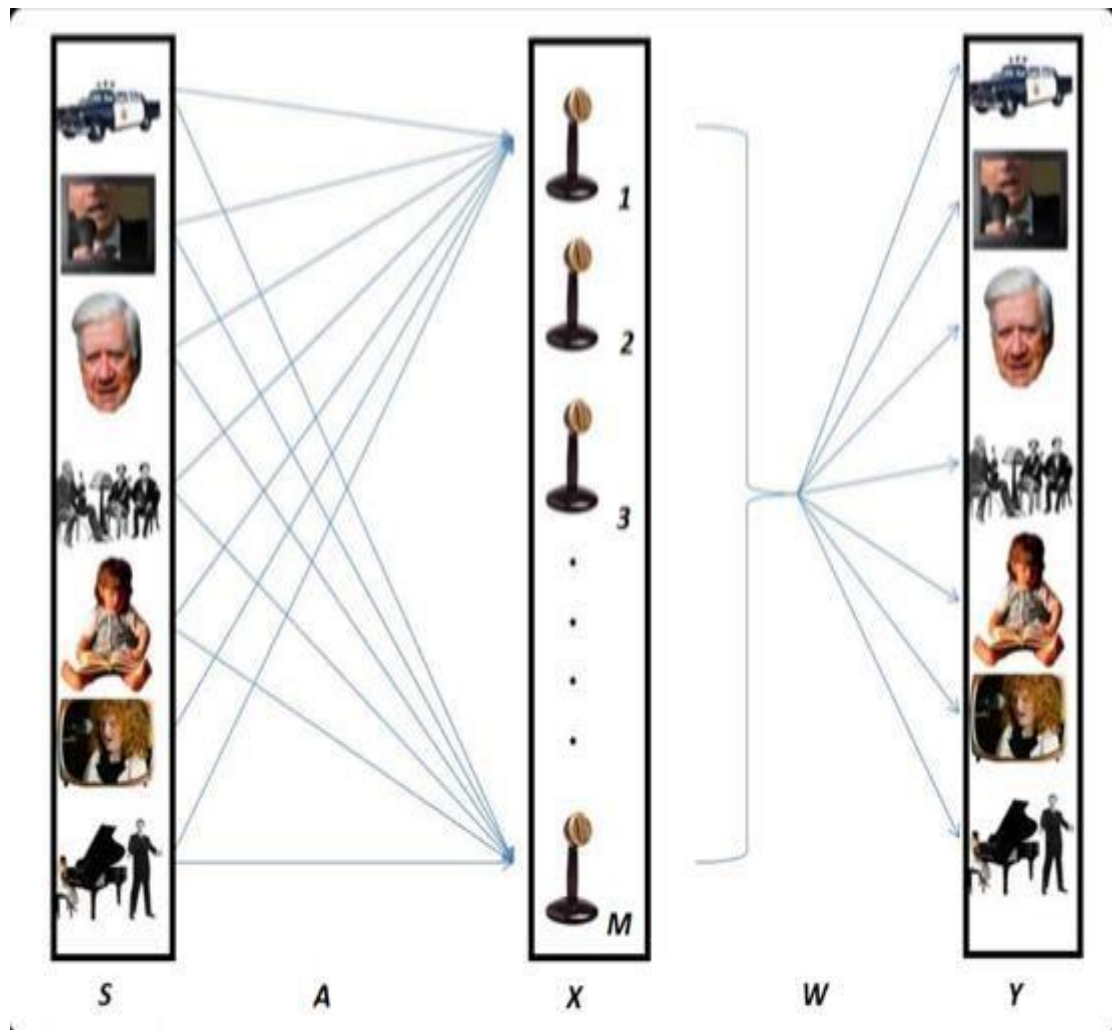


Figure.1. L'exemple de séparation de sources de mélanges instantanés

L'algorithme Fast ICA améliore l'efficacité de l'analyse indépendante des composants. Ça ne peut pas être appliqué aux applications en temps réel telles que l'amélioration du signal vocal et l'EEG / MEG extraction de caractéristiques pour interface informatique cérébrale (BCI). Afin de réaliser le traitement de signal en temps réel, l'algorithme Fast ICA peut être implémenté sur une matrice de portes programmable (FPGA) pour accélérer les calculs impliqués.

# Chapitre 1 **ANALYSE INDEPENDANTE DES COMPOSANTS**

---

## **1.1 Introduction :**

L'analyse indépendante des composants (ICA) est une méthode bien connue pour trouver la structure latente dans les données. L'ICA est une méthode statistique qui exprime un ensemble d'observations multidimensionnelles en tant que combinaison de variables latentes inconnues. Ces variables latentes sous-jacentes sont appelées sources ou composants indépendants et ils sont supposés être statistiquement indépendants l'un de l'autre [1].

$$X = A.S \quad (1.1)$$

Où  $S$  est la matrice source original,  $X$  est la matrice d'observation ( $S$  et  $X$  sont de même taille  $m \times n$ ) et  $A$  est une matrice de mélange  $m \times m$  inconnue.

Si les deux sources originales  $S$  et la façon dont les sources ont été mélangées sont tous inconnus, et seulement les signaux mixtes ou mélanges  $X$  peuvent être mesurés et observés, alors l'estimation de  $A$  et  $S$  est connu sous le nom de problème de séparation de source aveugle (BSS).



**Figure 1.1 séparation de source aveugle.**

Le modèle linéaire (1.1) est identifiable sous les restrictions fondamentales suivantes : les sources sont statiquement indépendantes, et la matrice de mélange  $A$  doit être de plein rang.

L'ICA fait partie d'une famille de techniques, incluant la PCA et la déconvolution aveugle, pour résoudre les problèmes de BSS. ICA est une méthode pour trouver des facteurs sous-jacents dans une donnée multidimensionnelle. Ce chapitre explique également la méthode PCA et ses limites. Enfin, un algorithme spécial ICA appelé FAST ICA est présenté et comparé à la méthode PCA

## 1.2 Indépendance statistique



L'indépendance statique est la clé de l'analyse indépendante des composants. Dans le cas de deux variables aléatoires différentes  $x$  et  $y$ ,  $x$  indépendant de la valeur de  $y$  si la connaissance de la valeur de  $y$  ne donne aucune information sur la valeur de  $x$ . L'Indépendance statistique est définie mathématiquement en terme de densités de probabilité comme les variables aléatoire et sont dites indépendantes seulement si [2] :

$$P_{X,Y}(x, y) = P_X(x) P_Y(y) \quad (1.2)$$

Où  $P_{X,Y}(x, y)$  est la densité commune de  $x$  et  $y$ .  $P_X(x)$  et  $P_Y(y)$  sont les probabilités marginales de la densité de  $X$  et  $Y$  respectivement. La probabilité marginale de  $X$  est donnée par :

$$P_X(x) = \int P_{X,Y}(x, y) dy \quad (1.3)$$

Considérons un vecteur random  $s = [s_1, s_2, \dots, s_N]$  avec des densités multi variés  $P(s)$ . Ce vecteur a des composants statiquement indépendants si la densité est :

$$P(s) = \prod_{i=1}^N P_i(s_i) \quad (1.4)$$

En d'autres termes lorsque deux variables  $s_1$  et  $s_2$  sont indépendantes la densité de  $s_1$  n'est pas affectée par  $s_2$ .

### 1.3 Théorème de la limite central

Le théorème de la limite central est le théorème le plus populaire de la théorie statique et joue un rôle important dans l'ICA.

$$X_k = \sum_{i=1}^k (Z_i) \quad (1.5)$$

Soit une somme partielle de séquence  $\{Z_i\}$  de variables aléatoires indépendantes et identiquement distribuées  $Z_i$ . Puisque la moyenne et la variance de  $X_k$  peuvent croître sans limite comme  $k \rightarrow \infty$ , considérons les vriables standarisées  $y_k$  au lieu de  $X_k$ .

$$y_k = \frac{X_k - m_x}{\sigma_k} \quad (1.6)$$

Où  $m_x$  et  $\sigma_k$  sont la moyenne et la variance de  $X_k$ . La distribution de  $y_k$  converge vers une distribution gaussienne avec une variance moyenne et unitaire nulle lorsque  $k \rightarrow \infty$ .

Ce théorème joue un rôle crucial dans l'ICA et le BSS. Un composant du vecteur de données  $x$  est de la forme

$$x_i = \sum_{j=1}^N a_{ij} S_j \quad (1.7)$$

Où  $a_{ij}$ ,  $j = 1, 2, \dots, N$  sont des coefficients de mélange constants et  $S_j$  sont les  $N$  signaux source inconnus. Le théorème de la limite centrale peut être énoncé comme « la somme de deux variables aléatoires indépendantes identiques distribuées est plus gaussienne que les variables aléatoires originales ».

Cela implique que les variables aléatoires indépendantes sont plus non gaussiennes que leurs mélanges. Par conséquent, la non gaussianité est l'indépendance [3].

## 1.4 Paramètres statistiques généraux : non gaussianité et kurtosis

L'objectif principal de tout modèle statistique est de trouver une représentation appropriée des paramètres d'un système multi variable qui rende la structure essentielle régissant les variables plus visibles. Cela présente généralement des obstacles calculatoires auxquels il faut s'attaquer. Chaque vecteur d'entrée dans  $X$  contient une combinaison linéaire d'échantillons de capteurs observés,  $W$  est la matrice de démixtion [3]. Le but, est de déterminer la matrice de sortie  $Y$ . Cependant, le système contient des sources inconnues sans connaissance préalable du type de bruit et de sa contribution au système. De plus, la difficulté de ce modèle réside dans le fait qu'il nécessite de déterminer les éléments de la matrice utilisant l'algèbre linéaire, qui trouve une solution simple à  $W$  seulement en supposant l'indépendance des signaux.

Les signaux acquis vont subir des transformations afin qu'ils deviennent indépendants.

Cette indépendance implique une recherche de la non gaussianité des signaux (Selon le théorème de la limite centrale, la non gaussianité est une forte mesure d'indépendance).

Les statistiques d'ordre supérieur utilisent Kurtosis ou cumulants de quatrième ordre pour mesurer la non gaussianité. L'aplatissement d'une variable aléatoire de moyenne nulle  $v$  est défini par [4]

$$\text{Kurt}(v) = \{v^4\} - 3(E\{v^2\})^2 \quad (1.8)$$

Où  $\{v^4\}$  = moment d'ordre quatre de  $v$

$\{v^2\}$  = moment d'ordre deux de  $v$

Pour une variable aléatoire gaussienne  $v$ ,  $\{v^4\}$  est égale à  $3(E\{v^2\})^2$ , de sorte que le kurtosis est nul pour une variable aléatoire gaussienne. Si  $v$  est une variable aléatoire non gaussienne, son kurtosis est soit positif soit négatif.

Particulièrement quand la valeur de kurtosis est positive les variables aléatoires sont appelées supergaussiennes ou leptokurtiques et quand elle est négative appelées subgaussiennes ou platykurtiques. Les variables aléatoires supergaussiennes ont une fonction de densité de probabilité de forme pointue au niveau de la moyenne avec des extrémités plus longues et étendues et les variables aléatoires subgaussiennes de forme plus arrondie autour de la moyenne (densité de probabilité plate) avec des extrémités plus courtes et resserrées. De ce fait, la non gaussianité est mesurée par la valeur absolue de kurtosis [5, 6, 7, 8].

## 1.5 Prétraitement des données pour ICA (Analyse en composantes principales PCA)

Il est souvent avantageux de réduire la dimensionnalité des données avant d'effectuer une ICA. Il se pourrait bien qu'il n'y ait que quelques composants latents dans les données observées de haute dimension, et la structure des données peuvent être présentée dans un format compressé. L'estimation de l'ICA dans l'espace original de haute dimension peut conduire à des résultats médiocres. Par exemple, plusieurs des dimensions d'origine peuvent contenir uniquement du bruit. En outre, un apprentissage excessif est susceptible de se produire dans l'ICA si le nombre de paramètres du modèle (c'est-à-dire la taille de matrice mélange) est important par rapport au nombre de point de données observés. Cependant, il faut veiller à ce que seules les dimensions redondantes soient supprimées et que la structure des données ne soit pas aplatie à mesure que les données sont projetées dans un espace de dimension inférieure. Dans cette section, deux méthodes de réduction de la dimensionnalité sont discutées : l'analyse en composantes principales et la projection aléatoire [4].

En plus de la réduction de dimensionnalité, une autre étape de prétraitement souvent utilisée en ICA est de rendre les signaux observés nuls et de les décorréler. La décorrélation supprime les dépendances de second ordre entre les signaux observés. Il est souvent réalisé par analyse en composantes principales

PCA transforme un processus tel que les données sont présentées le long d'un nouvel ensemble de dimensions orthogonales avec une matrice de covariance diagonale [9]. De plus le coefficient PC avec la plus grande variance est le premier composant principal ; le coefficient de PC avec la deuxième plus grande variance est le deuxième le plus important et ainsi de suite.

L'algorithme PCA comprend les étapes suivantes [9] :

Dans l'analyse en composantes principales (PCA), un vecteur observé  $X$  est d'abord centré en supprimant sa moyenne (en pratique, la moyenne est estimée comme la valeur moyenne du vecteur dans un échantillon).

Ensuite, le vecteur est transformé par une transformation linéaire en un nouveau vecteur, éventuellement de dimension inférieure, dont les éléments ne sont pas corrélés entre eux. La transformation linéaire est trouvée en calculant la décomposition de la valeur propre de la matrice de covariance. Pour un vecteur de moyenne nulle  $X$ , avec  $n$  éléments, la matrice de covariance  $C_X$  est donnée par [10] :

$$C_X = \{XX^T\} = EDE^T \quad (1.9)$$

Où  $E = (e_1, e_2, \dots, e_n)$  matrice orthogonale des vecteurs propres de  $C_X$

$D = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_n)$  matrice diagonale de valeur propre de  $C_X$  Le

blanchiment ou la sphérisation peuvent être décrits comme :

$$Z = PX \quad (1.10)$$

Où  $P$  est matrice de blanchiment et  $z$  est la nouvelle matrice blanchie  $P$  est défini comme :

$$P = D^{-1/2} \times E^T \quad (1.11)$$

L'estimation ICA subséquente est faite sur  $z$  au lieu  $x$ . Pour les données blanchies, il suffit de trouver une matrice de démixtion orthogonale si les composants indépendants sont également supposés blanchis.

## 1.6 Application de la Fast ICA pour un composant indépendant

Différentes mesures de la non linéarité sont introduites, c'est-à-dire des fonctions objectives pour l'estimation de l'ICA. En pratique, il faut également un algorithme pour maximiser la fonction des contrastes. Dans cette section, nous introduisons une méthode de maximisation très efficace adaptée à cette tâche. Supposons que nous ayons recueilli un échantillon du vecteur aléatoire de sphère (ou pré-blanchi)  $x$ , qui est dans le cas d'une collection de mélange linéaire de signaux source indépendants.

La méthode de base de l'algorithme Fast ICA est la suivante [4]:

1. Prendre un vecteur initial aléatoire  $W(0)$  et le diviser par sa norme.
2. Mettre  $K=1$ .
3. Soit  $W(k) = E \{(Z^T \times W(k-1))^3\} - 3 \times W(k-1)$  (1.12)
4.  $W(k)$  est divisé par sa norme.
5. Si  $|W^T(k)W(k-1)|$  n'est pas assez proche de 1, incrémenter  $k$  (i.e.  $k=k+1$ ), et retourner à l'étape 2. Sinon, l'algorithme est convergent et sort  $W(k)$ .

Le vecteur final de  $W(k)$

donné par l'algorithme et est égal à l'une des colonnes de la matrice de démixtion (orthogonale)  $B$ . En cas de séparation de source aveugle, cela signifie que  $W(k)$  sépare l'un des signaux sources non gaussiens de telle manière que l'expression  $W^T(k) \cdot x(t)$ ,  $t=1,2,\dots$  est égale à l'un des signaux sources [11, 12, 13, 14].

Pour estimer  $n$  composants indépendants, on exécute ces algorithmes  $n$  fois. Pour s'assurer que chaque fois on estime un composant indépendant différent, on a seulement besoin d'ajouter une projection orthogonalisante simple à l'intérieur de la

boucle. On peut donc estimer les composants indépendants un par un en projetant la solution actuelle  $W(k)$  sur l'espace orthogonal aux colonnes de la matrice démixtion  $B$  précédemment trouvée. Définir la matrice  $B$  comme la matrice dont les colonnes sont les colonnes précédemment trouvées de  $B$ . Puis ajouter l'opération de projection au début de l'étape 3. La figure (1.2) donne un organigramme de cet algorithme.

$$W = W - \bar{B}\bar{B}^T \times W \quad (1.13)$$

On devise  $W$  par sa norme. L'organigramme de la Fast ica est donné par la Figure. 1.2

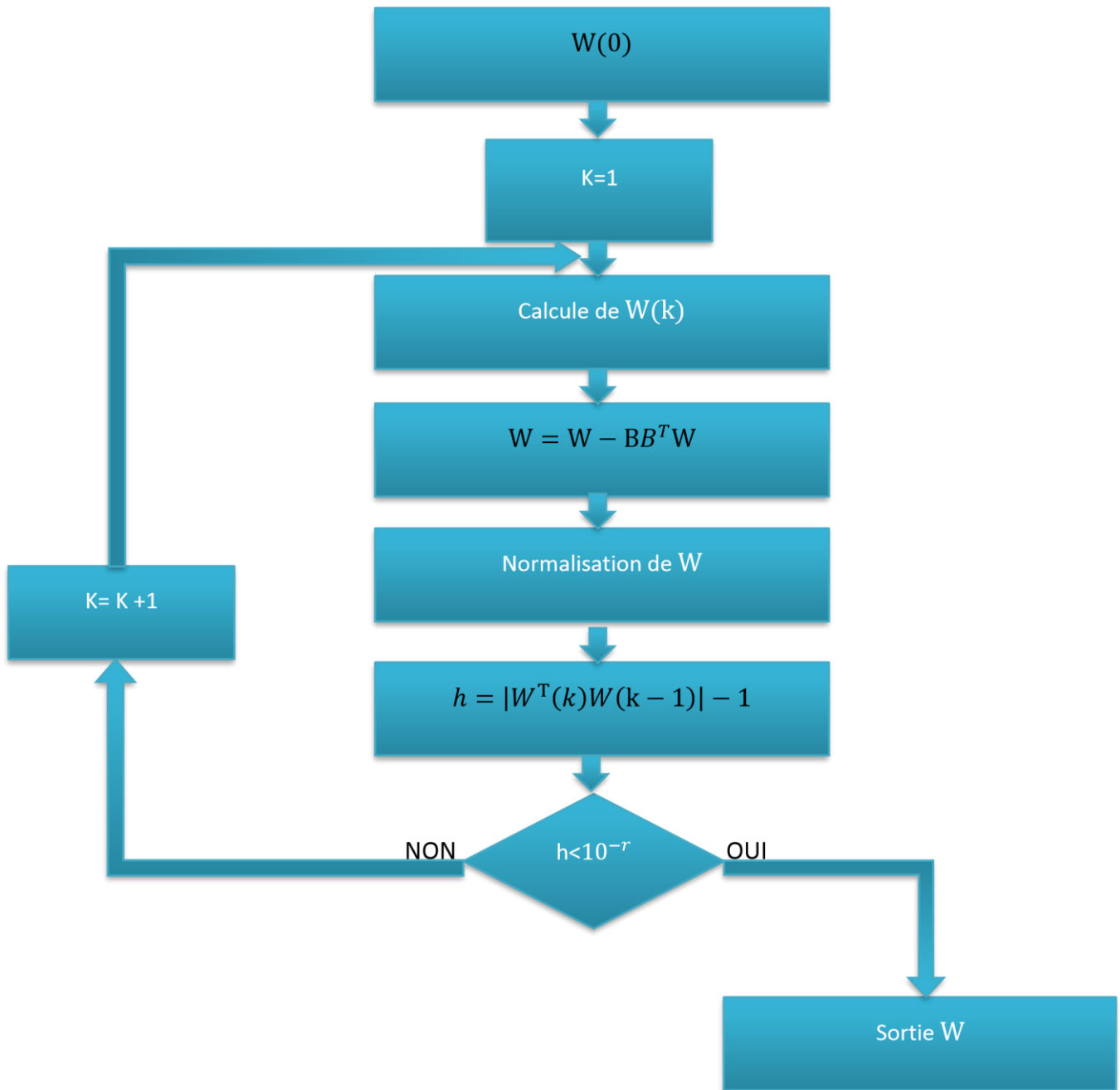


Figure 1.2 Organigramme de l'algorithme FAST ICA.

## 1.7 Domaines d'applications de l'ICA

L'ICA étant une technique de traitement de signal statique aveugle trouve application dans de nombreux domaines d'application tels que la séparation aveugle de voix mixte ou d'image, l'analyse de plusieurs types de données, l'extraction de caractéristiques, la reconnaissance de parole et d'image, identification du système, traitement du signal biomédical et plusieurs autres.

Signaux biomédicaux tels que l'électroencéphalogramme (EEG), la magnétoencéphalographie (MEG), et l'électrocardiogramme (ECG) sont généralement mesurés à partir de capteurs, les signaux mesurés sont pollués par des signaux de bruit inconnus, tels que les mouvements des yeux, le bruit musculaire et le bruit de puissance des instruments. Ce problème peut être résolu par l'algorithme d'analyse de composants indépendants (ICA).

Un autre domaine d'application d'un grand potentiel est celui des télécommunications. Un exemple d'application de communication realworld où les techniques de séparation aveugle sont utiles est la séparation du propre signal de l'utilisateur des signaux interférant d'autres utilisateurs dans les communications mobiles CDMA (Code- Division Multiple Access). Ce problème est semi-aveugle en ce sens que certaines informations préalables supplémentaires sont disponibles sur le modèle de données à estimer est souvent si élevé que des techniques appropriées de séparation des sources aveugle prenant en compte les connaissances antérieures disponibles fournissent une nette amélioration des performances par rapport aux techniques d'estimation plus traditionnelles.

ICA est utilisé avec succès pour la reconnaissance faciale. Le but de la reconnaissance faciale est de former un système capable de reconnaître et classer les visages familiers. Les méthodes traditionnelles de reconnaissance faciale ont employé des méthodes analogiques à l'ACP. Les lignes des images des visages constituent la matrice de données  $x$ . En utilisant ICA, une transformation  $W$  est apprise pour que  $u$  ( $u = Wx$ ) représente les images de face indépendantes.

Un réseau de capteurs est un domaine de recherche très récent, largement applicable et stimulant. Les données multi-capteurs présentent souvent des informations complémentaires sur la région étudiée et la fusion de données fournit une méthode efficace pour permettre la comparaison, l'interprétation et l'analyse de ces données. Fusion d'images et de vidéos dans une sous-zone du sujet plus général de la fusion de données traitant des données d'image et de vidéo.



ICA est également utilisé pour une reconnaissance vocale automatique robuste. Les applications de l'ICA comprennent également l'extraction de caractéristiques dans des images et la recherche de facteurs cachés dans les données financières.

## **1.8 Conclusion**

Dans ce chapitre, les modèles PCA et ICA ont été expliqués. L'analyse indépendante des composants (ICA) est une méthode permettant de trouver des facteurs sous-jacents à partir de données statistiques multivariées (multidimensionnelles). Ce qui distingue l'ICA des autres méthodes, c'est qu'elle recherche des composants à la fois statistiquement indépendants et non gaussiens dans lesquels la PCA décèle simplement les signaux en fonction de leurs variances. Ce chapitre a également expliqué l'algorithme FAST ICA et comment trouver toutes les composantes de  $W$ .

# Chapitre 2 GENERALITES SUR L'ELECTROCARDIOGRAPHIE ET PLATFORME D'IMPLEMENTATION

---

## 2.1 Introduction

Ce chapitre va présenter dans un premier temps l'électrocardiographie du point de vue appareillages techniques de captures de signaux etc., puis les plateformes sollicitées pour l'implémentation de la fast ica.

## 2.2 L'électrocardiographie

### **2.2.1 Introduction**

La maladie cardiaque est la cause principale du début d'invalidité et de décès prématurés des individus. Son incidence augmente avec l'âge. L'électrocardiographie (ECG) est l'outil de diagnostic utilisé pour évaluer la probabilité d'anomalies cardiaque. La transmission en temps réel de cette information est une grande avancée dans le domaine médicale.

Depuis le premier appareil du début du siècle, l'électrocardiographe a évolué en adaptant au fur et à mesure les technologies d'actualité. Ainsi, le signal analogique entraînant une aiguille est devenu numérique, exploité par logiciel.

Ce chapitre présente le principe de l'électrocardiographie et le type de signale capter, la différence entre un ECG normal est un ECG pathologique.

### **2.2.2 L'électrocardiographie**

L'électrocardiographie est une technique relativement peu couteuse permettant, à l'aide d'un simple examen et sans danger, de surveiller le bon fonctionnement de l'appareil cardiovasculaire.

L'électrocardiographie est l'étude des variations de l'enregistrement de l'activité électrique des cellules cardiaque, dont dépend la contraction du cœur. Il consiste a recueillir au niveau de la peau le champ électrique crée par ces courant d'activités de la fibre musculaire cardiaque, a l'amplifier puis l'enregistrer. L'électrocardiographie s'est révélé comme étant une technique primordiale pour la surveillance ou dans le diagnostic. L'abréviation usuelle utilisée pour parler de L'électrocardiographie est l'ECG, en anglais comme en français.

### **2.2.3 Principe de fonctionnement**

L'électrocardiographie (ECG) consiste à recueillir les variations du potentiel électrique, à les amplifier puis les enregistrer. Les signaux captés étant particulièrement faibles, des amplificateurs de hautes performance (gain, linéarité, minimum de bruit de fond) sont souvent nécessaires. Sauf pour des études particulières portant sur les aspects énergétiques de l'électrogénèse, on ne s'intéresse guère à la puissance des générateurs bioélectriques, ni aux courant qu'ils débitent.

## 2.2.4 Appareillage

Il existe deux types de L'électrocardiographie :

- Les anciens L'électrocardiographes dit les L'électrocardiographie analogiques, ces modèles utilisent plusieurs étages des circuits basés sur les amplificateurs d'instrumentations (figure 2.1).



**Figure. 2.1** électrographe analogique.

- Les L'électrocardiographes modernes, il se présentent sous forme d'appareils compacts intégrant de nombreuses fonctions. En effet, ces appareils comportaient un écran à cristaux liquides, une dizaine de dérivations, une imprimante, un logiciel d'exploitation des résultats, et une possibilité de stocker les enregistrements sur disque dur, ou de les transmettre sur une ligne téléphonique ou wifi ((Figure 2.2).



Figure. 2.2 électrographe numérique.

### 2.2.5 Chaîne d'acquisition des données de l'ECG

Le traitement du signal cardiaque dépend de la chaîne d'acquisition analogique, elle représente un rapport signal sur bruit avec reproduction du signal correct et fidèle, car elle permet un traitement numérique basé sur les algorithmes de détections d'anomalies et des erreurs. On distingue deux chaînes d'acquisitions des données, une chaîne analogique et l'autre est numérique.

#### *a Chaîne analogique*

La qualité de la mesure et du traitement du signal cardiaque dépend en grande partie de la chaîne d'acquisition analogique, celle-ci doit présenter un rapport signal sur bruit correct une reproduction fidèle du signal cardiaque, de manière à permettre ultérieurement un traitement numérique basé sur des algorithmes de détection d'anomalies

#### *b Chaîne numérique*

La chaîne de mesure numérique est unie d'une logique de commande à base de microprocesseur

a. Le conditionnement du signal Il

comprend les étapes suivantes :

- Le préamplificateur : Le signal électrique délivré par les électrodes est très faible, il est en général perturbé par des bruits, un préamplificateur se révèle donc nécessaire.
- Le filtre : Le signal peut être masqué par des bruits, la fonction principale d'un filtre consiste à éliminer certain nombre d'harmoniques.
- Amplificateur : apporte un gain en puissance suffisant pour que le signal puisse être correctement enregistré et tracé

b. Echantillonnage

c. Le microprocesseur

d. Le capteur

Le capteur joue un rôle très important dans ces deux chaînes, car il transforme le phénomène physique à un phénomène électrique. Il existe un lien entre le tissu biologique et l'appareil.

### **2.2.6 Structure de la peau**

La peau est un tissu vivant qui respire, formé de cellules dont la durée de vie est limitée mais qui se reproduisent régulièrement. Si on en fait une coupe on peut distinguer deux zones, le derme et l'épiderme.

Le derme conduit bien les ondes électriques émises par le cœur ou les différents muscles tandis que l'épiderme formé de cellules mortes plus ou moins desséchées est un mauvais conducteur.

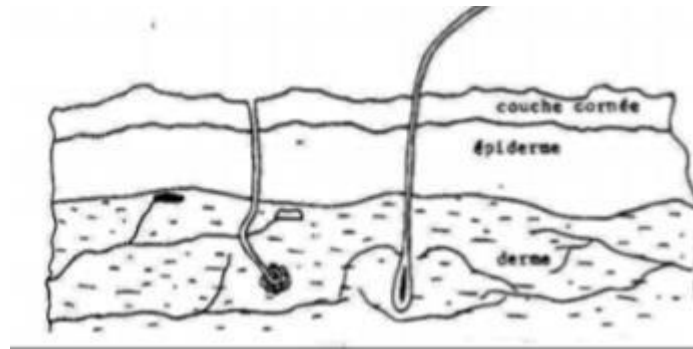


Figure. 2.3 Structure de la peau.

Dans ces conditions, pour capter correctement les signaux électriques venant de l'intérieur du corps, il faut soit utiliser des électrodes aiguilles plantées profondément dans le derme ce qui n'est pas sans inconvénients, soit utiliser des plaques que l'on met en contact avec l'épiderme. C'est cette dernière formule qui est utilisée dans la majorité des cas [15]

### 2.2.7 Les électrodes

Il s'agit de capter par voie externe les ondes électriques émises par le cœur. L'élément sensible sera donc une plaque faite d'un matériau conducteur que l'on mettra en contact avec la peau. Plus la surface de contact sera grande, plus l'impédance de contact diminuera. Avant de l'appliquer, il sera nécessaire de nettoyer la peau.

Deux types différents d'électrodes peuvent être utilisés pour la capture du signal cardiaque [22].

- a. Les électrodes bipotentiellles
- b. Les électrodes autocollantes

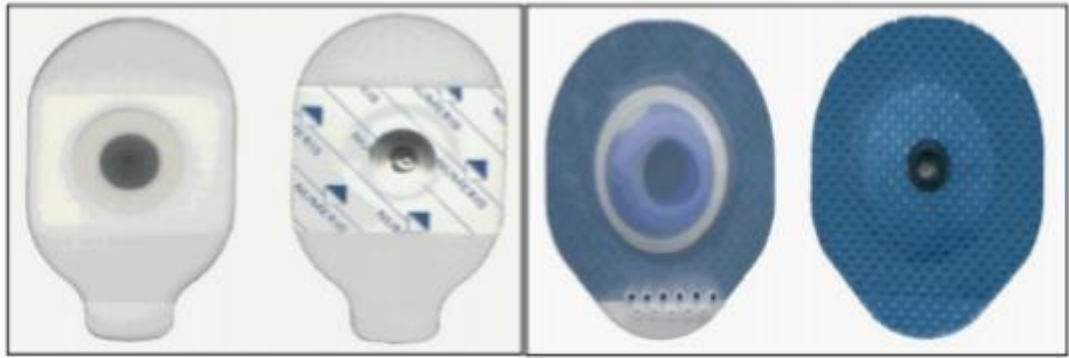


Figure 2.4 Les électrodes autocollantes

### 2.2.8 Types de bruits dans le signal ECG

Sur tout enregistrement électrocardiographique, il peut apparaître des événements indésirables pouvant brouiller le tracé et parfois, induire en erreur le diagnostic final. Les effets indésirables peuvent avoir plusieurs sources : techniques ou pathologiques. En partant du principe que les bruits fréquents en électrocardiographie sont des bruits additifs, les caractéristiques de ces bruits sont l'amplitude, la périodicité et la bande spectrale. Ces bruits peuvent être classés selon leurs origines en grandes catégories, les bruits d'origine technique et les bruits d'origine physique [23].

#### a. Bruits techniques

Les bruits d'origine technique sont les bruits qui sont causés par le matériel utilisé lors de l'enregistrement et dont les plus courants sont :

- Le bruit du réseau.
- Interférence électromagnétique.
- Mouvements des câbles électriques.
- La saturation des instruments de mesure.
- Mauvaise qualité du câblage.



## b. Bruits physiques

Les bruits d'origine physique sont des artefacts engendrés par, soit des activités électriques du corps humain telles que les contractions musculaires, soit par les mouvements lors de la respiration.

### **2.2.9 ECG fœtal**

La fréquence cardiaque fœtale (fHR) et l'analyse morphologique de l'électrocardiogramme fœtal (fECG) sont deux des plus outils les plus importants utilisés pour examiner l'état de santé du fœtus pendant la grossesse.

Une méthode alternative pour obtenir la morphologie fHR et fECG instantanée est l'enregistrement abdominal de la fECG qui considère un réseau d'électrodes placés sur l'abdomen maternel. La limitation de cette technique est très faible rapport signal sur bruit (SNR) de la fECG enregistrée disponible. Ceci est principalement dû au fait que le signal fECG est généré par une petite source (cœur fœtus). En outre, il doit se propager à travers différents milieux d'atténuation pour atteindre la surface du ventre maternel. Ainsi, l'amplitude des signaux fECG contenus dans les signaux abdominaux (ADS) devient plus petite autour de la 28<sup>e</sup> à la 32<sup>e</sup> semaine de l'âge gestationnel en raison de l'apparition de la couche isolante appelée vernix caseosa. Pour résoudre ce problème on va utiliser la technique de séparation FAST ICA [24].

## **2.3 PLATE FORME D'IMPLEMENTATION**

### **2.3.1 Introduction**

Notre projet est l'implémentation de la méthode FAST ICA pour faire la séparation d'un signal ECG (maman fœtus). Nous avons d'abord fait une simulation sur Matlab Simulink pour aller à FPGA. Nous allons dans ce chapitre présenter Matlab Simulink, les circuits programmables FPGA, ensuite nous présentons les l'outil de conception Xilinx system generator.

### 2.3.2 Matlab Simulink

Matlab est un logiciel de calcul matriciel à syntaxe simple, avec ses fonctions spécialisées. Il peut être aussi considéré comme un langage de programmation adapté pour les problèmes scientifiques [16].

Simulink est l'extension graphique de Matlab permettant de travailler avec des programmes en blocs.

### 2.3.3 Field programmable gate array (FPGA)

Un FPGA est un circuit intégré (CI), qui est défini comme une matrice de blocs logiques configurables (Configurable Logic Blocks, CLB) reliés entre eux par des interconnexions entièrement reconfigurables. Ces blocs et interconnexions peuvent être programmés par l'utilisateur via des cellules mémoire statiques [17]. L'architecture d'un FPGA se décompose en deux types de ressources : les ressources de traitement (incluant les mémoires, la logique, les registres) regroupées en blocs logiques de différents types, et les ressources d'interconnexions programmables qui relient les blocs logiques entre eux.

La programmation d'un circuit reconfigurable consiste donc à spécifier la fonctionnalité de chaque bloc logique et à organiser le réseau d'interconnexion afin de réaliser la fonction demandée [18].

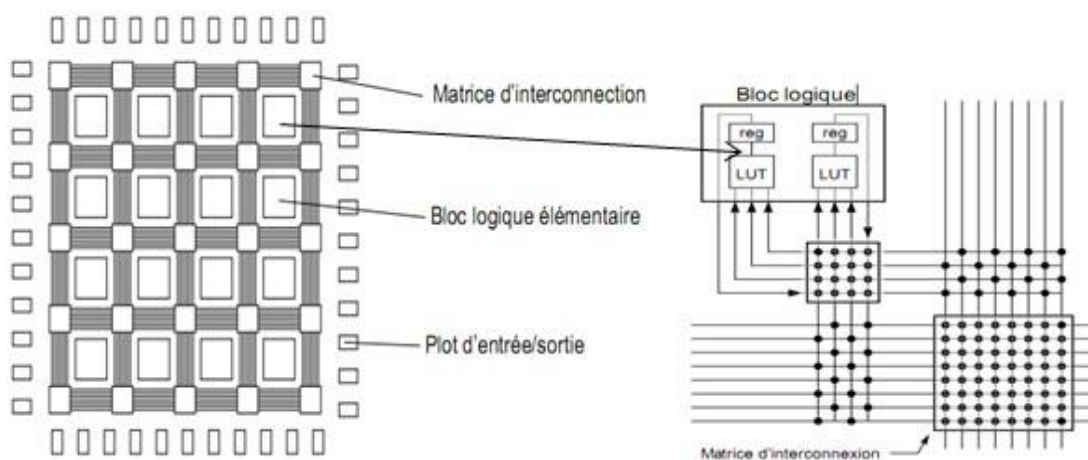


Figure 2.5 Schéma d'une carte FPGA.

Les FPGA sont des circuits intégrés programmables disposant de quelques centaines de milliers à quelques millions de portes logiques configurables. Il est important de noter que dans le cas d'un FPGA, on parle de configuration plus que de programmation. En effet, les données que l'utilisateur charge dans le FPGA affectent directement l'architecture matérielle de ce dernier ; il ne s'agit pas d'instructions purement logicielles exécutables par un processeur mais d'un circuit réalisé à partir du circuit logique disponibles dans le circuit FPGA [17].

Les FPGA sont donc des circuits logiques qui se configurent. La mise en œuvre de ceux-ci est radicalement différente de la programmation à laquelle nous pouvons d'ordinaire être habituée : celle des microprocesseurs ( $\mu P$ ) ou microcontrôleurs ( $\mu C$ ). En effet, ces derniers fonctionnent de façon séquentielle : tâche 1 puis tâche 2 (tâche 1 et tâche 2 indépendantes) Cela vient du fait même de leur constitution : lecture du registre d'instruction, traitement de l'instruction etc. [17].

Les FPGA, quant à eux, jouissent d'un parallélisme total si bien que dans l'exemple précédent la tâche 1 et la tâche 2 s'effectuent simultanément. Le parallélisme leur confère une grande rapidité en comparaison des  $\mu P$  et  $\mu C$ . Toutefois, cela a pour effet de considérablement modifier la façon dont on « programme » ces composants (le séquençage par exemple). Il existe deux principaux langages pour configurer les FPGA ce sont des langages dits de description matériel, Hardware Description Language (HDL) dont les deux largement répandus sont le Verilog et le VHDL [17].

Pour résumer, les FPGA sont des circuits numériques dotés d'une grande multitude de portes logiques programmables par l'utilisateur permettant la réalisation de circuits des fonctions de traitement dédiés à un besoin. Les FPGA peuvent servir à plusieurs applications ce qui représente un sérieux avantage.

### 2.3.4 Constitution interne

Nous avons évoqué précédemment qu'un FPGA est constitué d'une matrice de CLB. Ces blocs sont eux-mêmes composés de quatre tranches (Slices) qui sont-elles même formées par deux cellules logiques (Logic Cell, LC). [17] La Figure 3-2 présente la structure d'un CLB.

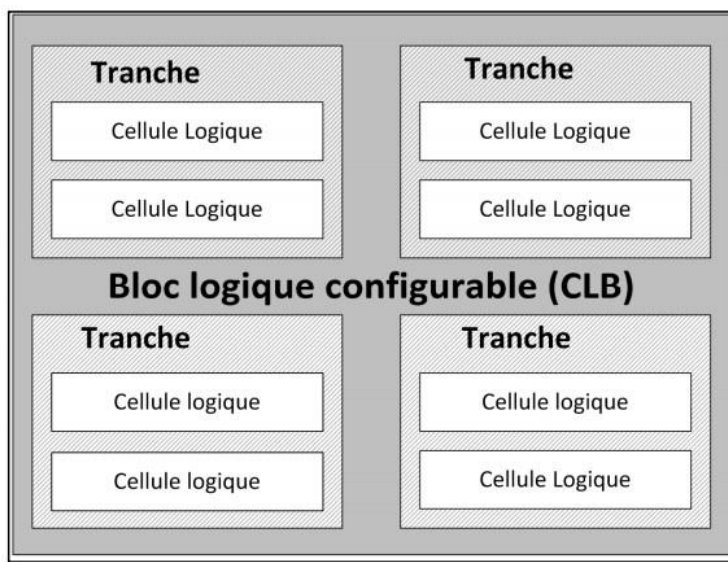


Figure 2.6 bloc logique configurable (CLB)

### 2.3.5 Le circuit XC3SD1800A-4FG676C

Le dispositif Xilinx XC3SD1800A-4FG676C conçu dans Spartan-3A 1800 DSP starter platform fournit quatre bancs d'E / S : deux sont à tension fixe et deux sont des tensions d'E / S sélectionnables. Dans certains cas, la conversion de tension des signaux d'E/S peut être nécessaire pour répondre aux exigences des périphériques connectés à une banque d'E / S particulière ou des modules EXP branchés sur les connecteurs d'extension EXP [21].



**Figure 2.7 un circuit FPGA**

### **2.3.6 SYSTEM GENERATOR**

Il s'agit d'une toolbox développée par Xilinx pour être intégrée dans l'environnement Matlab-Simulink et qui laisse l'utilisateur créer des systèmes hautement parallèles pour FPGA. Les modèles créés sont affichés sous forme de blocs, et peuvent être raccordés aux autres blocs et autres toolboxes de Matlab-Simulink. Une fois le système complété, le code VHDL généré par l'outil Sysgen reproduit exactement le comportement observé dans Matlab. Le passage vers la plateforme matérielle pour des tests sur le terrain est rapide [19].

Xilinx System Generator for DSP est un outil de conception pour Simulink qui permet de concevoir, simuler et développer des systèmes DSP. Cet outil de conception est constitué des blocs de Xilinx, en tant que bibliothèque Simulink qui peut être mappé directement dans le matériel FPGA. En utilisant Xilinx System Generator pour DSP, non seulement les algorithmes DSP peuvent être conçus et simulés, mais aussi un code de (HDL) peut être généré en utilisant le codeur HDL pour les FPGA de Xilinx [19].

Par conséquent, Xilinx System Generator pour DSP est considéré comme un outil de haut niveau pour la conception des systèmes DSP en fournissant la modélisation du système et la génération automatique de code HDL de MATLAB ou Simulink.

Utilisation de bibliothèques de blocs Xilinx dans un environnement de modélisation basé sur un modèle Simulink ne nécessitent une expérience préalable de la conception RTL et FPGA de Xilinx, l'itinéraire et la synthèse sont automatiquement exécutées afin de générer un fichier bit FPGA. Xilinx System Generator fournit également la fonctionnalité d'effectuer une Co-simulation matérielle lorsqu'une conception basée sur un modèle fonctionne sur Simulink et FPGA.

### 2.3.7 Installation

Pour installer Sysgen, il faut installer Xilinx ISE, un logiciel de programmation de langage VHDL

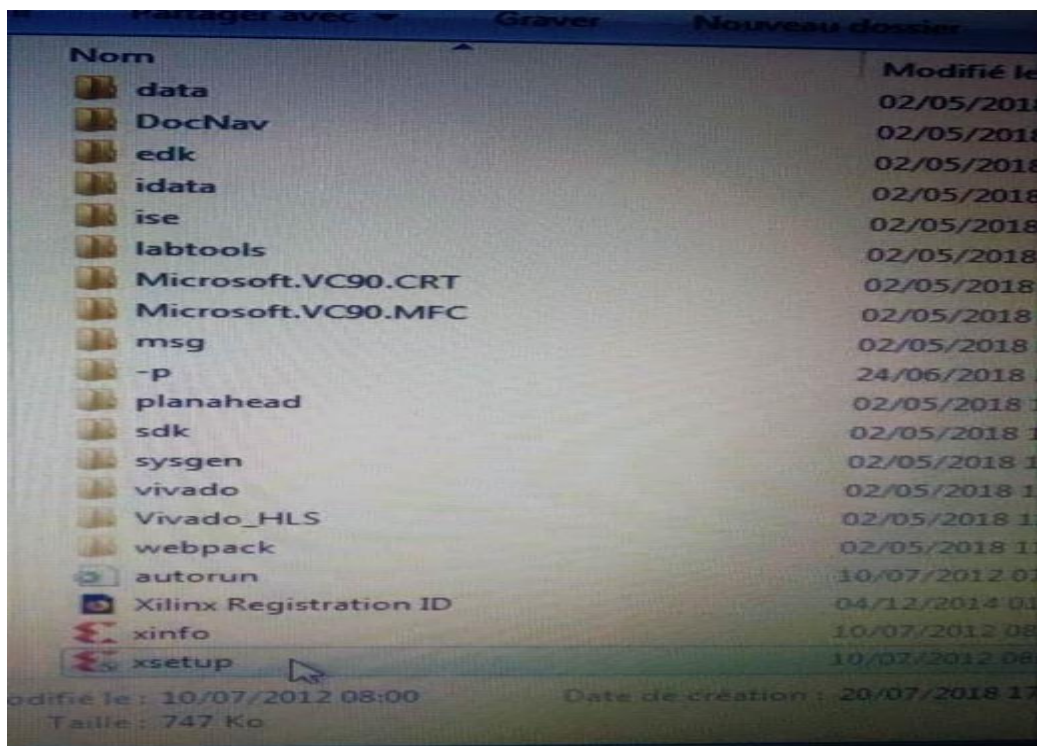


Figure 2.8 Lancement de l'installation

Au cours de l'installation de Xilinx ISE le logiciel demande de configurer Sysgen avec la version Matlab correspondant (notre cas Xilinx ISE 14.2 et MATLAB 2012-a).

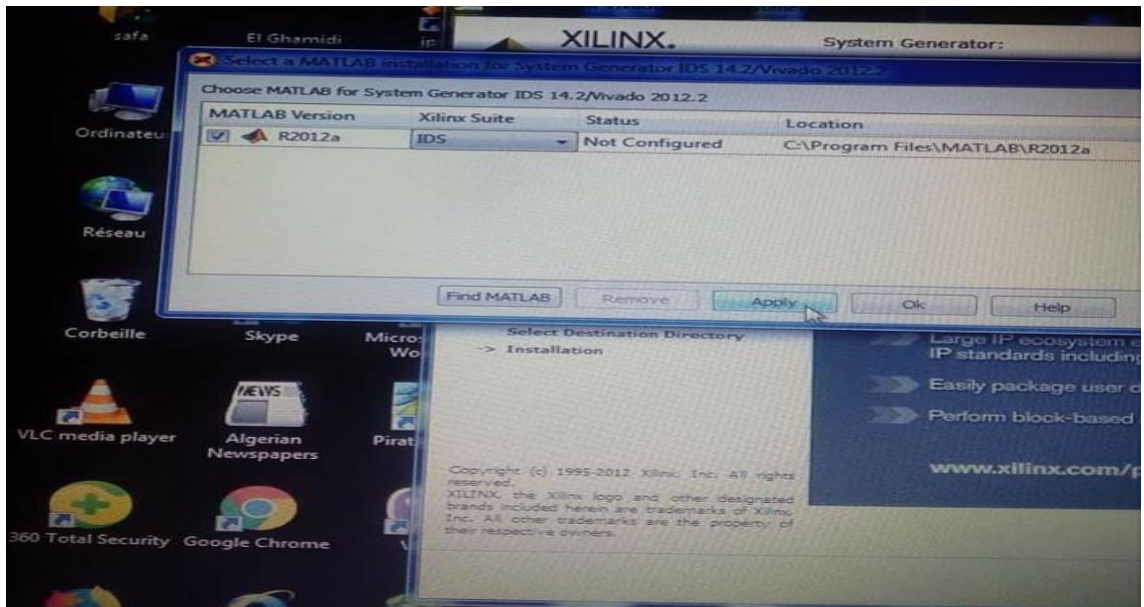


Figure 2.9 Configuration de Matlab avec system generator

A la fin le logiciel demande de lui insérer la licence.

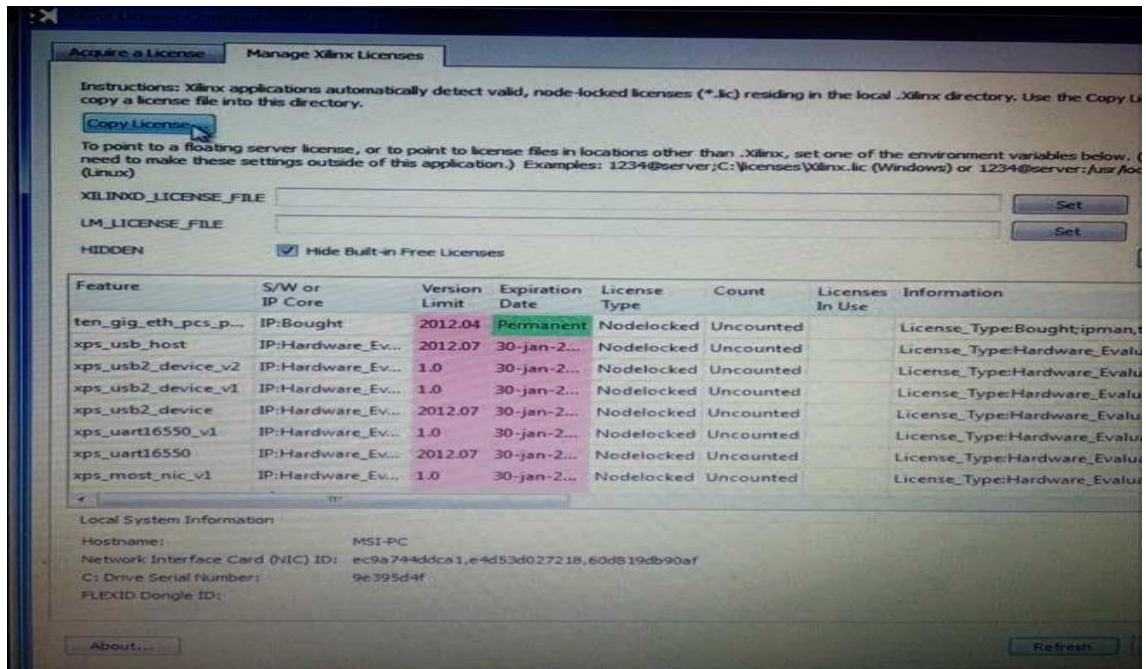
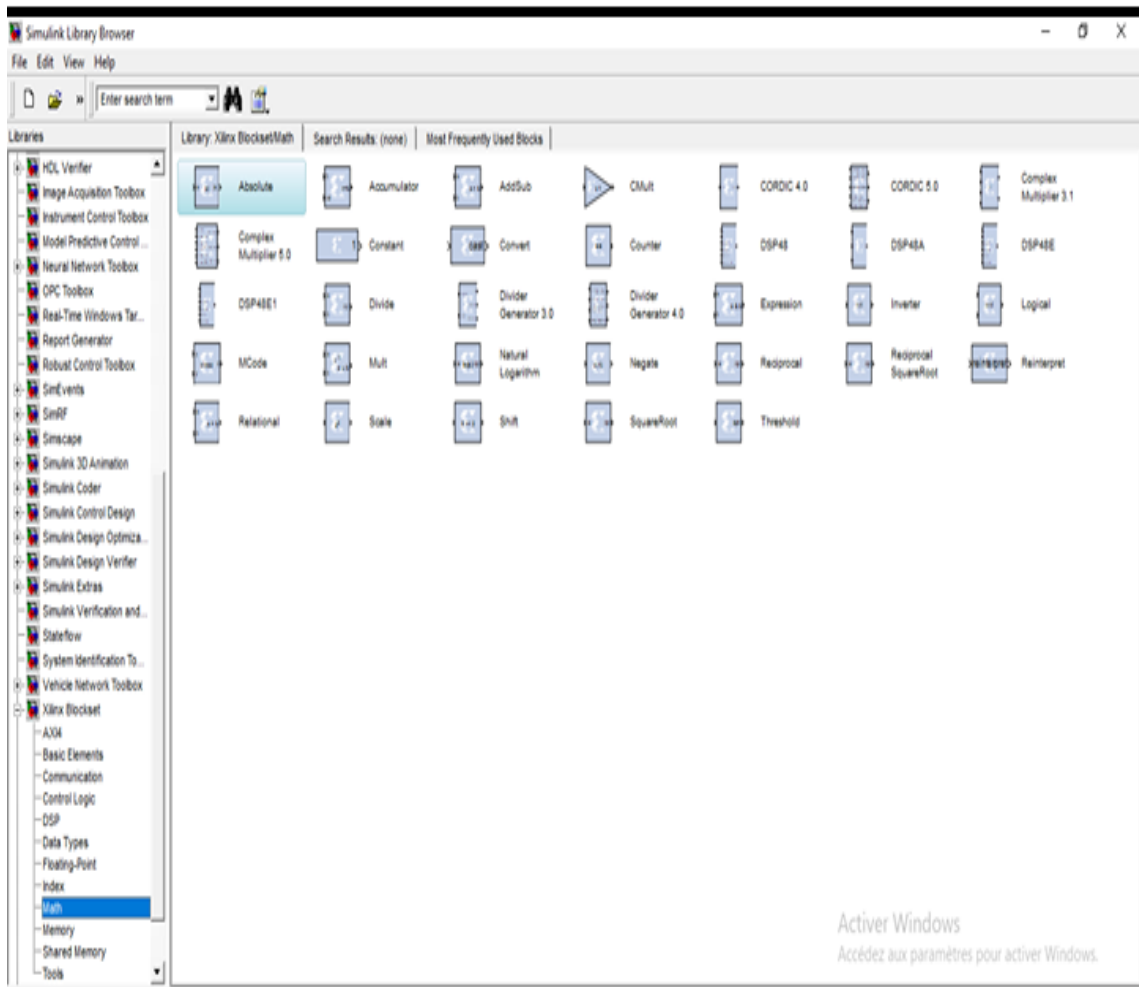


Figure 2.10 Insertion de licence

A la fin de l'installation on vérifie si les blocs de system generator sont apparus dans la fenêtre Simulink





**Figure 2.11 Bibliothèque de system generator**

Pour travailler sur system generator on clique sur Démarrer => system generator => exécuter en tant qu'administrateur.



## 2.4 Conclusion

Nous avons d'abord donné une présentation de la technique de l'électrocardiographie, son principe de fonctionnement, les types des appareillage utilisées et ses composant essentiel et nous avons présenté le problème d'observation de l'ECG foetal. Ensuite Nous avons donné une description globale des circuits FPGA. Puis nous avons vu l'outil de conception de Xilinx system generator for DSP, et les étapes d'installation. Dans le chapitre suivant on va voir les différentes implémentations et les blocs utilisés dans l'implémentation de la méthode FAST ICA.

# Chapitre 3    **IMPLEMENTATION DE L'ALGORITHME FAST ICA RESULTATS ET DISCUSSIONS**

---

## **3.1 Introduction**

Ce projet consiste en l'implémentation VHDL de la méthode de séparation aveugle des signaux sources quelconques BSS puis de signaux mère-foetus, pour ce faire nous avons dans une première phase implémenté la méthode sous Matlab avec une approche hard, en d'autres termes nous n'avons effectuer aucun appel de fonction et donc nous avons utilisé que les opérations de base. La deuxième consiste à refaire le même travail sous Simulink. Quant à la troisième nous avons entamé l'implémentation de la BSS sous xilinx system generator pour une implémentation VHDL.

## **3.2 Implémentation du centrage**

Le processus de centrage (figure3.1) consiste à soustraire les moyens de signaux mixtes de  $x_1$  et  $x_2$  respectivement. D'abord, les éléments de signaux mixtes sont accumulés un par un. Après avoir obtenu la somme de  $x$ ,  $\mu$  est obtenu en divisant la somme par la longueur de l'échantillon.

L'opération de multiplication est utilisée à la place de la division (multiplier par  $1 /$  longueur de l'échantillon).

Deuxièmement, la moyenne est soustraite des données de signaux mixtes pour obtenir le centrage. L'opération est formulée comme suit :

$$X(i) = X(i) - \left[ \sum_{i=1}^{samplelength} x(i) \right] \times (1/samplelength) \quad (3.1)$$

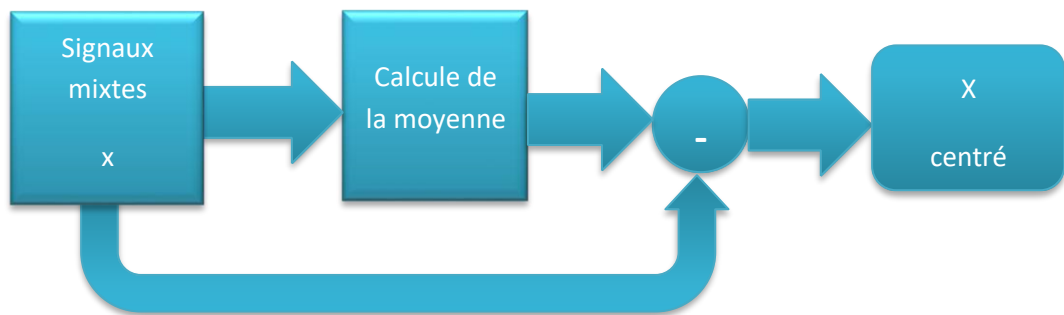


Figure 3.1 schéma fonctionnel de la mise en œuvre du centrage

### 3.3 Implémentation du blanchiment

La première étape est de trouver la matrice de blanchiment  $W$ . elle est donnée par :

$$W = D^{-1/2} \times E^T \quad (3.2)$$

Où  $D = \text{diag}(\lambda_1, \lambda_2)$  = matrice diagonale des valeurs propres de la matrice de covariance  $C_X$ .

$E = (e_1, e_2)$  = matrice orthogonale des vecteurs propres de  $C_X$ .

$C_X = E \{XX^T\}$  matrice  $2 \times 2$ .

$$X = \begin{pmatrix} X_1 \\ X_2 \end{pmatrix} = \begin{pmatrix} X_1(1), X_1(2), \dots, X_1(samplelength) \\ X_2(1), X_2(2), \dots, X_2(samplelength) \end{pmatrix}$$

$$X_2 \quad X_2(1), X_2(2), \dots \dots \dots, X_2(\text{samplelength})$$

Pour calculer le  $C_x$  on doit calculer trois multiplications :  $X_1 \times X_1$ ,  $X_2 \times X_2$  et  $X_2 \times X_1$ .

$$\text{car } X_1 \times X_2 = X_2 \times X_1$$

$C_x$  peut donc être dévisé en multipliant la somme par 1/ la longueur de l'échantillon. Cette opération peut être formulée comme

$$C_x = \begin{pmatrix} C_{x00} & C_{x01} \\ C_{x10} & C_{x11} \end{pmatrix}$$

$$C_{x00} = [\sum_{i=1}^{\text{samplelength}} X_1(i) \times X_1(i)] \times (1/\text{samplelength}) \quad (3.3)$$

$$C_{x10} = C_{x01} = [\sum_{i=1}^{\text{samplelength}} X_1(i) \times X_2(i)] \times (1/\text{samplelength}) \quad (3.4)$$

$$C_{x11} = [\sum_{i=1}^{\text{samplelength}} X_2(i) \times X_2(i)] \times (1/\text{samplelength}) \quad (3.5)$$

Une fois la matrice de covariance calculée, l'étape suivante consiste à déterminer la matrice orthogonale des vecteurs propres de  $C_x$  (E) et la matrice diagonale des valeurs propres de  $C_x$  (D).

$$\sigma = \frac{C_{x11} - C_{x00}}{2 \times C_{x10}} \quad (3.6)$$

$$T = \frac{\text{sign}(\sigma)}{\text{abs}(\sigma) + \sqrt{1 + (\sigma \times \sigma)}} \quad (3.7)$$

$$C = \frac{1}{\sqrt{1 + (\sigma \times \sigma)}} \quad (3.8)$$

$$S = T \times C \quad (3.9)$$

$$E = \begin{bmatrix} C & S \\ -S & C \end{bmatrix} \quad (3.10)$$

$$D = E^T \times [C_x \times E] \quad (3.11)$$

La matrice de blanchiment  $P$  est obtenu en multipliant  $D^{-1/2}$  par  $E^T$

$$P = D^{-1/2} \times E^T \quad (3.12)$$

Les données blanchies  $Z$  sont obtenues en multipliant  $P$  par  $X$

$$Z = PX \quad (3.13)$$

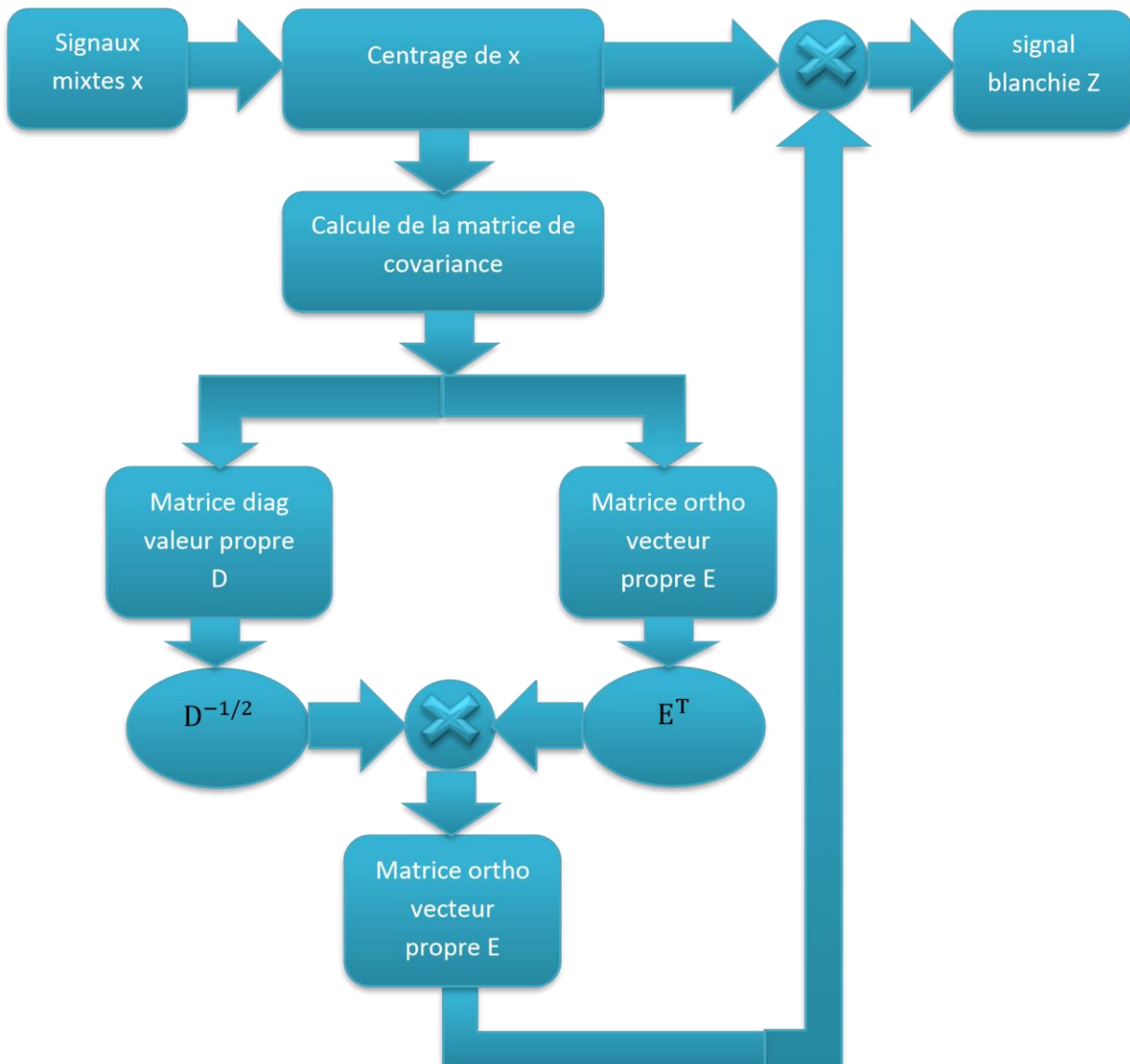


Figure 3.2 schéma fonctionnel de la mise en œuvre du blanchiment

### 3.4 Calcule de la norme

On initialise un vecteur aléatoire  $W = \begin{bmatrix} W_1 \\ W_2 \end{bmatrix}$  sa norme est déterminé comme suit  $W_2$

$$norm\_W = \sqrt{(W_1 \times W_1) + (W_2 \times W_2)} \quad (3.14)$$

On dérive  $W_1$  et  $W_2$  par leur norme

$$W = \begin{bmatrix} W_1/norm\_W \\ W_2/norm\_W \end{bmatrix} \quad (3.15)$$

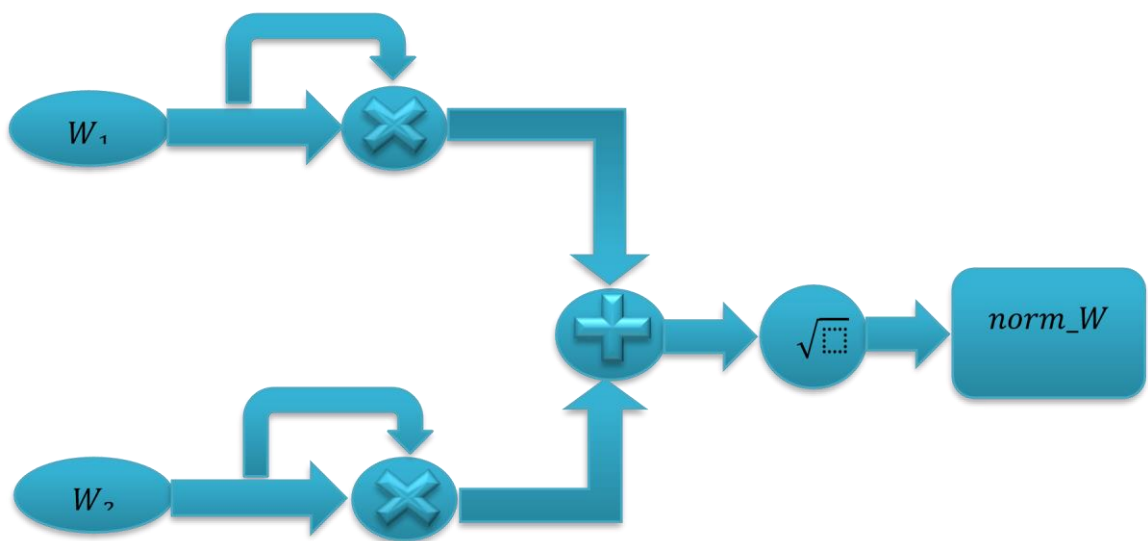


Figure 3.3 schéma fonctionnel du calcul de la norme

### 3.5 Implémentation de la FAST ICA

L'équation pour le calcul du vecteur de séparation  $W$  (figure 3.4) est exprimée comme :

$$W(k) = E \{ (Z^T \times W(k-1))^3 \} - 3 \times W(k-1) \quad (3.16)$$

On implémente d'abord  $pre\_W(k)$  où  $pre\_W(k)$  est le résultat du calcul

$$pre\_W(k) = \sum_{i=1}^{samplelength} [Z(i)(Z(i) \times W(k-1))^3] \quad (3.17)$$

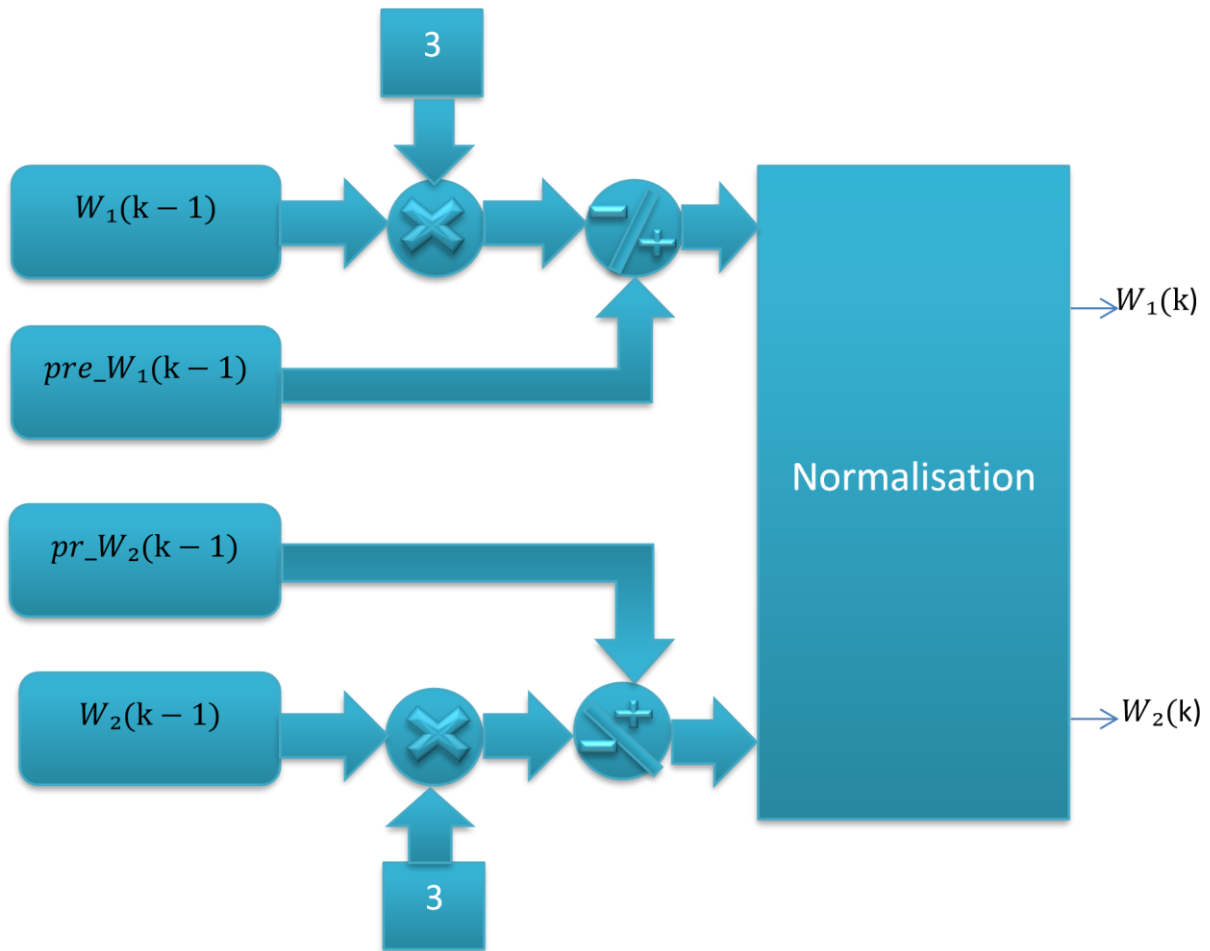


Figure 3.4 schéma fonctionnel de l'implémentation de  $W$

La valeur normalisée  $W\_new$  est comparée à l'ancienne valeur  $W\_old$  et si les valeurs ne correspondent pas, alors  $W\_new$  est renvoyé à l'entrée du bloc et également stocké comme  $W\_old$ . Quand  $W\_new = W\_old$  alors cette valeur est donnée à la sortie comme un vecteur convergé  $W$  qui donne une composante indépendante, un nouveau vecteur aléatoire  $W$  est supposé et il est décorrélé avec le  $W$  précédent et est de nouveau mis au processus d'itération pour obtenir une valeur convergente optimisée.

### 3.6 Implémentation sous Matlab

Nous avons utilisé deux sources de signaux s1 et s2 pour tester notre algorithme. Ces derniers sont un signal sinusoïdal et un signal triangulaire Les figures (3.5a) et (3.5b) sont les signaux sources.

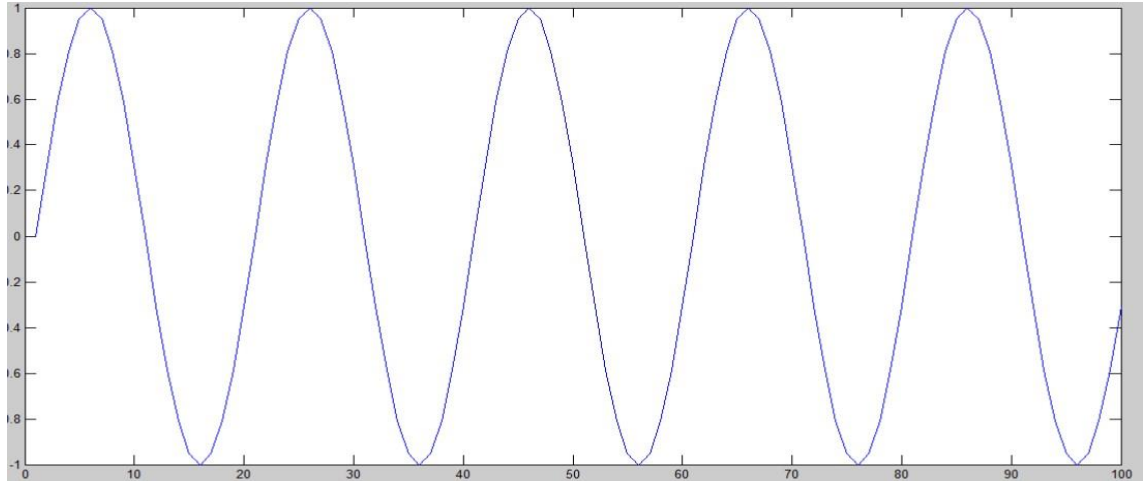


Figure 3.5a signal source sinusoïdale Matlab

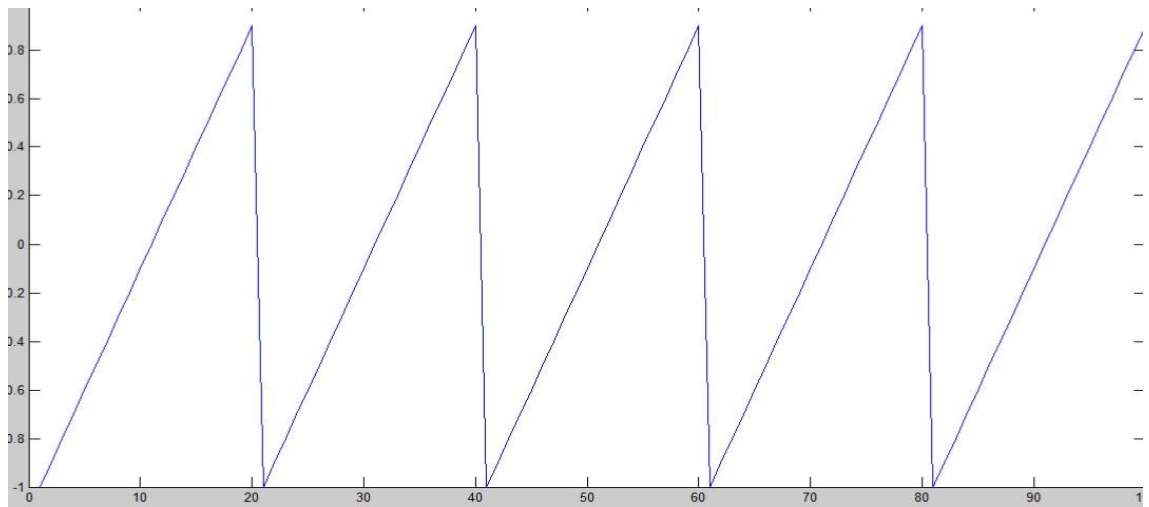
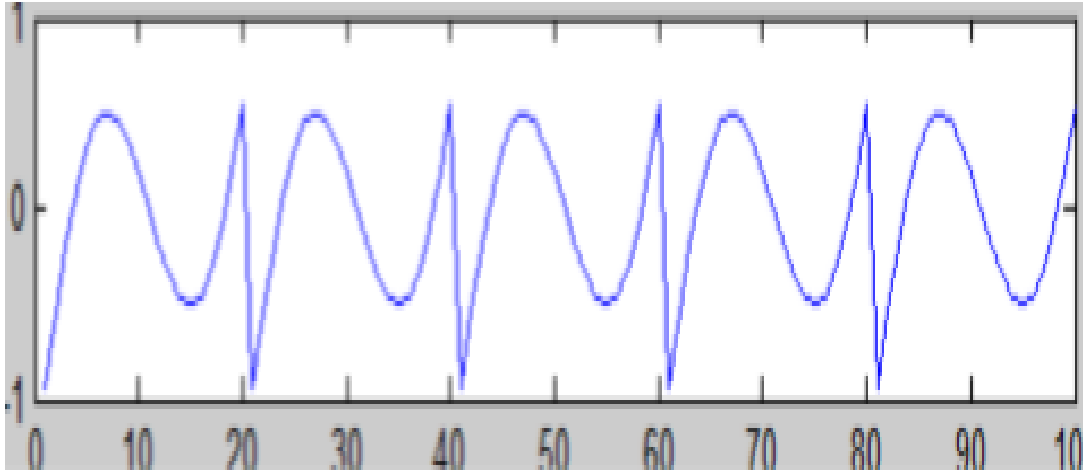


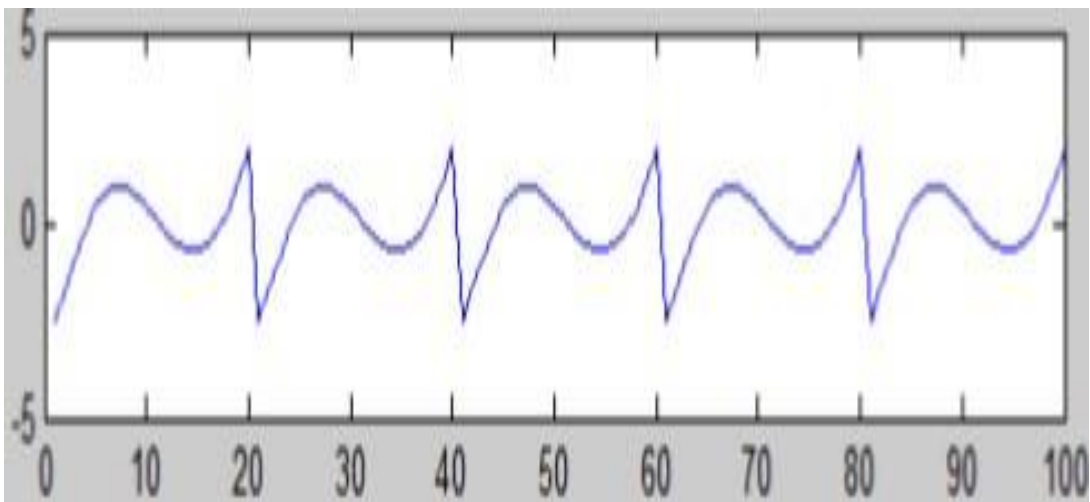
Figure 3.5b signal source triangulaire Matlab



Ensuite, les signaux mixtes sont produits en multipliant la matrice de mélange aléatoire et signaux source. Les figures (3.6a) et (3.6b) montrent les signaux mixtes. Pour le programme Matlab utilisé voir Annex « figure A ».

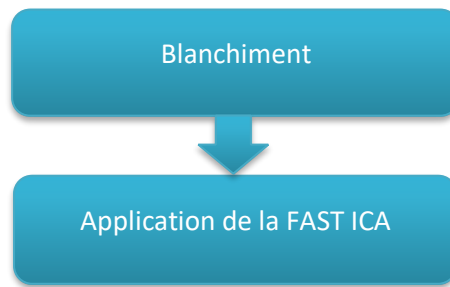


**Figure 3.6a signal mélange 1 Matlab**



**Figure 3.6b signal mélange 2 Matlab**

On rappelle la méthode :



**Figure 3.7** Algorithme de la BSS

Il s'agit de faire un blanchiment suivi de l'application de la méthode FAST ICA

### 3.6.1 Le blanchiment

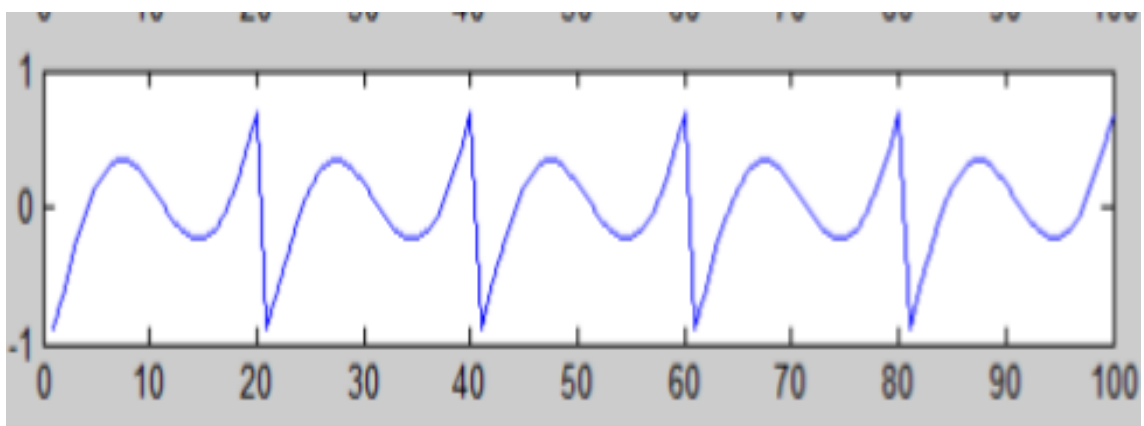
Voir la Figure. 3.2 représente le schéma fonctionnel, qui explique les étapes du blanchiment.

Centrage :	<pre> v = mean(x')'; g=x- v*ones(1,size(x,2)); </pre>
Covariance :	<pre> c00=0; c01=0; c10=0; c11=0; for j=1:length(x(1,:)) c00= c00+g(1,j)*g(1,j); end c00=c00/length(x(1,:)); for j=1:length(x(1,:)) c10= c10+g(1,j)*g(2,j); end c10=c10/length(x(1,:)); c01=c10;  for j=1:length(x(1,:)) c11= c11+g(2,j)*g(2,j); end c11=c11/length(x(1,:)); Cx=[c00 c01;c10 c11] sigma=(c11- c00)/(2*c10); if sigma&lt;0 sgn=-1; </pre>
	<pre> else sgn=1; end T=sgn/(abs(sigma)+sqrt(1+sigma^2));  C=1/sqrt(1+T^2); </pre>

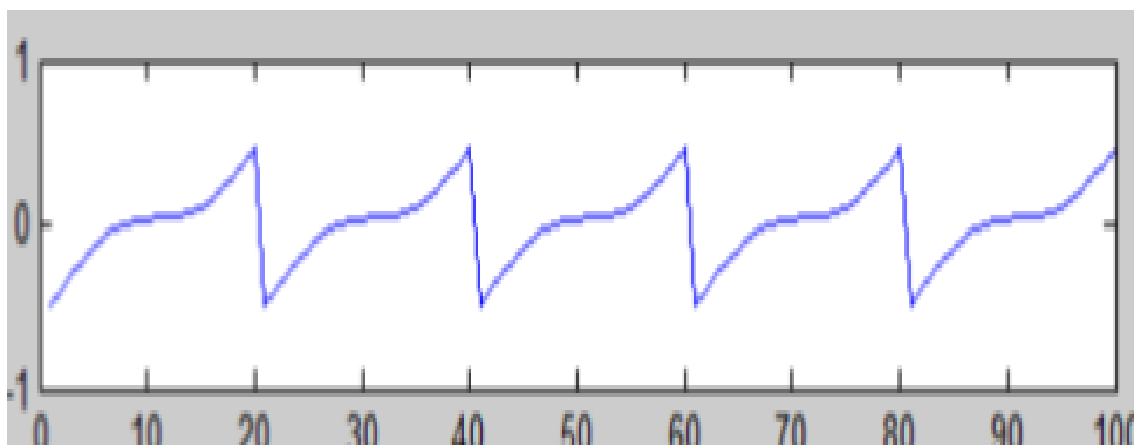
Valeurs propres et vecteurs propres de la covariance :	<pre> T=sgn/(abs(sigma)+sqrt(1+sigma^2)); C=1/sqrt(1+T^2); S=T*C; E=[-C S;-S C]; D=E'*[Cx*E] D11=1/sqrt(D(1,1)) D22=1/sqrt(D(2,2)) D1=sqrt(inv(D)) D1=[D11 0;0 D22]; P=D1*E';     c1 = cov(x',1); [E, D] = eig(c); </pre>
Blanchiment :	<pre> WhiteT = E*diag(diag(D).^(-0.5))*E'; Z = WhiteT*g; </pre>

**Tableau 3.1 programme Matlab de la partie blanchiment**

Les signaux mixtes centrés sont présentés dans les figures (3.8a) et (3.8b).



**Figure 3.8a signal mélange 1 centré Matlab**



**Figure 3.8b signal mélange 2 centré Matlab**

Les figure 3.9a et 3.9b représentent les signaux mélange blanchis

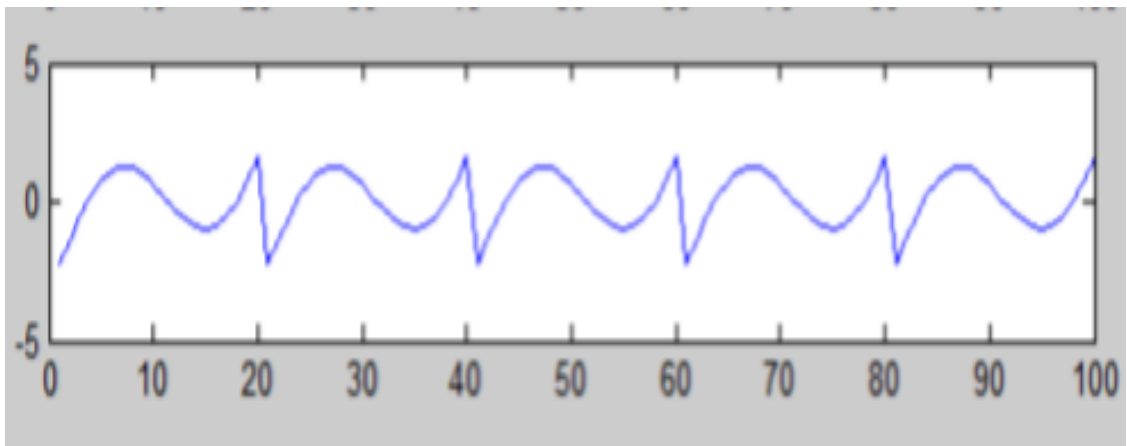


Figure 3.9a signal mélange 1 blanchis Matlab

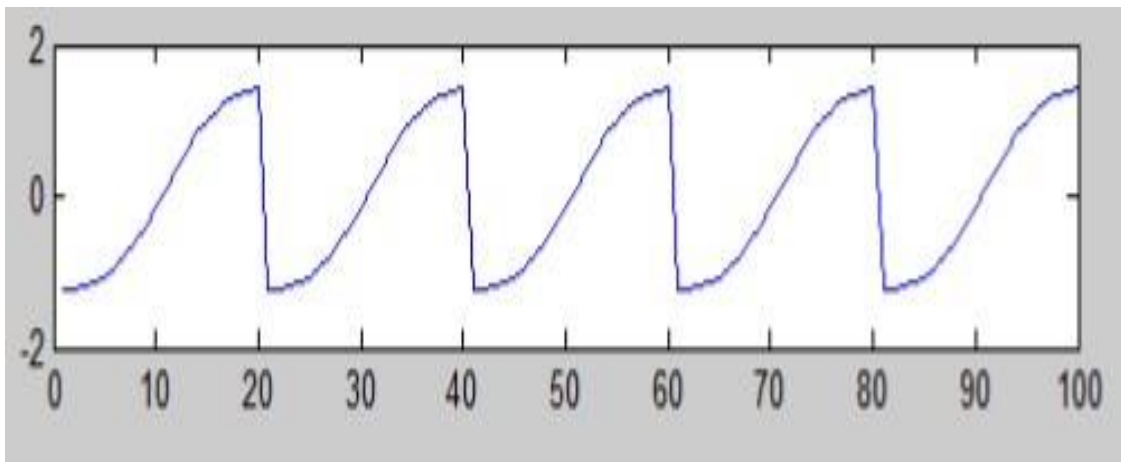


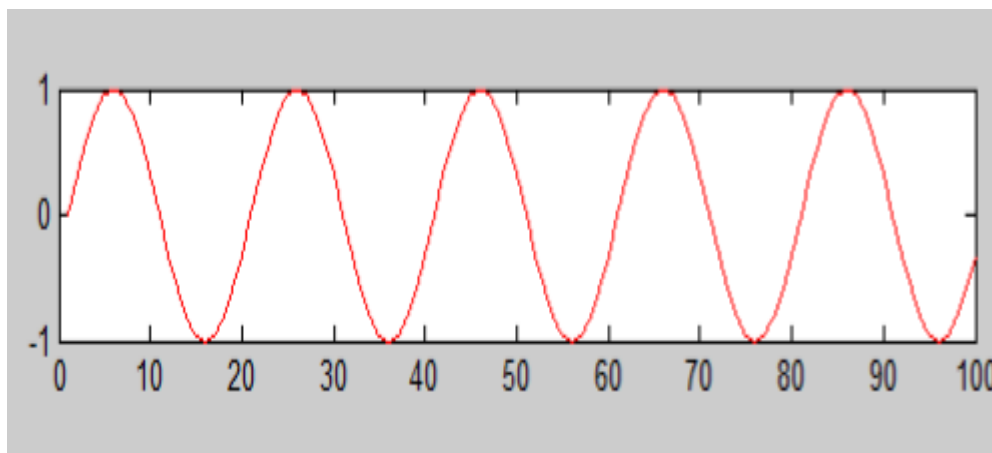
Figure 3.9b signal mélange 2 blanchis Matlab

### 3.6.2 La méthode FAST ICA

Méthode fast ica :	<pre> n=2; m = size(y1,1); W=[rand(1);rand(1)]; W=[2;5]; max=41; norm_W=sqrt((W(1))^2+(W(2))^2); w1=(W(1))/norm_W; w2=(W(2))/norm_W; w=[w1; w2]; Z1=Z(1,:); Z2=Z(2,:); w = randn(m, 1); w = E{xg(w^{T}*x)} - E{g'(w^{T}*x)}w w = w/norm(w, 2); end sound=(w'*Z); </pre>
--------------------	---

**Tableau 3.2 programme Matlab de la partie FAST ICA.**

La matrice de démixage est trouvée par l'algorithme FAST ICA et les signaux de composantes indépendantes estimés représentés sur les figures (3.10a) et (3.10b) sont obtenus en multipliant la matrice de démixtion et signaux mélange.



**Figure 3.10a Signal mélange 1 séparé Matlab**

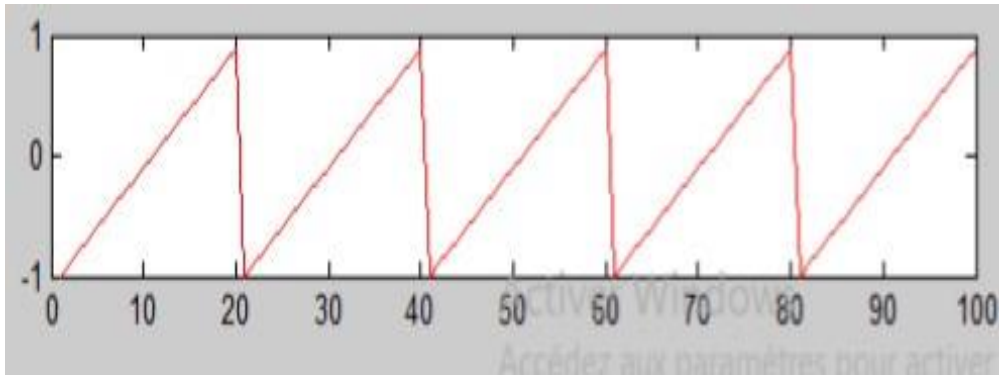


Figure 3.10b Signal mélange 2 séparé Matlab

### Commentaires

- Le programme de la fast ica ne fait aucun appel de fonction : il n'y a que des opérations de base
- Le programme a été appliqué à des signaux bien connus pour tester la fast ica qui ont bien répondu, autrement dit les résultats de la séparation sont positifs.

### 3.7 Implémentation sous Matlab Simulink

Le même travail est refait avec les blocs Simulink Les figures (3.11) montre les signaux sources.

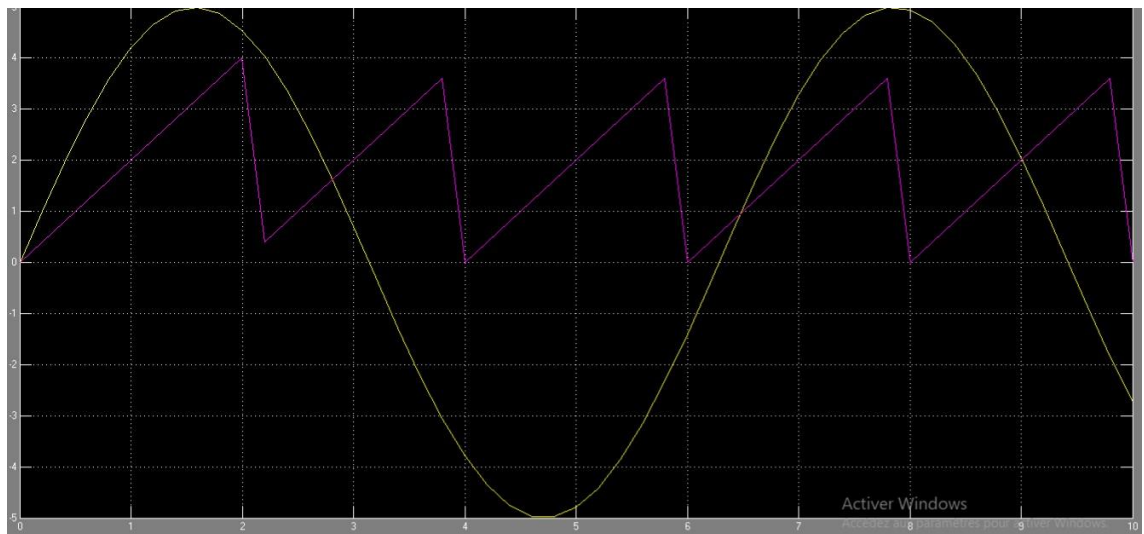


Figure 3.11 signaux source Simulink

Ensuite, les signaux mixtes sont produits en multipliant la matrice de mélange aléatoire et signaux source. La figures (3.12) montre les signaux mélange.

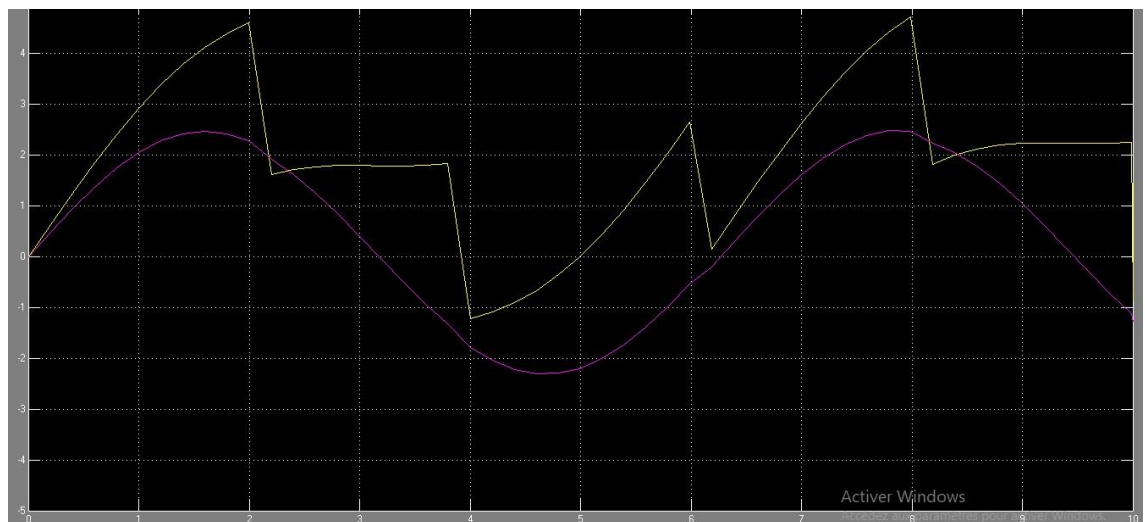


Figure 3.12 signal mélange Simulink

La figure (3.13) représente le schéma bloc utilisé pour obtenir le signal centré  $g$ .

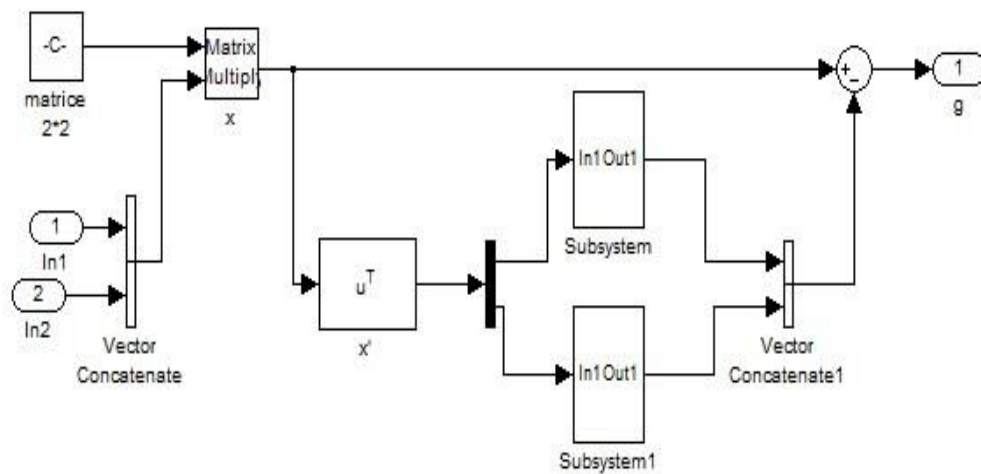


Figure 3.13 Schéma bloc du centrage

Les sous-systèmes correspondants sont présentés en annexe.

La figure 3.14 représente les signaux mixtes centrés

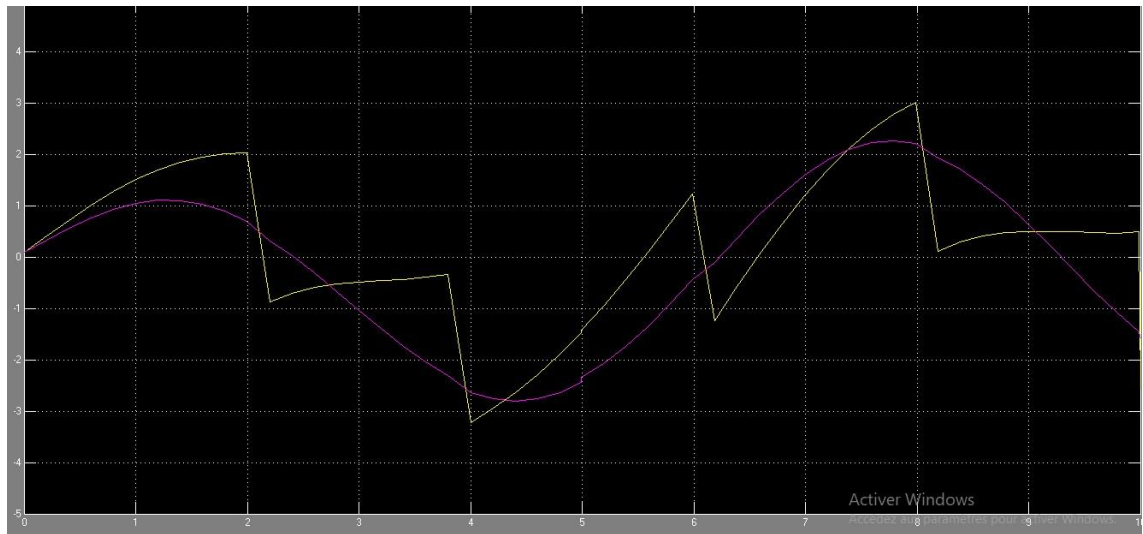


Figure 3.14 signal mélange centré Simulink

### 3.7.1 Le blanchiment

Le même travail de la partie blanchiment effectué sous Matlab se fait sous Simulink.

Le calcul des éléments  $c_x$  de la matrice de blanchiment sous blocs Simulink est montré dans la figure 3.15.

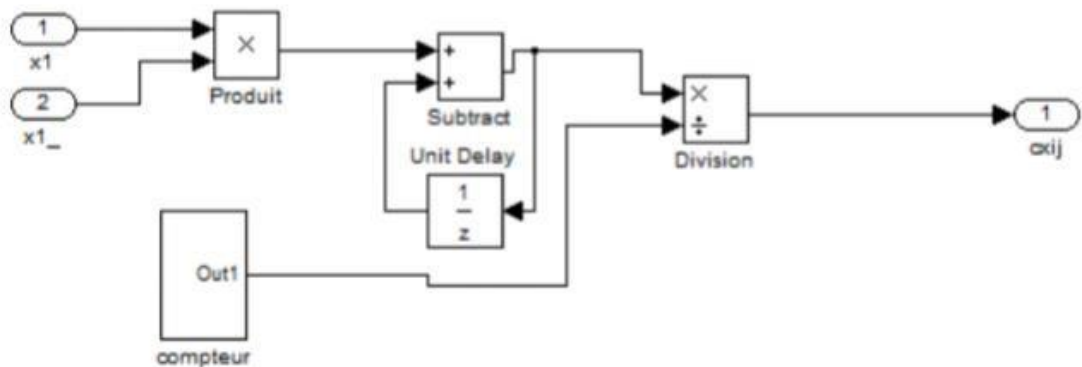


Figure 3.15 Schéma bloc du calcul des éléments de la matrice de covariance.

La matrice orthogonale des vecteurs propres orthogonale  $E$  et la matrice diagonale des valeur propre  $D$  sont ensuite calculées. La figures 3.16 montre le schéma bloc du calcul de la matrice  $E$ .



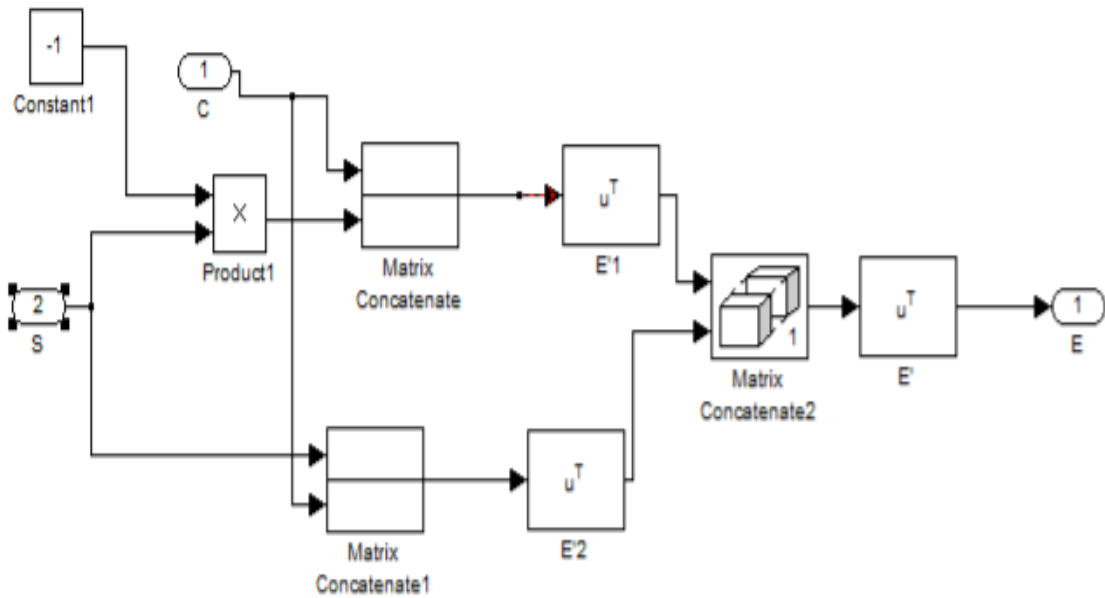


Figure 3.16 Schéma bloc de la détermination de la matrice E.

La génération des signaux blanchis se fait en utilisant les vecteurs et valeurs propres, ces derniers sont utilisés pour le calcul de la matrice de blanchiment P, la figure 3.17 introduit le schéma bloc correspondant.

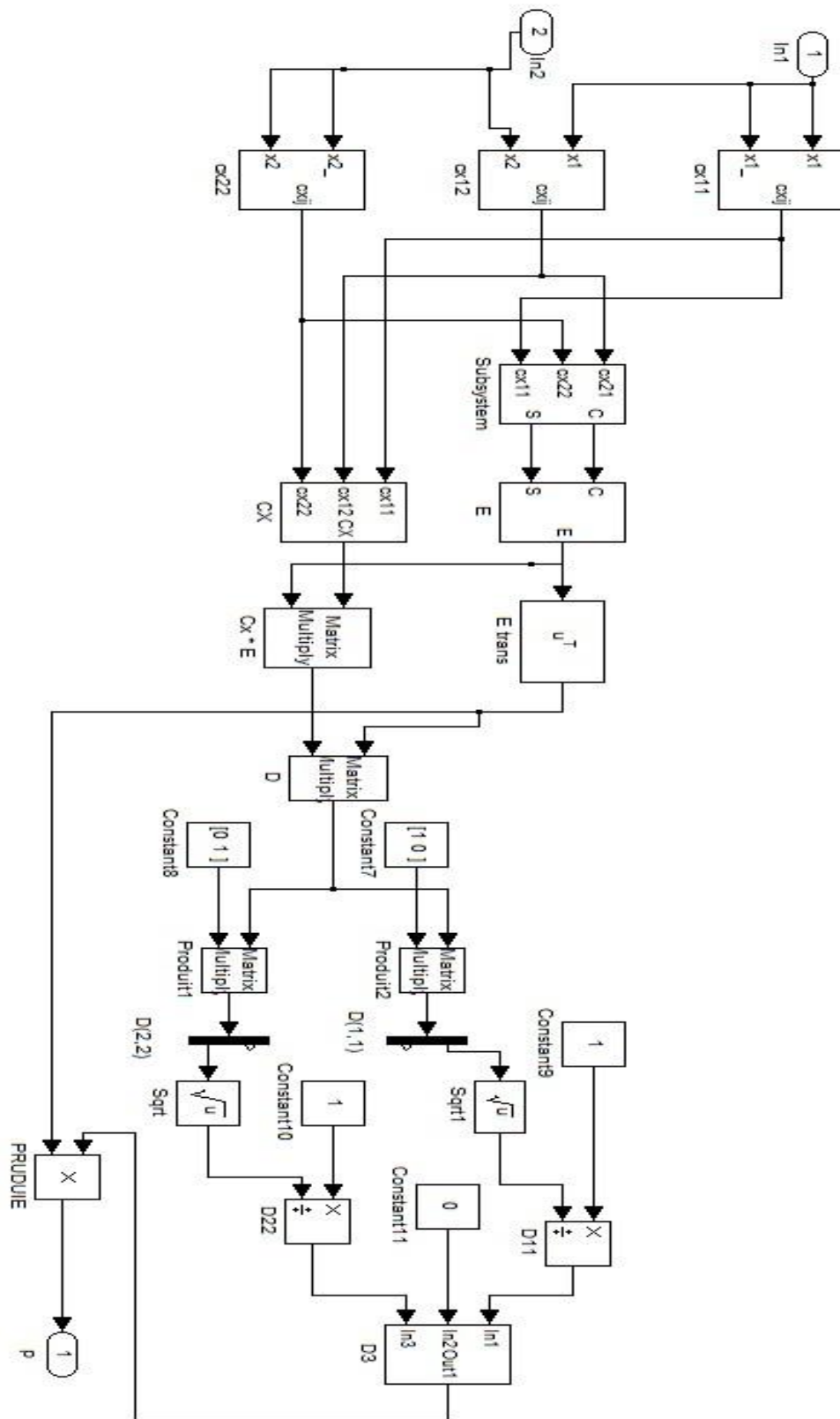


Figure 3.17 Schéma bloc du blanchiment.

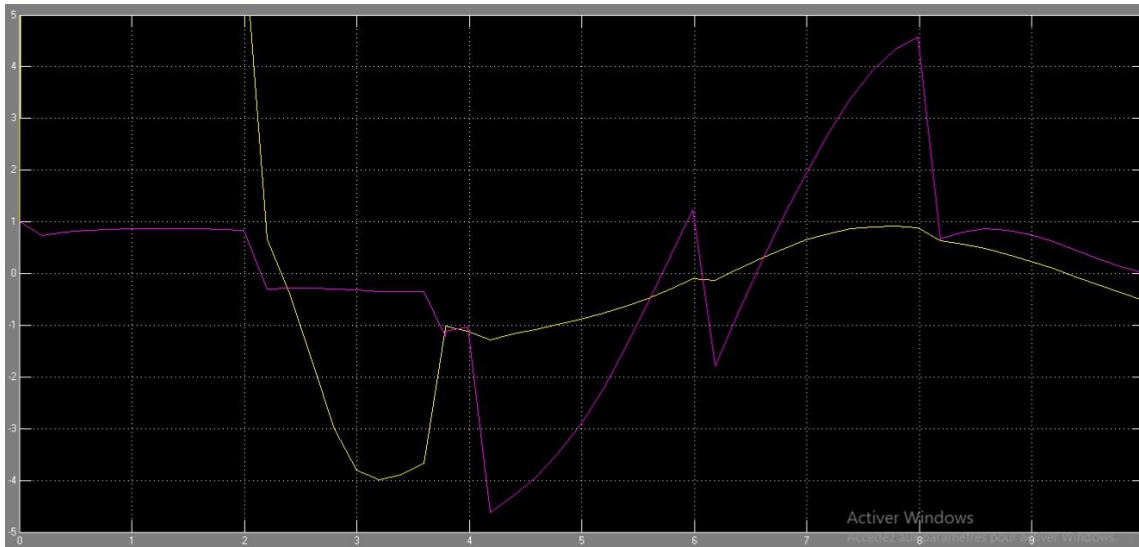


Figure 3.18 signal mélange blanchie.

### 3.7.2 La méthode FAST ICA

La matrice de démixtion est trouvée par l'algorithme FAST ICA (figure 21) et les signaux de composantes indépendantes estimés sont obtenus en multipliant la matrice de démixtion et les signaux mixtes (voir la figure 3.20), pour déterminer cette dernière on doit passer par le calcul des  $W$  comme le montre le schéma bloc de la figure 3.19 et figure 3.20.

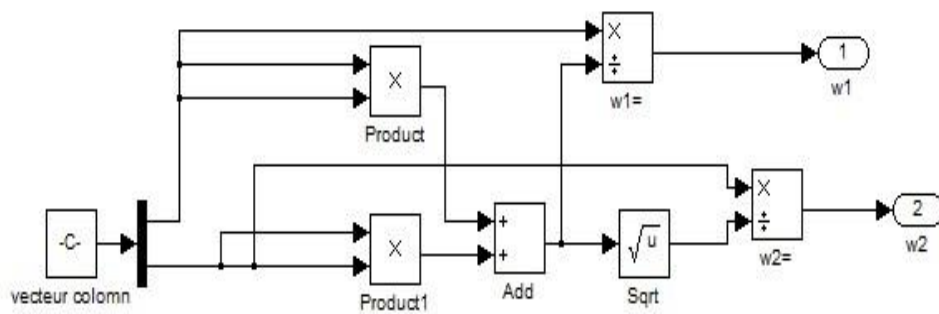


Figure 3.19 Schéma bloc génération des  $W$

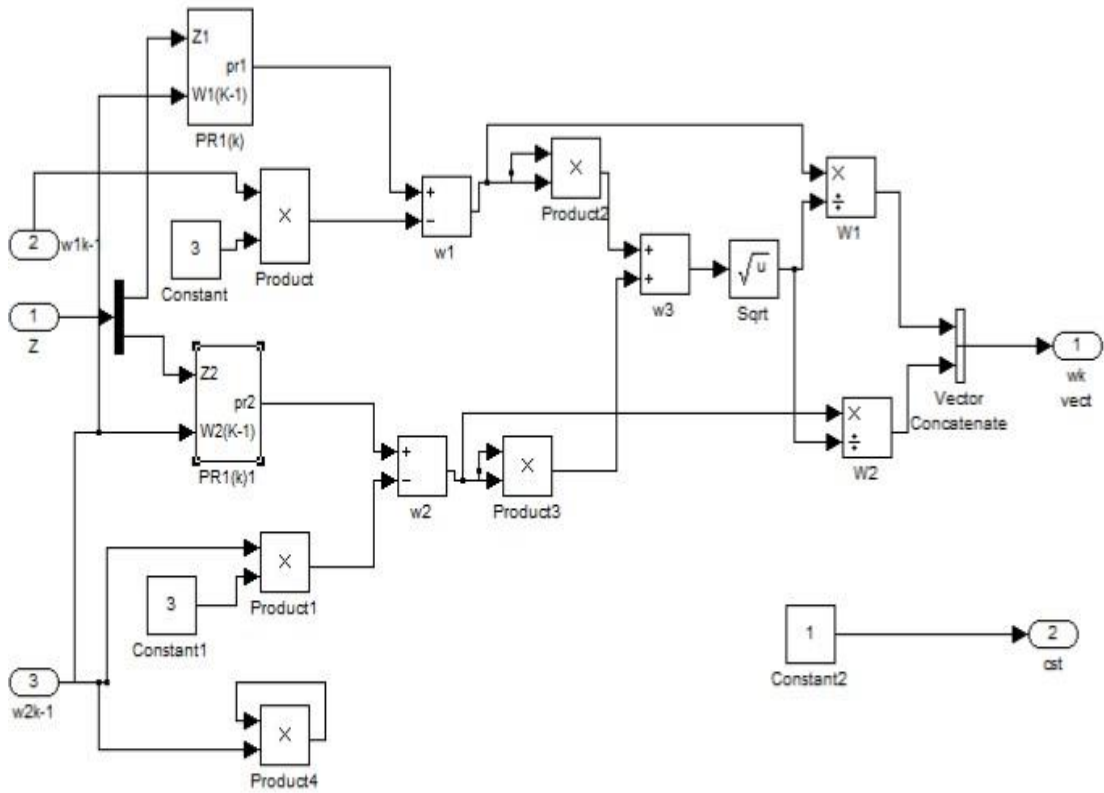


Figure 3.20 Schéma bloc du calcul des  $W_k$

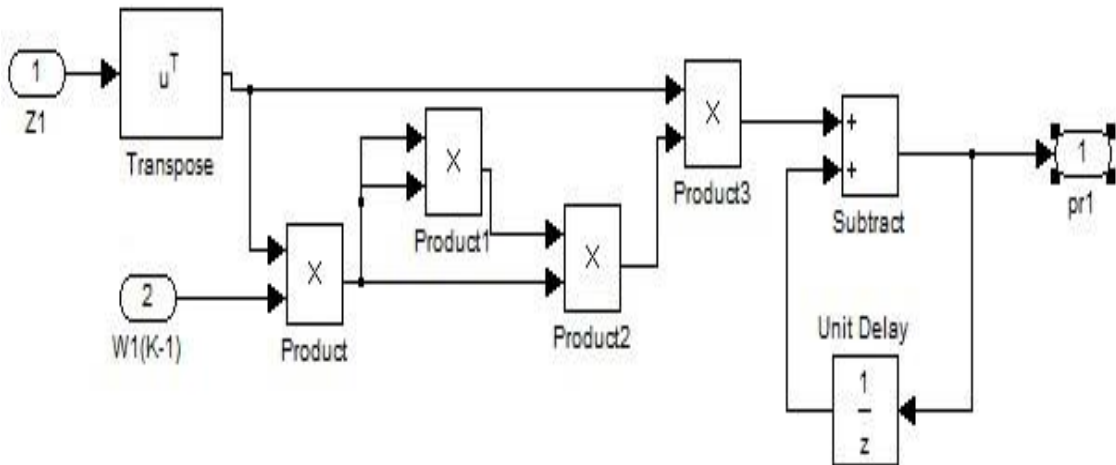


Figure 3.21 Schéma bloc du calcul des pré-w

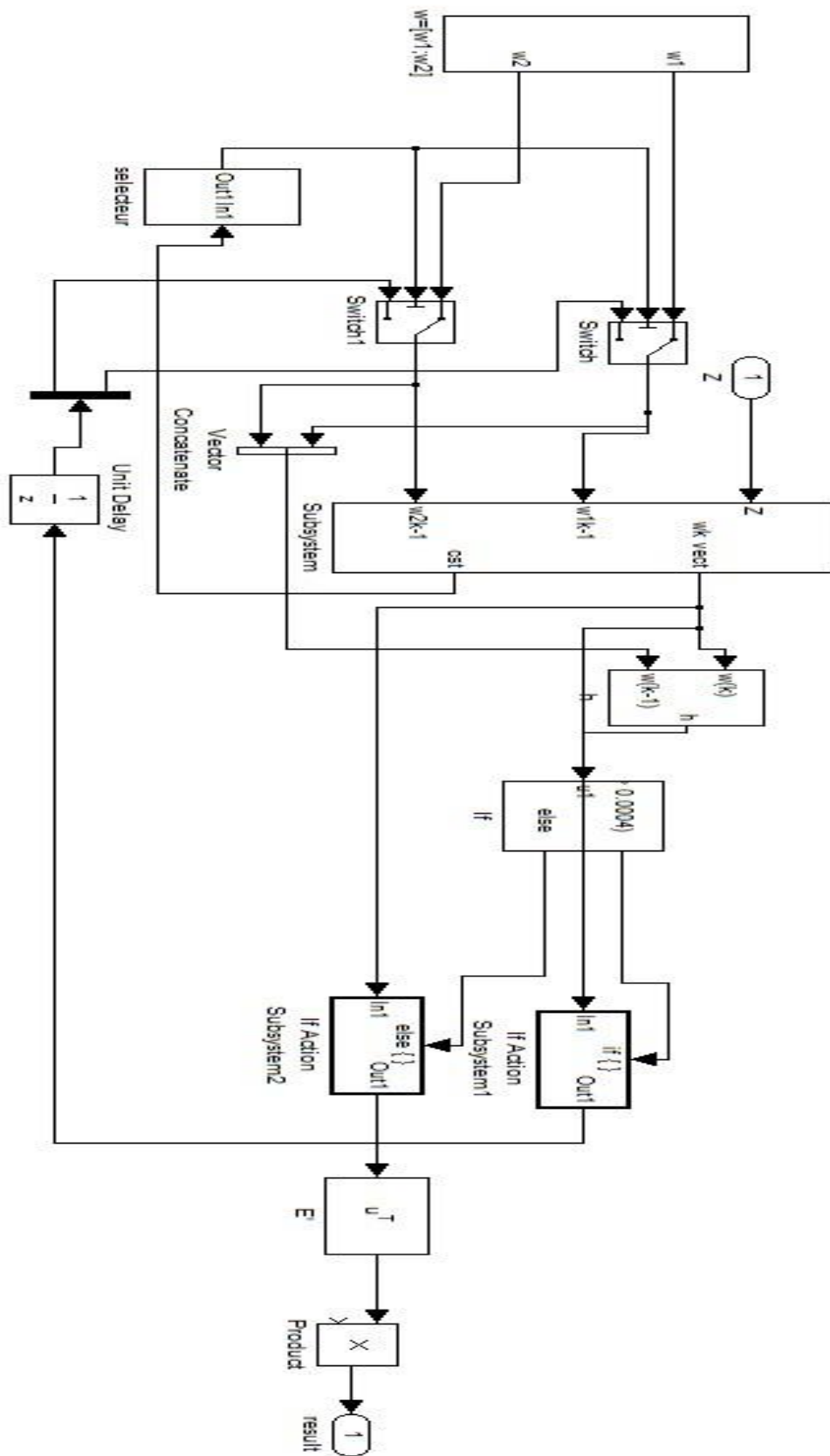


Figure 3.22 Schéma bloc de la méthode.

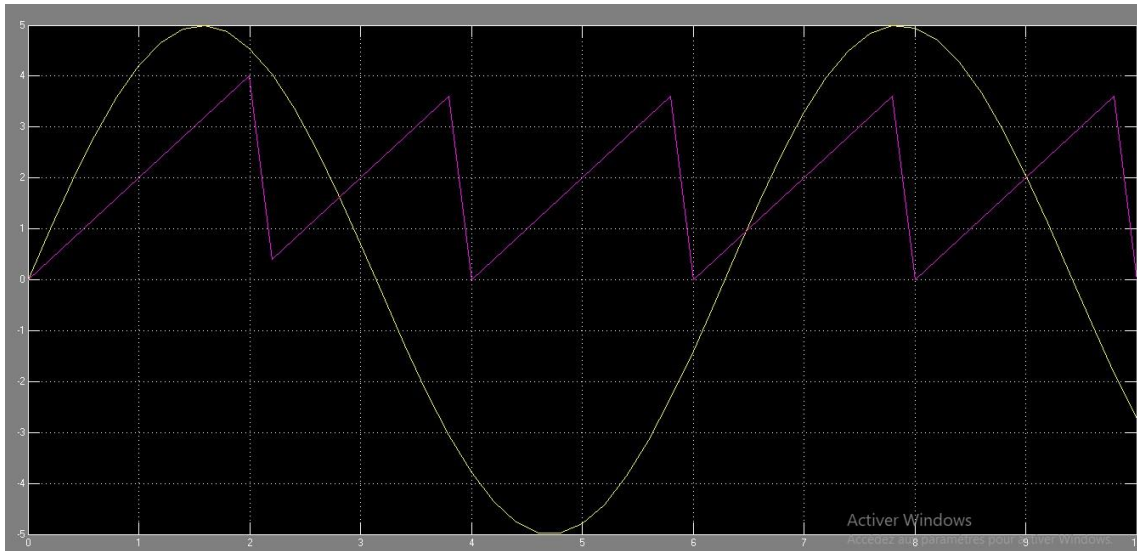


Figure 3.23 Signal mélange séparé sur Simulink

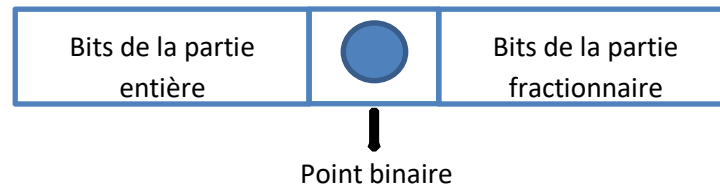
### Commentaires

- L'implémentation sous Simulink des différentes parties de la méthode a donné des résultats comparables à ceux fournis par l'implémentation Matlab.
- Nous allons adopter cette architecture dans la partie xilinx system generator.
- Comme remarque nous avons utilisé sous Simulink des blocs pouvant manipuler les matrices avec une grande souplesse. Ceci ne sera malheureusement pas le cas sous xilinx system generator.
- L'exécution de la fast ica en utilisant les signaux réels n'a pas pu se faire par manque de temps.

## 3.8 Simulation sous system generator

La conception de la FAST ICA sous xilinx system generator passe par plusieurs étapes. Il s'agit de faire un prétraitement, c'est le blanchiment suivi de la technique FAST ICA. En VHDL les variables doivent être clairement déclarées soit en virgule fixe ou en virgule flottante. La configuration en virgule flottante peut s'avérer inefficace pour notre implémentation, étant donné que cette dernière nécessite beaucoup de ressources pour la mise en œuvre du FPGA. Donc nous avons optés pour une déclaration de variable

en virgule fixe, ou la variable se compose d'une partie entière et une partie fractionnaire comme indiqué dans la figure 3.23.



**Figure 3.24 représentation du point fixe**

La longueur de mot a été sélectionnée en fonction de plusieurs tentatives de simulation. La plupart des résultats étaient erronés lorsqu'une petite longueur de mot était utilisée car cette dernière n'était pas suffisante pour représenter les valeurs. Après plusieurs tentatives de simulation, le choix de la taille du mot a été décidée pour être la même pour différents blocs d'implémentation. La longueur de mots a été définis à (64 :32), ce qui implique 64 bits avec 32 bits représentant la partie entière et 32bits représentant les bits fractionnaires. De cette façon, la partie entière peut représenter des nombres compris entre  $2^{32}$ .

### **3.8.1 Implémentation hard**

Les deux étapes à mettre en œuvre (blanchiment et Fast ICA) n'ont aucun contrôle l'une sur l'autre. Les données qui sont manipulées se présentent sous forme de paquets. Chaque paquet contient  $64 \times 100 \times 2$  bits de données. Pour des raisons de simulation, un seul paquet de 100 échantillons est utilisé pour tester notre implémentation.

#### ***a Blanchiment***

Le bloc de blanchiment (figure 3.21) comprend trois étapes. La première étape est le bloc de centrage (figure 3.22) La deuxième étape calcule la matrice de covariance des signaux centrés tandis que la troisième étape calcule les valeurs propres et les vecteurs propres de la matrice de covariance.

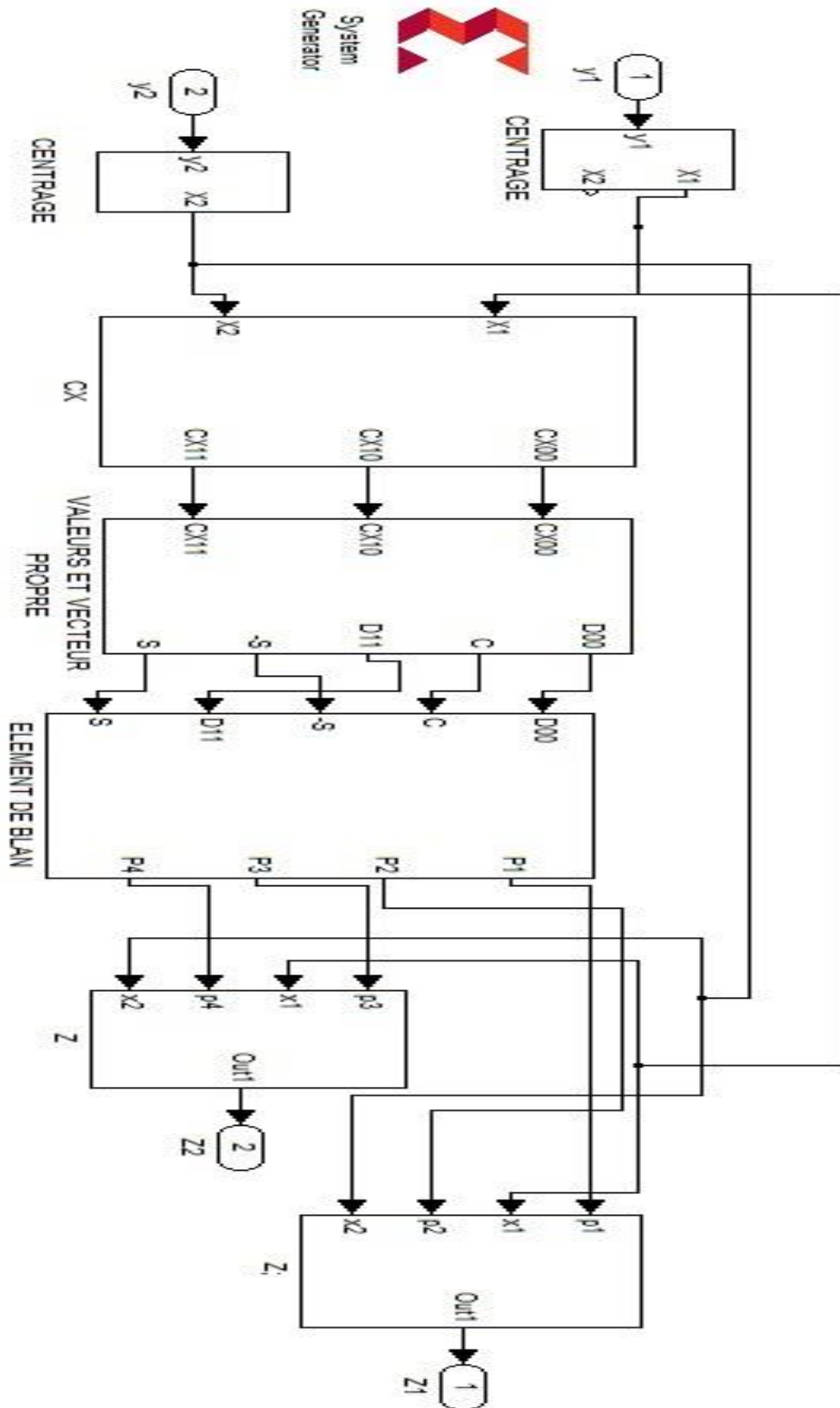


Figure 3.25 blanchiment sous system generator.

Les sous-systèmes de la figure 3.25 sont présentés en l'annexe.



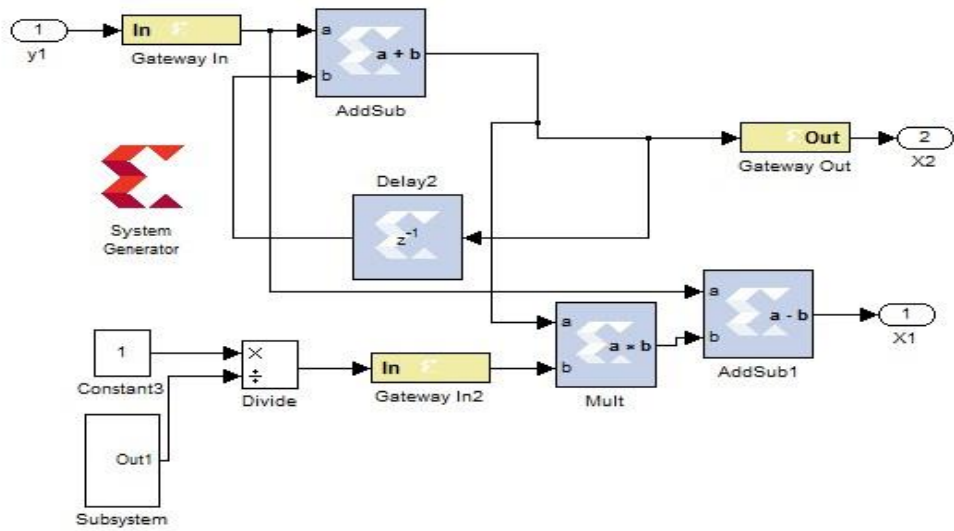


Figure 3.26 centrage sous system generator.

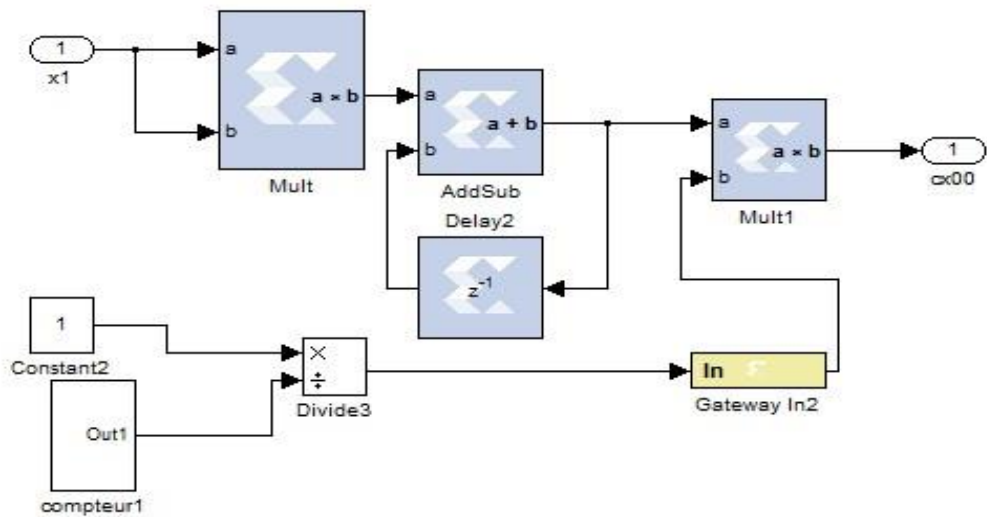


Figure 3.27 calcul des éléments de la matrice de covariance  $c_x$  sous system generator

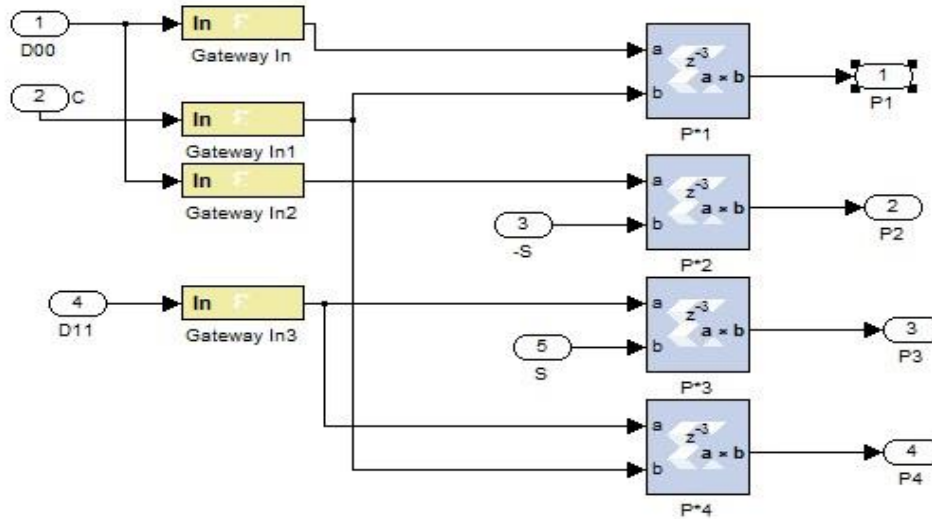
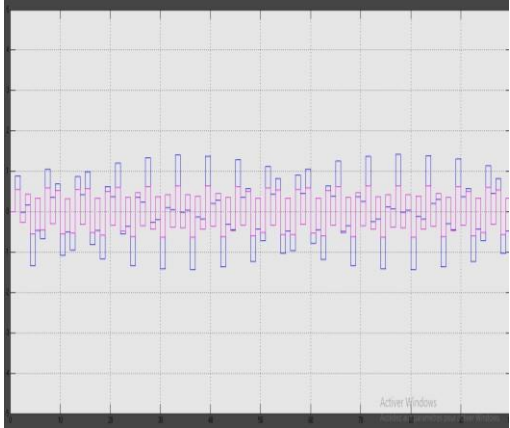
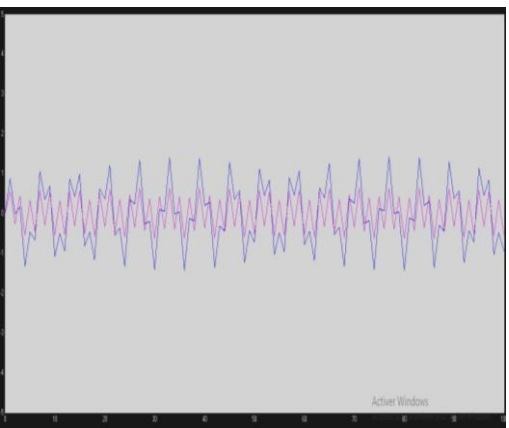
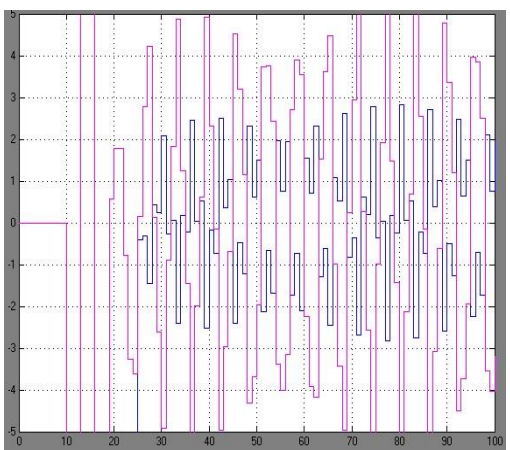
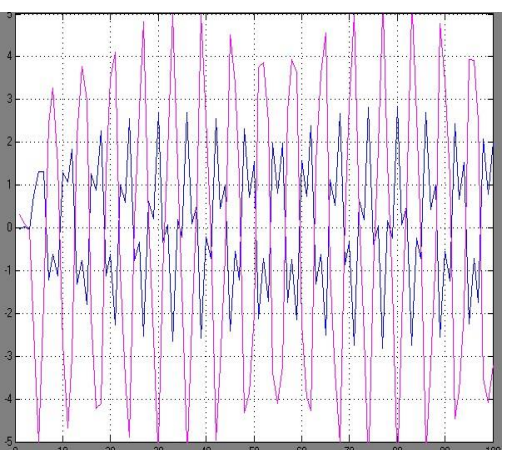


Figure 3.28 calcul des éléments de la matrice de blanchiment sous system generator.

### Commentaires

- Nous avons implémenté la partie blanchiment sous xilinx system generator, en décortiquant chaque bloc de notre architecture sous Simulink : toutes les opérations sur les tableaux ont été réalisées élément par élément.
- Pour le test des résultats nous avons considéré les blocs séparément d'une part et comparé les résultats de sortie de chaque bloc avec ceux obtenus en utilisant Simulink d'autre part, par conséquent :

	Résultats sous XSG	Résultats sous Simulink
Centrage		
Matrice de covariance	$\begin{bmatrix} 0.431 & 0.202 \\ 0.202 & 0.1265 \end{bmatrix}$	$\begin{bmatrix} 0.4316 & 0.2023 \\ 0.2023 & 0.1267 \end{bmatrix}$
Matrice Vecteurs propres E	$\begin{bmatrix} 0.4709 & -0.3124 \\ 0.3124 & 0.4709 \end{bmatrix}$	$\begin{bmatrix} 0.4756 & -0.3124 \\ 0.3124 & 0.4756 \end{bmatrix}$
Matrice diagonal valeurs propre D	$\begin{bmatrix} 2.111 & -0.899 \\ -0.899 & 0.4251 \end{bmatrix}$	$\begin{bmatrix} 2.119 & -0.912 \\ -0.912 & 0.3124 \end{bmatrix}$
Matrice de blanchi P	$\begin{bmatrix} 1.008 & 0.6554 \\ -0.1346 & 0.2059 \end{bmatrix}$	$\begin{bmatrix} 1.008 & 0.6668 \\ -0.1346 & 0.2049 \end{bmatrix}$
Signal blanchi		

### **3.9 Conclusion**

C'est la partie xilinx system generator qui s'est avérée particulièrement difficile. Ceci étant dû entre autres à la méconnaissance de l'outil lui-même (xilinx system generator). Malgré cela nous avons pu faire une première conception ou quelque module ont donné des résultats.

# Conclusion générale

---

Le cahier des charges de ce projet exigeait de nous une implémentation de l'algorithme de separation aveugle des sources sous VHDL en passant d'abord par Matlab et Matlab Simulink.

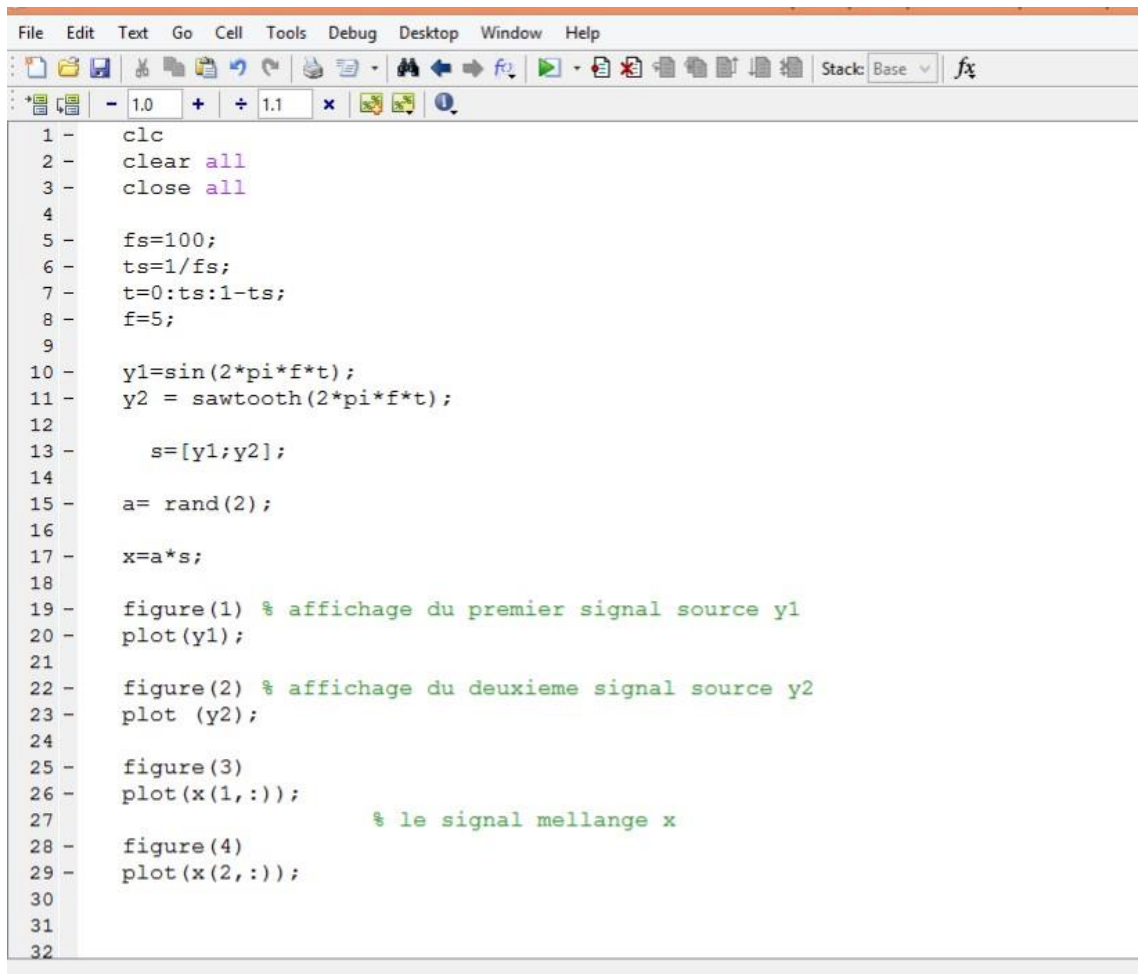
Nous avons implémenté la fast ica sous Matlab et Matlab Simulink avec succès, il faut noter que l'implémentation a été effectuée sans utilisation d'appel de fonction ceci dans le but de préparer l'implémentation de l'algorithme sous VHD.

Pour l'implémentation sous VHDL, nous avons opté pour l'utilisation de l'outil xilinx system generator étant donné sa souplesse de manipulation. Xilinx System Generator fournit également la fonctionnalité d'effectuer une Co-simulation matérielle lorsqu'une conception basée sur un modèle fonctionne sur Simulink et FPGA.

Cet outil nous a causé énormément de problèmes vue la méconnaissance de ce dernier, malgré cela nous avons pu réaliser avec beaucoup de difficultés quelques parties de l'algorithme, ce qui nous permet d'espérer la réussite de l'implémentation totale de la Fast ICA dans un futur très proche par d'autres étudiants qui seront intéressés par ce sujet. En résumé, on peut estimer le taux de respect des cahiers des charges a dépassé les quatre-vingt pourcent (80%).

Il est a noté que nous avons implémenté la fast ica en utilisant deux signaux connus, malheureusement le temps ne nous a pas permis d'aller au-delà de cette phase, et appliquer nos programmes a des signaux réels tels que les signaux ECG mère-fœtus, ce travail peut donc constituer une perspective future.

Durant la réalisation de ce projet, nous avons appris énormément de choses telles que la manipulation des blocs Simulink et des blocs Xilinx System Generator.













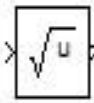


```
1 - clc
2 - clear all
3 - close all
4
5 - fs=100;
6 - ts=1/fs;
7 - t=0:ts:1-ts;
8 - f=5;
9
10 - y1=sin(2*pi*f*t);
11 - y2 = sawtooth(2*pi*f*t);
12
13 -     s=[y1;y2];
14
15 - a= rand(2);
16
17 - x=a*s;
18
19 - figure(1) % affichage du premier signal source y1
20 - plot(y1);
21
22 - figure(2) % affichage du deuxieme signal source y2
23 - plot (y2);
24
25 - figure(3)
26 - plot(x(1,:));
27 -                                     % le signal mellange x
28 - figure(4)
29 - plot(x(2,:));
30
31
32
```

Figure. A. Générer un signal mixte avec programme Matlab

## Blocs Simulink

On résume les blocs Simulink utilisés, leur symbole et fonction dans le tableau suivants (Tableau 3.1)

Nom du bloc	symbole	Fonction
Sine wave		Signal sinusoïdal
Demux		Extraire les composantes d'un signal vectoriel d'entrée et délivre des signaux séparés
Vector concatenate		Concaténer vecteur d'entrée
Scope		Afficheur de signaux
Matrix multiple		Faire la multiplication matriciel

Transpose		Transpose une matrice
Add		Sommer et soustraire
Unit DeLay		Retient et retarde sa saisie par la période d'échantillonnage
Dévide		Faire la division de sa première entrée par sa deuxième entrée
Constante		Génère une valeur constante réelle ou complexe
Square root		Calcule la racine carrée
Sign		Indique le signe de l'entrée
Display		Afficher des valeurs





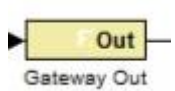

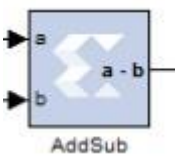
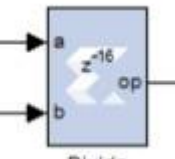
To Workspace		Entre le signal et écrit les données de signal dans un espace de travail
--------------	---	--

Tableau 1 blocs Simulink utilisés

## Blocs système generator

On résume quelques blocs system generator utilisés, leur symbole et fonction dans le tableau suivants (Tableau 3.2)

Nom du bloc	Symbole	Fonction
Gateway in		Entrée pour la partie de Xilinx du modèle Simulink
Gateway out		Convertir les données en virgule fixe à virgule flottante.
Constant		Génère une valeur constante
Addsub		Sommer et soustraire
Devide		Faire la division de sa première entrée par sa deuxième entrée


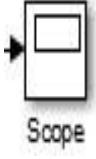
System generator		Compiler la conception de Xilinx dans le matériel
Scope		Afficheur de signaux

Tableau 2 blocs system generator utilisés

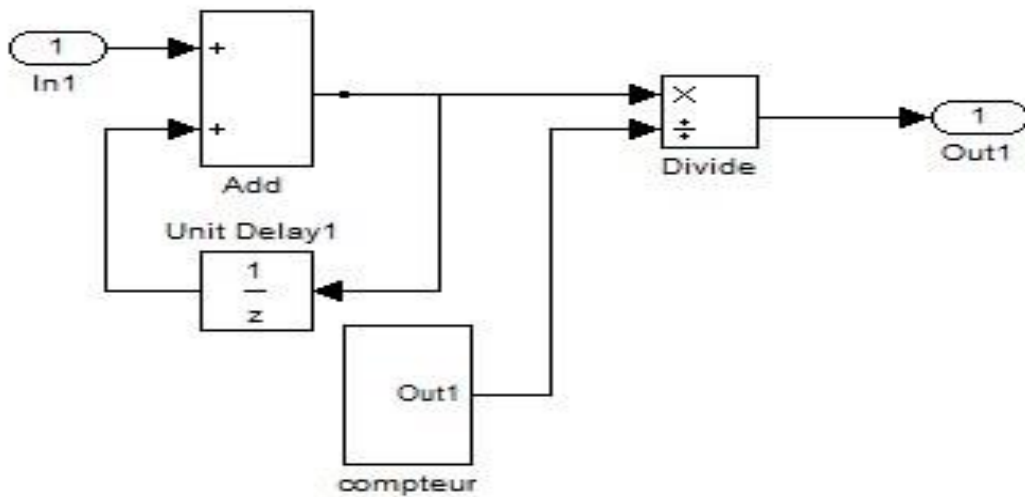


Figure B. schéma bloc Simulink du sous system du centrage.

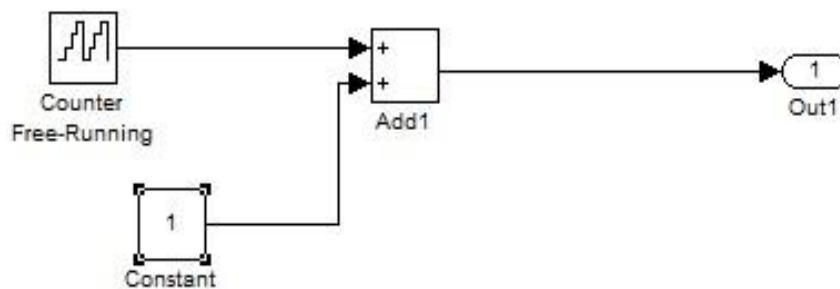


Figure C. schéma bloc Simulink du compteur.

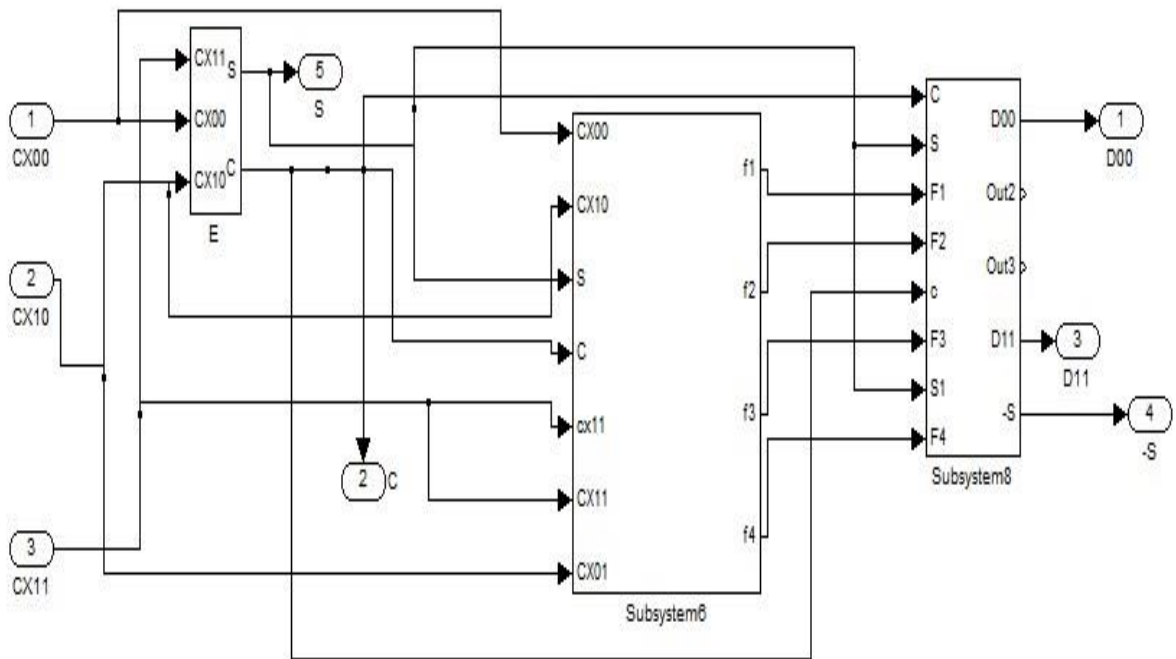


Figure D. schéma bloc system generator du calcul des valeurs et vecteurs propres. Avec  
 $f = C_x \times E$

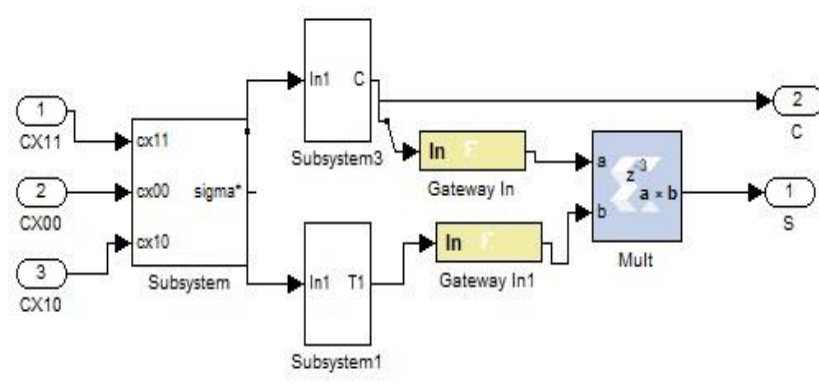


Figure E. schéma bloc system generator du calcul des éléments de E.

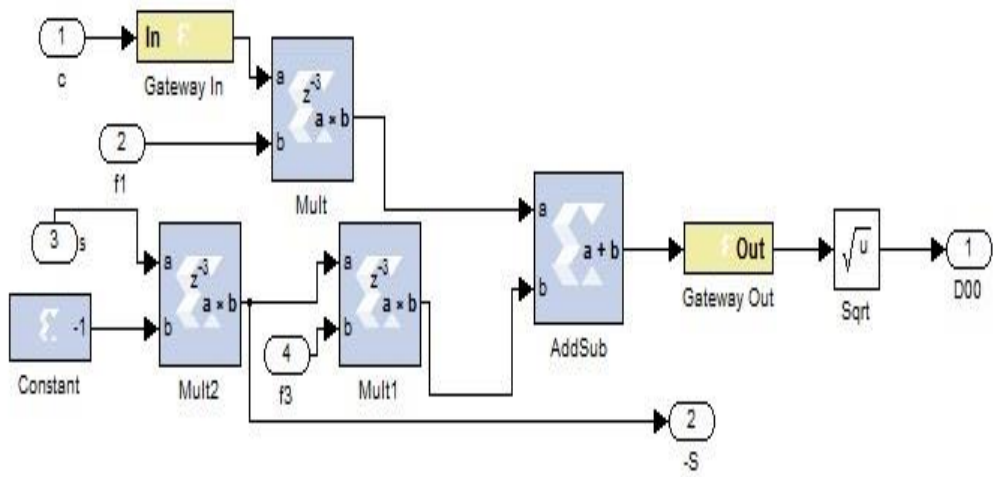


Figure F. schéma bloc system generator du calcul des éléments de D.

# Bibliographie

---

- [1]. Aapo Hyvärinen and Erkki Oja Neural Networks Research Centre Helsinki University of Technology P.O. Box 5400, FIN-02015 HUT, Finland Neural Networks, 13(4-5):411-430, 2000
- [2]: J.F.Cardoso, —Blind Signal Separation: Statistical Principles||, Proc. of IEEE, vol9, No.10, pp. 2009-2025, 1998.
- [3] Kuo-Kai Shyu,Ming-Huan Lee, Yu-Te Wu, Po-Lei Lee — Implementation of pipelined FastICA on FPGA for Real-time Blind source separation||. IEEE Trans. Neural Netw.,vol.19,no.6,June 2008.
- [4] Francisco Castells, Pablo Laguna, " Principal Component Analysis in ECG SignalProcessing"Hindawi Publishing Corporation EURASIP Journal on Advances in Signal ProcessingVolume 2007, Article ID 74580, 21 pages
- [5] A. Hyv"arinen. One-unit learning rules for independent component analysis: A statistical analysis. Advances in Neural Information Processing Systems, 9:480–486, 1997. [6] A. Cichocki and S. Amari. Adaptive Blind Signal and Image Processing,. Wiley, 2002.
- [7] C. M. Bishop. Neural Networks for Pattern Recognition. Clarendon Press, 1995.
- [8] A. Hyv"arinen. Beyond independent components. In Proc. Int. Conf on Articial Neural Networks, Edinburgh, UK, pages 809–814, 1999.
- [9] Aapo hayvarinen "Independent Component analysis". Adaptive and Learning Systems for Signal Processing, Communications, and Control Published Online: 15 May 2002.

- [10] J.-F. Cardoso. Blind signal separation: Statistical principles. Proceedings of the IEEE, 86:2009–2025, Oct. 1998.]:
- [11] A. Papoulis. Probability, random variables and stochastic processes. McGraw-Hill, 2 edition, 1984
- [12] A. Hyv"arinen, J. Karhunen, and E. Oja. Independent Component Analysis. John Wiley Sons, 2001.
- [13] S. C. Douglas. Blind source separation and independent component analysis: A crossroads of tools and ideas. In 4th International Symposium on Independent component analysis and blind signal separation (ICA2003), pages 1–10, Nara, Japan, 2003.
- [14] A. Hyv"arinen. Survey on independent component analysis. Neural Computing Surveys, 2:94–128, 1999.
- [15]. COUILLEZ Sandrine Ecole I.A.D.E. de Reims 1"ere ann"ee Promotion 2007- 2009 LE TRACE ELECTRIQUE CARDIAQUE DYNAMIQUE Le 16 mai 2008.
- [16] <https://www.riouflatglass.com/procede-float.html>
- [17] NOIROT SIMON, commande par FPGA : de la modalisation a l'impl"ementation, sur site web <http://espace.etsmtl.ca>.
- [18] steven derrien etude quantitative des techniques de partitionnement de r"eseaux de processeurs pour l'impl"ementation sur circuits FPGA, sur site web <http://www.irisa.fr>.
- [19] JEAN-GABRIEL MAILLOUX prototypage rapide de la commande vectorielle sur FPGA "a l'aide des outils simulink-system generator, sur site web <http://constellation.uqac.ca>
- [20] P. Comon. Independent component analysis: A new concept? Signal Processing, 36:287– 314, Apr. 1994.
- [21] spartan 3A 1800DSP starter platform User Guide, Xilinx inc., June 23, 2009.
- [22] MEZIANE Hadj Boum"edi"ene. Acquisition de signaux Electrocardiogrammes (ECG) "a l'aide de la carte DSPACE 02 juillet 2003 ING"ENIEUR D'ETAT ELECTRONIQUE

[23] Rangayyan Rangaraj M., —Biomedical Signal Analysis: A case study approach||, Wiley-IEEE press, December 2001.

[24]. Dragoş-Daniel Țarălungă. Extraction foetale d'ECG à partir de signaux abdominaux: revue de la suppression des composantes fondamentales des interférences de lignes de puissance et de leurs harmoniques.2014.