

TABLE DES MATIÈRES

	Page
INTRODUCTION	1
CHAPITRE 1 REVUE DE LITTÉRATURE	3
1.1 Oscillation hautes-fréquences	3
1.2 Exemple d'HFOs typiques	4
1.3 Intérêt sur l'hippocampe	6
1.4 Intérêt sur les neurones GABAergiques	10
1.5 Activité gamma des neurones GABAergiques	11
1.6 Détection automatique des HFOs	13
1.7 Détecteur initial	16
1.8 Définition des seuils	18
1.9 Conclusion	22
CHAPITRE 2 LA DÉTECTION VISUELLE	23
2.1 Introduction	23
2.2 Méthodologie	23
2.3 Exemple de marqueurs	28
CHAPITRE 3 LE DÉTECTEUR SUPERVISÉ	33
3.1 Objectif	33
3.2 La base de données	34
3.3 Variabilité temporelle	34
3.4 Stabilité d'amplitude	38
3.5 Stabilité spatiale	43
3.6 Conclusion	47
CHAPITRE 4 APPRENTISSAGE	49
4.1 Objectif et méthodologie	49
4.2 Résultats actuels	52
CHAPITRE 5 CLASSIFICATION FINALE	57
5.1 Introduction au score	57
5.2 La proximité des pointes épileptiques	58
5.3 La proximité spatiale	60
5.4 La durée maximale de la détection	62
5.5 Résultats	66
CONCLUSION	71
RECOMMANDATIONS	73

ANNEXE I	CODE MATLAB – CLASSE SUBJECT_HFO	75
ANNEXE II	CODE MATLAB – CLASSE C_DETECTOR	81
ANNEXE III	CODE MATLAB – APPRENTISSAGE SEUIL DE DÉTECTION	89
ANNEXE IV	CODE MATLAB – PRÉPARATION DES DONNÉES	109
ANNEXE V	CODE MATLAB – EXEMPLE COMPLET	117
BIBLIOGRAPHIE		119

LISTE DES TABLEAUX

	Page
Tableau 1.1	Résultats du détecteur initial publié19
Tableau 1.2	Résultats détecteur initial sur nouveaux patients.20
Tableau 3.1	Base de données des patients utilisés dans ce travail.....34
Tableau 4.2	Ratio d'énergie relative à l'activité de fond.39
Tableau 4.1	Comparaison détecteur initial et optimisé.....53
Tableau 4.2	Comparaison détecteur initial et optimisé sans PJ et DR.55
Tableau 5.1	Statistique de cooccurrence.....59
Tableau 5.2	Précision des marqueurs61
Tableau 5.3	Durée moyenne des marqueurs63
Tableau 5.4	Résultats classification finale.....68

LISTE DES FIGURES

	Page
Figure 1.1 HFO Gamma	4
Figure 1.2 HFO Ripple	5
Figure 1.3 L'hippocampe	6
Figure 1.4 Activité in vitro (bas) vs EEG intracrânien (haut)	7
Figure 1.5 Schéma de l'hippocampe	9
Figure 1.6 Fonctionnement normal et anormal des neurones GABAergics.....	10
Figure 1.7 Oscillation gamma (GFO) pre-ictale dans l'hippocampe.....	11
Figure 1.8 Séquence d'activation des neurones durant une GFO	12
Figure 1.9 Quatre exemples de vrais HFO gamma filtrés	14
Figure 1.10 Décomposition d'un HFO (dans l'encadré) en bandes étroites	16
Figure 1.11 Méthodologie de détection.....	17
Figure 2.1 Exemple de HFO Gamma	24
Figure 2.2 Exemple de HFO ripple	25
Figure 2.3 Pointe filtrée vs HFO filtré	27
Figure 2.4 Vrai HFO (gamma)	28
Figure 2.5 Vrai HFO (ripple)	28
Figure 2.6 Faux négatif (gamma)	29
Figure 2.7 Faux négatif (ripple)	29
Figure 2.8 Exemple faux négatif (gamma) dont on peut douter de l'existence	30
Figure 2.9 Exemple faux négatif (ripple) dont on peut douter de l'existence.....	31
Figure 3.1 Stabilité temporelle SBL.....	35

Figure 3.2	Stabilité temporelle PH	35
Figure 3.3	Stabilité temporelle (CML)	36
Figure 3.4	Stabilité temporelle (JC)	37
Figure 3.5	Énergie relative de sous-ensembles (SBL)	40
Figure 3.6	Énergie relative de sous-ensembles (CML)	41
Figure 3.7	Énergie relative de sous-ensembles (PH)	42
Figure 3.8	Canaux marqués de références (patient SBL)	44
Figure 3.9	Cartes spatiales 40-20	45
Figure 3.10	Cartes spatiales 10 min	46
Figure 4.1	Diagramme d'apprentissage du détecteur	50
Figure 4.2	Seuils optimisés par apprentissage	51
Figure 4.3	ROC apprentissage	54
Figure 5.1	Taux des marqueurs de l'expert	59
Figure 5.2	Distribution des scores (SBL)	65
Figure 5.3	Impact du score (CML)	66
Figure 5.4	Impact du score (MA)	67

LISTE DES ABRÉVIATIONS, SIGLES ET ACRONYMES

HFO	Oscillation haute-fréquence (high frequency oscillation)
EEG	Électroencéphalogramme
FPR	Taux de faux positives (false positive rate)
ILE-	Évènement comme-ictal (ictal-like event)
SLE	Évènement au niveau sujet (subject level event)
GFO	Oscillation de fréquence gamma (gamma-frequency oscillations)
HS	Neurone hippocamposeptal
GABA	Neurone gamma-aminobutyrique
FIR	Filtre à réponse impulsionnelle finie
Tb	Fenêtre d'activité de fond (background)
Ts	Fenêtre tampon
Td	Fenêtre de détection
Fc	Fréquence centrale d'une bande étroite
RMS(t)	Valeur RMS d'une bande étroite à l'instant t
RMS_b(t)	Valeur RMS de l'activité de fond d'une bande étroite à l'instant t
D_f(t)	Vecteur de détection
TH	Seuil de détection

LISTE DES SYMBOLES ET UNITÉS DE MESURE

Hz	Hertz
mV	milli Volt
μ V	micro Volt
RMS	root-mean-square (énergie)
sec	seconde
t	temps

INTRODUCTION

L'épilepsie est une maladie neurologique caractérisée par des changements soudains et anormaux du comportement du cerveau d'un individu. Ces crises se manifestent sous plusieurs formes, dont des convulsions et des pertes de mémoire. Au Canada, environ 0.6% de la population en est atteinte et 42 cas sont diagnostiqués chaque jour (Epilepsie Canada, 2015). Parmi ces patients, environ 30% sont pharmacorésistants, c'est-à-dire pour lesquels les médicaments pour prévenir les crises sont peu ou pas efficaces. Ces cas nécessitent une chirurgie, qui consiste dans l'ablation locale de la zone qui cause les crises épileptiques, que l'on appelle *foyer épileptogène*.

C'est principalement pour ces cas qu'autant d'effort est mis dans la localisation du foyer épileptogène. Les neurologues prônent l'analyse des électroencéphalogrammes de scalp (EEG) pour la première localisation et, dans le doute, les EEG intracrâniens pour une localisation plus précise. Cette pose d'électrodes intracrâniennes nécessite une opération invasive, la première localisation est donc primordiale pour éviter une opération inefficace. La méthode conventionnelle est par la détection de pointes épileptiques, mais celles-ci sont souvent diffuses sur beaucoup de canaux, rendant la localisation difficile. Plusieurs recherches s'intéressent à une autre activité épileptique, plus difficile à détecter, mais qui offre une information plus focale de la zone recherchée. Parmi ces nouveaux biomarqueurs, on trouve les oscillations hautes fréquences (HFO).

Les HFO sont des événements de courte durée, d'au moins quatre oscillations, dont l'amplitude est beaucoup plus petite que l'activité cérébrale normale. Elles sont généralement détectées sur les données intracrâniennes, là où l'effet du bruit est beaucoup plus faible que sur le scalp. L'intérêt du travail présenté dans cet ouvrage est de pouvoir détecter cette activité sur le scalp, où cette information pourrait être utilisée avant la pose des électrodes intracrâniennes (J.D. Jirsh *et al.* (2006), Melani *et al.* (2013)).

Comme ces oscillations se font dans une plage de fréquences où l'activité cérébrale régulière est rare, soit plus de 40 Hz, il est possible de filtrer pour plus facilement détecter les HFO. Comme la plage de fréquence des HFO s'étend des ondes gamma (40-80 Hz) jusque dans les ripples (80-200 Hz), une détection manuelle d'un neurologue doit se faire à l'aide de deux filtres qui coupent sur ces bandes. L'expert devra trouver ces oscillations visuellement à travers le bruit causé par les différentes couches physiques séparant le cortex et les électrodes et les artéfacts externes tels les mouvements du sujet. Pour limiter cet effet, et parce que l'activité épileptique y est à son plus fort, les données analysées se font pendant le sommeil profond. Suite à une discussion avec l'expert, nous savons que la détection visuelle de toutes les HFO sur trente minutes de données, avec 31 canaux, nécessite environ 15 heures. L'intérêt de faire une détection accélérée à l'aide de logiciels, tel que proposée par ce travail, est donc très élevé.

L'objectif de ce détecteur semi-automatique est donc de recenser le maximum de « candidats HFO » avec un minimum de faux positifs. L'expert pourra utiliser cette information pour faire sa détection de manière beaucoup plus rapide et éventuellement trouver des candidats qui lui auraient échappé en temps normal. Pour y arriver, cette recherche propose une méthodologie qui prend en considération l'information extraite d'un sous-ensemble de marqueurs de l'expert. On appelle « marqueur de l'expert » un HFO trouvé par celui-ci. Ce marqueur l'information d'une bande spectrale contient une bande de fréquences, le canal où le HFO a été observé ainsi que son début et sa fin. Il sera démontré qu'en utilisant des paramètres variables en fréquence et propres à chaque patient, la performance de détection est augmentée significativement. Les détections semi-automatiques auront également un pointage qui évaluera la probabilité qu'une détection soit un HFO réel. Ce système de pointage est conçu en considérant d'autres informations extraites des marqueurs de l'expert, telles les caractéristiques spatiales et la proximité avec les pointes épileptiques, celles-ci étant facilement trouvées par des détecteurs automatiques. Ce système permettra à l'utilisateur de contrôler son niveau de biais, et de retirer rapidement les détections les moins probables d'être de vrais positifs.

CHAPITRE 1

REVUE DE LITTÉRATURE

1.1 Oscillation hautes-fréquences

Une HFO est une activité cérébrale qui semble être un bon biomarqueur du foyer épileptogène, voire meilleur que les pointes épileptiques. En effet, les HFOs sont mesurées avant, après et pendant les crises épileptiques et principalement dans le SOZ (Andrade-Valenca *et al.* (2011)). Elles sont plus stables spatialement que les pointes et sont mesurées sur moins de capteurs. L'intérêt des HFOs est par contre très récent, aucune définition formelle n'est donc acceptée pour le moment. Néanmoins, il y a plusieurs caractéristiques observées qui sont globalement acceptées par les experts. Ces caractéristiques sont les suivantes (Melani *et al.* (2013)):

- 1- fréquence d'oscillation dans des bandes spectrales étroites (40-80 Gamma, 80-250 Ripple, 250-500 Fast Ripple);
- 2- au moins 4 oscillations;
- 3- typiquement entre 80 et 100 msec pour les HFO Gamma ou Ripple;
- 4- amplitude significative par rapport au signal EEG global.

Notons que la 4e caractéristique engendre une subjectivité dans la détection visuelle et automatique.

Afin d'éventuellement arriver à une définition formelle des HFO, il faudra mieux comprendre leur nature et leur rôle dans la pathologie. La section suivante fera la synthèse de ce que les experts y comprennent actuellement.

1.2 Exemple d'HFOs typiques

Les figures 1.1 et 1.2 présentent des deux types d'HFOs typiques marqués par un expert, les gammas et les ripples. Les HFO-gamma sont contenus dans la bande de fréquence 40-80 Hz, et les HFO-ripple sont contenus dans la bande 80-200 Hz. La figure 1.1 illustre un exemple de HFO-Gamma. Les canaux en rouge sont ceux marqués par l'expert à partir de l'échantillon 1000, identifié par une ligne verticale pointillée. La deuxième ligne pointillée identifie la durée typique d'un HFO de cette bande. Notons un artéfact cardiaque au canal 7, qui pourra causer une difficulté lors de la détection automatique.

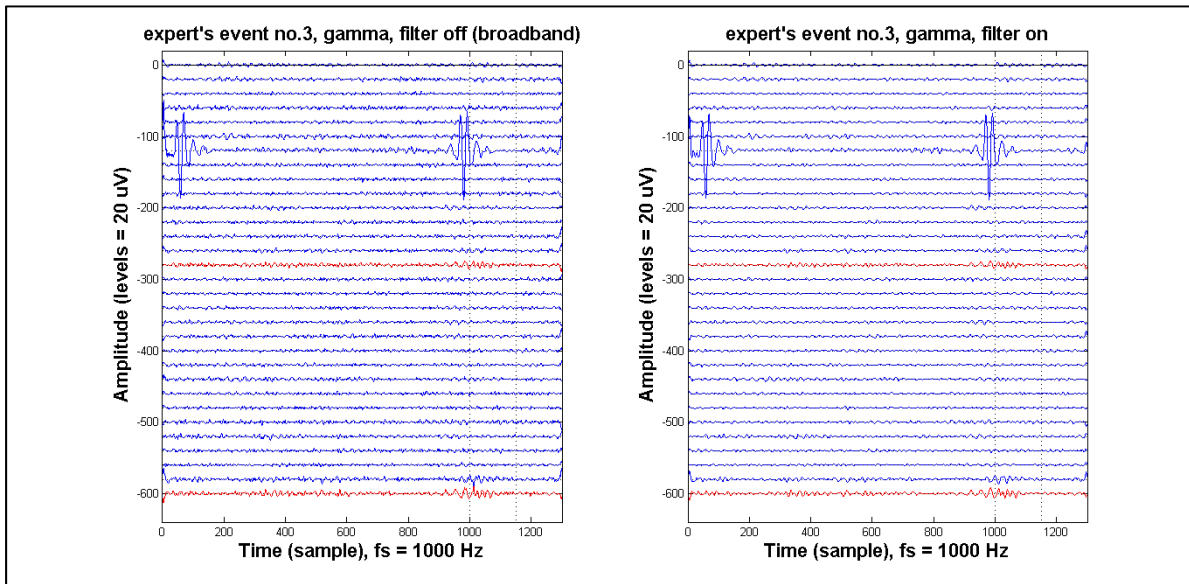


Figure 1.1 HFO-Gamma

Les canaux utilisés par l'expert pour faire la détection du HFO gamma (début à $t = 1000$) sont indiqués en rouge. À gauche : Signaux filtrés dans la bande 40-200 Hz; à droite : Signaux filtrés dans la bande 40-80 Hz

Le graphique 1.1a représente le marqueur avec un filtre large bande de 40-200Hz, tandis que la figure 1.1b applique un filtre approprié au type de marqueur, soit 40-80 Hz pour un gamma.

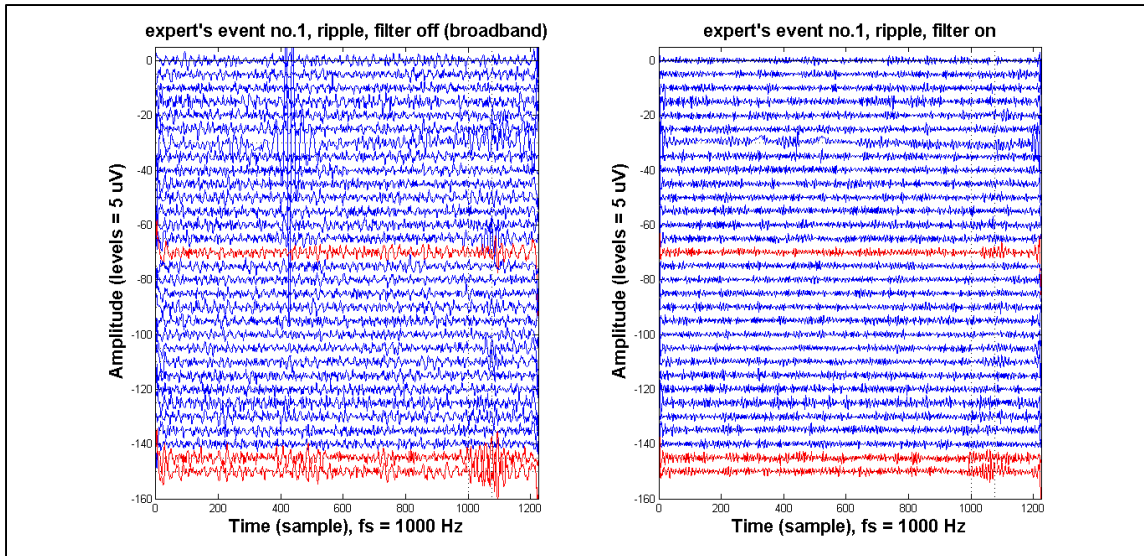


Figure 1.2 HFO-Ripple
Légende voir figure 1.1

La figure 1.2 représente un exemple typique d'HFO- Ripple, contenu dans la bande de fréquence 80-200 Hz. Encore une fois, la figure 1.2a représente le marqueur avec un filtre large 40-200 et la figure 2b avec le filtre approprié de 80-200 Hz. Notons premièrement la présence de l'artéfact cardiaque en 1.2a et sa disparition presque totale en 1.2b. Le filtre 80-200 Hz permet l'élimination de ce type d'artéfact.

Notons également que la figure 1.2b semble plus bruitée que la figure 1.1b. En effet, la détection d'HFO est plus difficile en haute fréquence qu'en basse fréquence, autant pour la détection visuelle de l'expert que pour la détection automatique.

Outre leur bande de fréquence respective, les HFO-Gamma et HFO-Ripple respectent la même définition, tel que présenté dans la section 1.1.

1.3 Intérêt sur l'hippocampe

Les prochaines sections visent à mieux comprendre le phénomène de la génération des HFOs et l'historique des recherches qui ont menés à s'y intéresser. Afin d'y parvenir, il faut d'abord expliquer le rôle de l'hippocampe dans l'activité épileptique.

L'hippocampe est une composante du cerveau dont une des fonctions principales est le contrôle du système nerveux central (Freund and Buzsaki (1996), Bernard *et al.* (2001)). On s'intéresse à l'hippocampe en épilepsie à cause de son rôle central dans le contrôle du système nerveux.

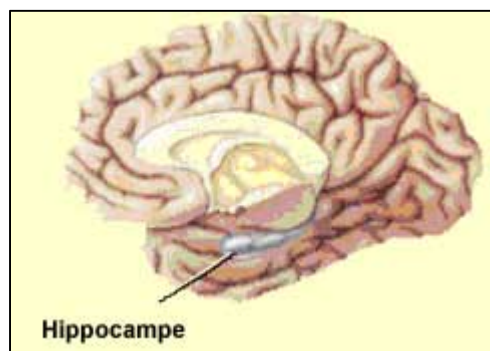


Figure 1.3 L'hippocampe
Tirée de Lecerveau.mcgill.ca (2015)

Le contrôle du système nerveux est principalement basé sur l'activité de trois types de neurones (Freund et Buzsaki (1996)), les neurones pyramidaux, les interneurones et les neurones GABAérgiques.

Les neurones pyramidaux reçoivent un grand nombre d'influx nerveux par leur grand nombre de synapses et retransmettent cette information sur une grande distance et à grande vitesse. (Bernard *et al.* (1997))

Les interneurons servent de liens entre deux types de neurones. Une fois excités, ils sécrètent un neurotransmetteur excitateur, du glutamate, qui excitera à son tour un autre neurone. (Freund et Buzsaki (1996))

Les neurones GABAergics servent d'inhibiteur par la sécrétion d'acide gamma-aminobutyric (GABA) qui empêche l'activation prolongée des neurones. Les anticonvulsants pour épileptiques traditionnels amplifient la création de GABA. (Bernard *et al.* (1997))

En 2002, à partir de mesures in vitro d'hippocampes de cerveaux épileptiques, on y découvre un comportement qui mènera vers l'intérêt pour les HFOs (Cohen *et al.* (2002)).

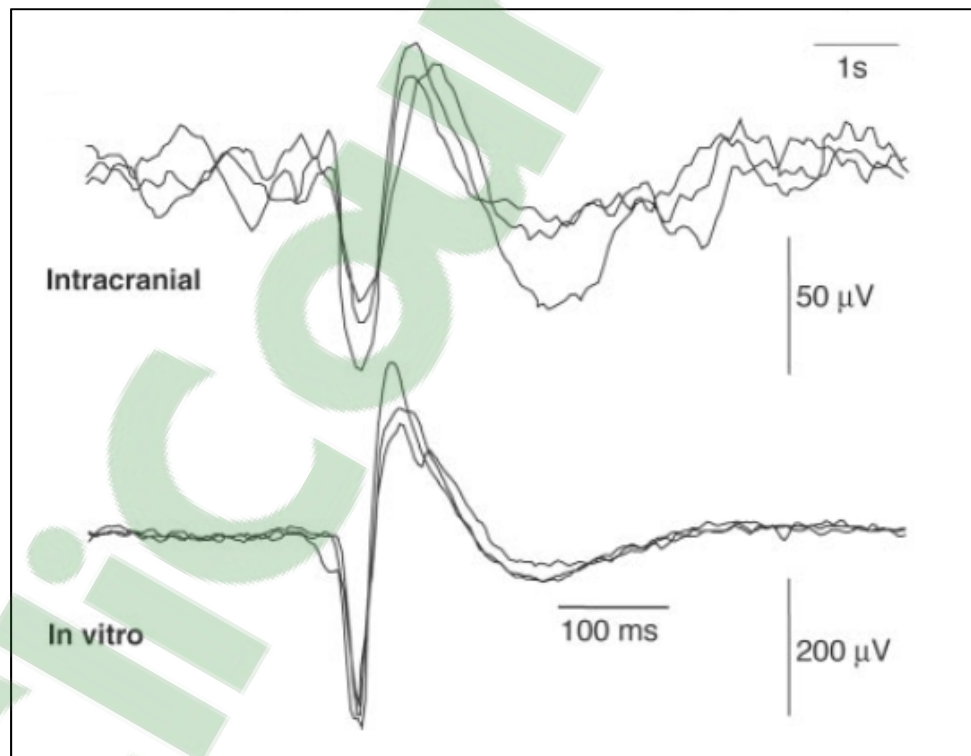


Figure 1.4 Activité in vitro (bas) vs EEG intracrânien (haut)
L'activité spontanée de l'hippocampe (in vitro) est similaire à celle mesurée sur l'EEG du patient épileptique
Tirée de Cohen et al. (2002)

Il s'avère que l'hippocampe d'un patient ayant souffert d'épilepsie sur une longue période possède des séquelles physiques apparentes (Cohen *et al.* (2002)). On tente de comprendre si ces « blessures » sont la cause des crises épileptique de ce patient ou bien si elles résultent des crises. En comparant l'activité in vitro de cet hippocampe, Cohen *et al* ont découvert une activité spontanée et brève, ce qui est anormal à l'activité régulière et stable d'un échantillon sain. De plus, cette activité est fortement corrélée avec les pointes épileptiques enregistrées chez le même patient via un EEG intracrânien. La figure 1.4 montre la grande ressemblance entre les deux activités. (Cohen *et al.* (2002)) Cette activité est enregistrée plus spécifiquement que le subiculum, une région de l'hippocampe contenant des neurones pyramidaux qui communiquent directement avec le cortex. Il est donc probable que l'activité épileptique, ou du moins une partie, seraient originaires de l'hippocampe.

Après une étude plus approfondie, on observe également que la génération de ces pointes est liée à l'interaction entre les neurones pyramidaux, les interneurones et les neurones GABAergics. Ce qui est plus intrigant pour les experts est le rôle que ces derniers semblent jouer dans la balance neuronale. Il semblerait que les neurones GABAergics, normalement inhibiteurs, jouent un rôle excitateur dans la génération de ces pointes. Ce phénomène sort complètement de la fonction primaire qui avait été observée dans ce type de neurones jusqu'à présent. (Cohen *et al.* (2002))

Cette conclusion mènera d'autres travaux portants sur les multiples facettes que pourraient avoir les neurones GABAergic et leur rôle dans l'activité épileptique.

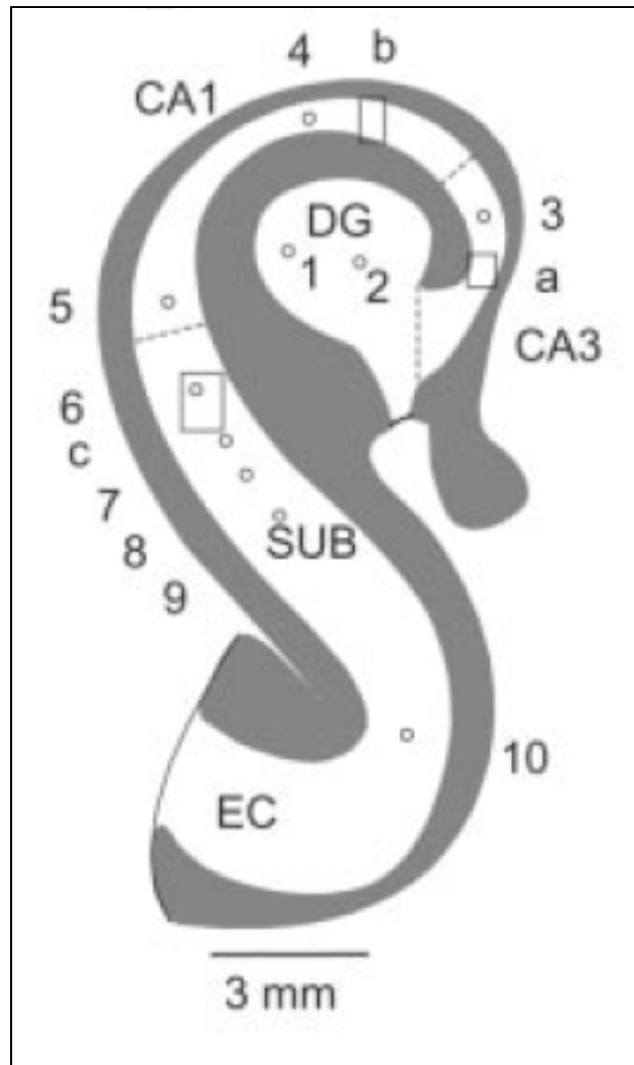


Figure 1.5 Schéma de l'hippocampe

Le subiculum (SUB) contient les neurones pyramidaux en communication avec le cortex. L'activité de la figure 1.4 est enregistrée dans cette région. CA1, CA3 et DG (Gyrus dentelé) sont d'autres régions de l'hippocampe
Tirée de Cohen et al. (2002)

1.4 Intérêt sur les neurones GABAergiques (Cossart, Bernard et Ben-Ari (2005))

En 2005 (Cossart, Bernard et Ben-Ari(2005)), suivant les découvertes présentées dans la section précédente, les experts cherchent sur ce mécanisme du changement, présumé anormal, des neurones GABAergiques vers un mode excitateur. Ils ont une caractéristique unique et fondamentale qui leur permet d'alterner de leur mode inhibiteur à un mode excitateur. Cette fonction n'est pas partagée par leurs récepteurs produisant du glutamate, ce qui peut mener à une condition pathologique.

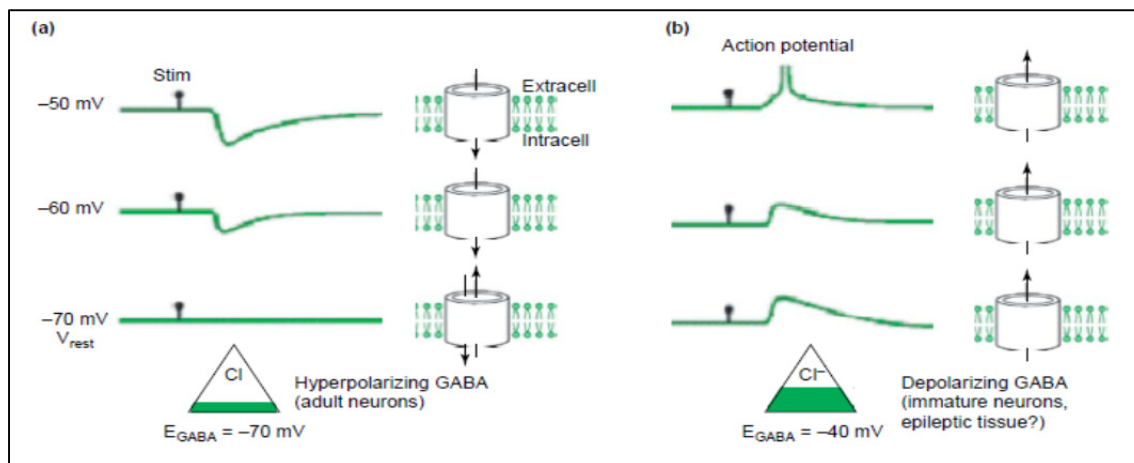


Figure 1.6 Fonctionnement normal et anormal des neurones GABAergiques
 (a) : fonctionnement normal de ces neurones avec un changement de la polarisation de la membrane. (b) fonctionnement anormal chez un patient épileptique, la polarisation est inversée et reste en mode excitateur
 Tirée de Cossart, Bernard et Ben-Ari(2005)

Comme on le voit à la figure 1.6, le fonctionnement normal (a) des neurones GABAergiques est d'absorber les ions de Cl^- jusqu'au potentiel d'action à -70mV (pour un adulte), où le flux est alors nul. Ce mode est typique d'un neurone inhibiteur. Si la concentration de Cl^- est trop importante (b) avec une polarisation autour de -40 mV dans un hippocampe avec les tissus endommagés par l'épilepsie, le flux de Cl^- est alors inversé, les neurones expulsent les ions Cl^- , rendant le neurone excitateur. Ce comportement est uniquement observé chez les sujets épileptiques, renforçant l'hypothèse du rôle de ces neurones dans l'activité épileptique.

1.5 Activité gamma des neurones GABAergiques (Bernard et al. (2012))

En 2012, l'étude sur le changement de mode des neurones GABAergiques mène à s'intéresser aux oscillations hautes fréquences, principalement les ondes Gamma (Bernard et al. (2012)). En utilisant des stimulations électrique et chimique sur des hippocampes in vitro chez le rat immature, il est possible de stimuler un événement très semblable à une crise épileptique appelé un *ictal-like event* (ILE). On y observe que des GFO (gamma-frequency oscillations) sont toujours présents dans plusieurs régions de l'hippocampe aux instants précédents les IEL (pre-ictal).

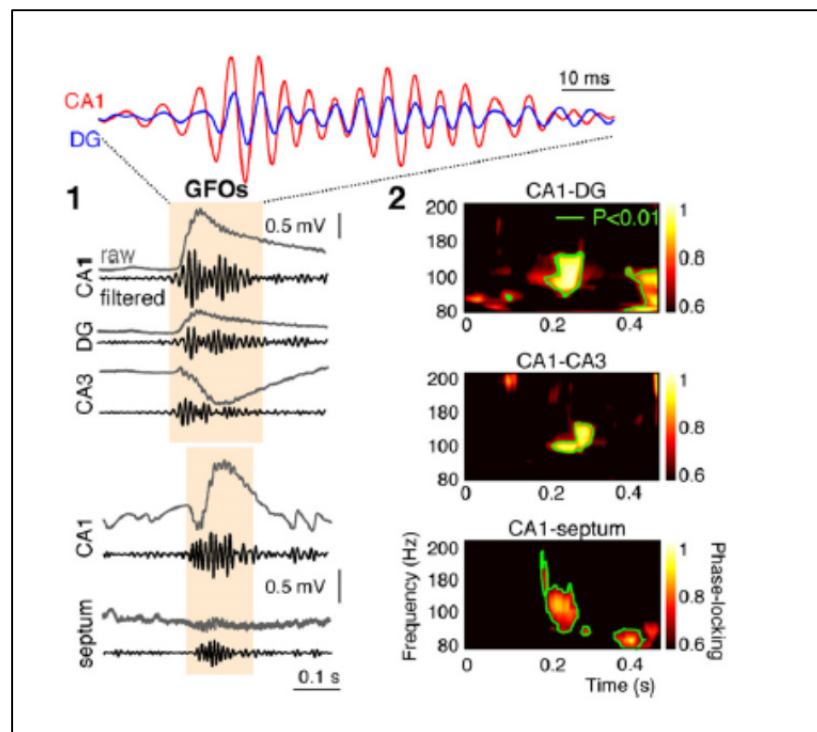


Figure 1.7 Oscillation gamma (GFO) pre-ictale dans l'hippocampe
 (1) Synchronie entre les régions CA1 et DG de l'hippocampe. (2)
 Augmentation de la synchronie entre les régions à 100 Hz observée
 sur la transformée de Fourier
 Tirée de Bernard et al. (2012)

Avant l'activité gamma, aucun neurone GABA n'était actif sauf un seul type, les neurones hippocamposeptal (HS) qui ont eu une activation soudaine. L'activité de HS est en phase avec la GFO observée et survient toujours *avant* les GFO d'environ 10-300 ms. La séquence d'activation des neurones est présentée à la figure 1.8.

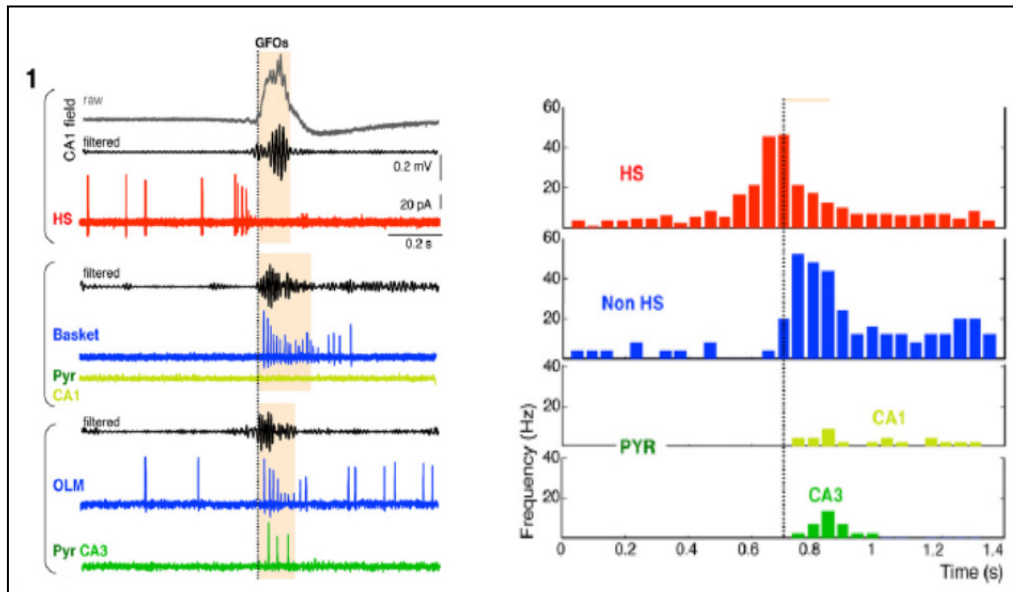


Figure 1.8 Séquence d'activation des neurones durant une GFO
 Dans les deux colonnes, on montre des neurones HS (rouge), des interneurons (bleu) et des neurones pyramidaux (vert et jaune). On note (à droite) une séquence d'activation entre ces quatre types de neurones
 Tirée de Bernard et al (2012)

On y voit clairement que l'activité des neurones HS devient spontanée juste avant un HFO, une activité de hautes fréquences et très brève. Lorsque cette activité est suffisamment forte, les interneurons sont activés, et ils activent à leur tour les cellules pyramidales dans CA1 et CA3. Puisque ces dernières sont stimulées, cela pourrait expliquer la présence des oscillations hautes fréquences projetées sur le cortex puisque les neurones pyramidaux y sont liés.

Cette découverte suggère que les cellules HS ont un rôle central dans la génération des HFOs. De plus, l'ablation ou la suppression des neurones GABA n'empêche pas les ILE,

mais les GFOs observés dans l'hippocampe sont abolis. Inversement, l'ablation ou la suppression des interneurons ne supprime ni les ILE, ni les GFO. Les HFOs sont donc probablement un effet de l'épilepsie et non une cause.

La prochaine section présente une méthodologie qui vise à rendre la détection visuelle des HFOs plus rigoureuse.

1.6 Détection automatique des HFOs (von Ellenrieder et al. (2011))

Un HFO n'a pas de définition bien établie. Les différentes recherches sur ce sujet n'ont pas atteint de consensus, mais les critères suivants seront ceux utilisés dans cette recherche pour la conception du détecteur; (von Ellenrieder *et al.* (2011), Gotman *et al.* (2006))

- 1- le spectre de l'évènement doit être contenu entre 40 et 200 Hz;
- 2- il doit avoir au moins 4 oscillations stables;
- 3- il doit être d'amplitude significative par rapport à l'activité de fond.

On définit donc la bande spectrale 40-200 Hz comme la bande large et des bandes de 10 Hz contenues dans la bande large comme les bandes étroites. La figure ci-dessous montre des exemples d'HFO, représentés dans le signal global, dans le signal filtré dans la bande large et finalement dans les bandes étroites. Comme on peut le constater, la détection visuelle dans la bande large nécessite une grande subjectivité due à l'absence de consensus, certains HFO étant peu distinctifs du bruit qui l'entoure.

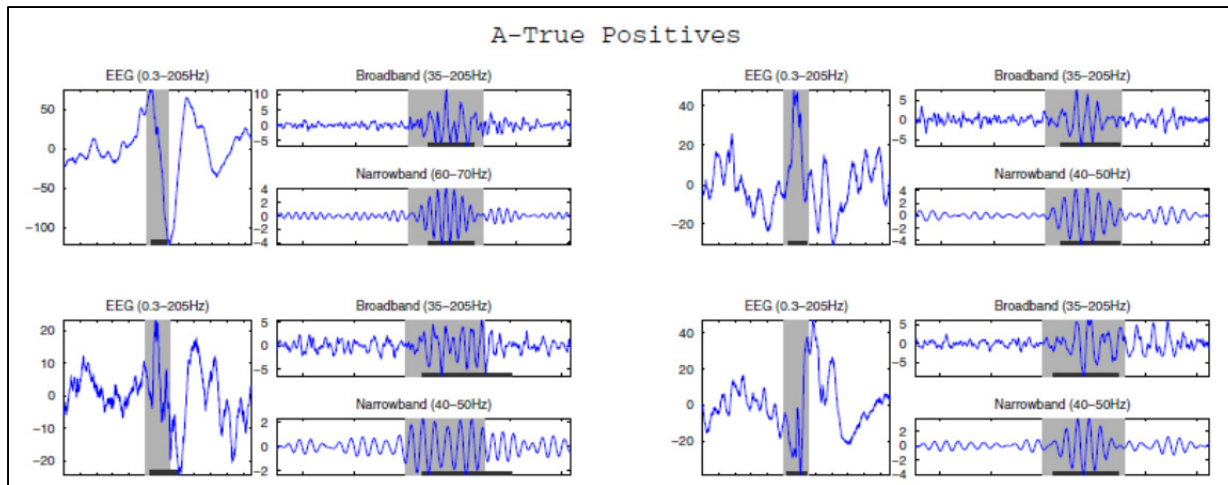


Figure 1.9 Quatre exemples de vrais HFO gamma filtrés

L'oscillation devient plus identifiable avec le filtre le plus étroit. La région ombragée est la fenêtre contenant le HFO

Tirée de von Ellenrieder et al. (2011)

C'est avec cette définition que l'algorithme de Nicolas Ellenrieder (von Ellenrieder et al. (2011)) fut conçu. Cet algorithme, qui servira de base pour ce projet, est présenté dans la section suivante.

La performance du détecteur sera évaluée en fonction de sa sensibilité et de sa spécificité. À partir de données marquées par un expert, une performance idéale sera de détecter tous les marqueurs identifiés sans faux-positifs, en ne considérant que l'instant de la détection. Donc si un évènement fut détecté au même moment, mais sur plusieurs fréquences ou sur plusieurs canaux, ils seront tous traités comme une seule détection. La sensibilité peut être calculée par un pourcentage des marqueurs de l'expert détectés avec succès, tandis que la spécificité sera présentée sous forme de faux positifs par minute, noté FPR.

La méthodologie pour calculer la sensibilité du détecteur consiste à vérifier si chaque marqueur de l'expert fut trouvé par le détecteur, c'est à dire avec au moins un canal détecté sur la même fenêtre temporelle que le marqueur. Notons qu'une détection faite sur un canal différent que le marqueur, mais dans la même fenêtre temporelle est considérée comme un vrai positif. De manière itérative chaque marqueur gamma est vérifié, et ensuite les marqueurs ripple. La sensibilité est définie comme la somme de marqueurs trouvés sur le nombre de marqueurs total. Le nombre total de marqueurs est le nombre de marqueurs au cours du temps, les détections simultanées sur plusieurs canaux pendant au moins un échantillon ne comptant que pour un seul marqueur. Pour le taux de faux positifs la méthodologie est similaire. Cette fois de manière itérative on vérifie si chaque détection arrive au même moment qu'un marqueur soit gamma ou ripple. Le nombre total de détections qui n'ont pas de marqueur au même moment sur la durée totale de l'enregistrement donne le taux de faux positifs.

Les critères de performance arbitrairement choisis pour ce projet pour considérer une détection comme satisfaisante sont d'obtenir une sensibilité d'au moins 75% et un FPR d'au plus 15. Une sensibilité est souhaitée pour assurer de trouver

1.7 Détecteur initial (von Ellenrieder et al. (2011))

La détection consiste à isoler tous les événements potentiels de HFO dans un enregistrement EEG de scalp, avec un minimum de faux positifs. La stratégie proposée est de décomposer le signal en bande de fréquences de 10 Hz, qui aura pour principal effet de limiter le bruit de fond du signal (background), en plus d'assurer que la fréquence d'oscillation sera contenue dans une même bande de fréquences. Cela évitera les fausses détections à large bande, comme une pointe épileptique filtrée. Utiliser des bandes étroites de 10 Hz permettra au détecteur automatique d'ignorer cet effet. La séquence de détection est la suivante: chaque canal est traité séparément et indépendamment des autres. Chaque canal est filtré dans 16 bandes de fréquences allant de 40 à 200 Hz. Dans la figure 1.10, nous pouvons apercevoir qu'un HFO présent à certaines fréquences (40-50 Hz) ne le sera pas dans les autres bandes (140-150 Hz).

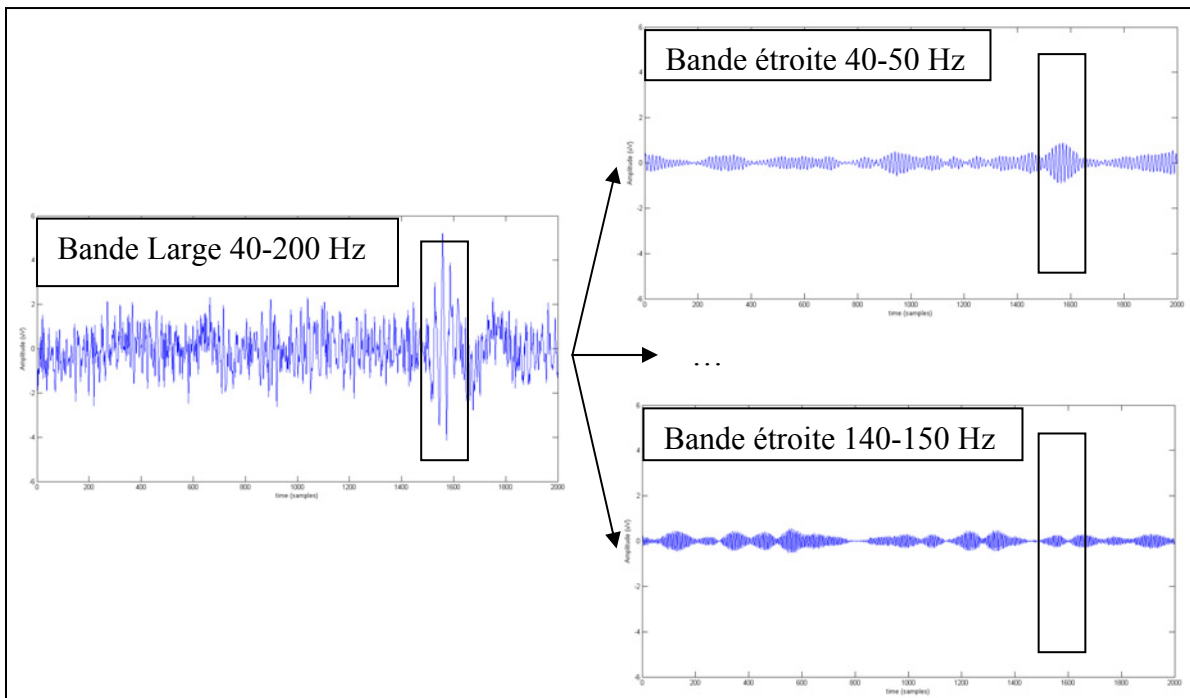


Figure 1.10 Décomposition d'un HFO (dans l'encadré) en bandes étroites
On détecte le HFO dans une des bandes spectrales, ici dans la bande 40-50 Hz

Les filtres utilisés sont des FIR de Butterworth d'ordre 300. Une fois le signal d'un canal décomposé, il calcule deux valeurs d'énergie (RMS) pour chaque échantillon t , sur chaque bande étroite. La valeur $RMS(t)$ contiendra la valeur RMS de la fenêtre T_d centrée sur l'instant t , de largeur de 4 oscillations à la fréquence centrale de la narrowband (F_c). La valeur $RMS_b(t)$ contiendra la valeur RMS d'une fenêtre T_b de durée fixe choisie par l'utilisateur (entre 1 et 30 secondes). Cette fenêtre est située derrière la fenêtre RMS, avec 100 msec de séparation T_s entre les deux pour éviter le chevauchement d'information.

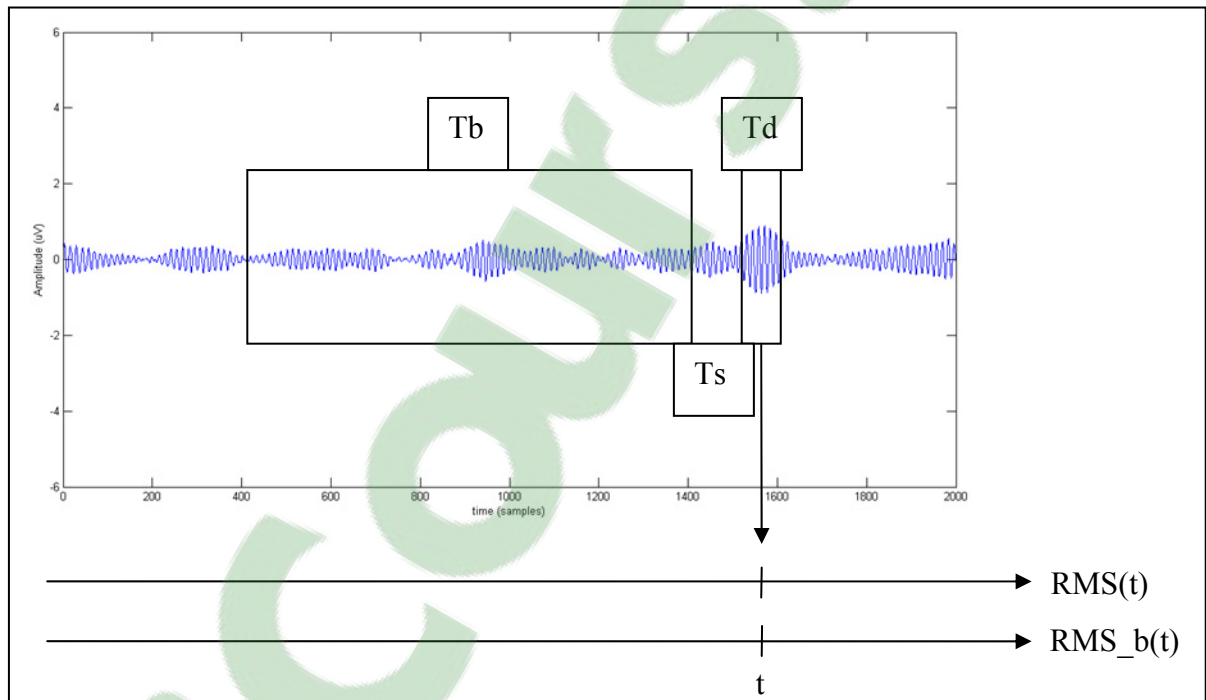


Figure 1.11 Méthodologie de détection

$RMS_b(t)$ est l'énergie contenue dans la fenêtre T_b . Cette fenêtre précède (délai T_s) la fenêtre T_d centrée sur l'instant t dans laquelle on calcule l'énergie $RMS(t)$

Si $RMS(t)$ est supérieur à $RMS_b(t)$ multiplié par un seuil de détection TH , l'instant t est considéré comme un instant candidat HFO. Il sera donc noté que le vecteur de détection $d_f(t)$ est égal à 1, ou 0 si la condition précédente n'est pas remplie. Une fois la bande étroite complètement détectée, une séquence de 1 dans $d_f(t)$ sera considérée comme une détection, avec un temps de commencement et de fin. Toutes les détections survenant au même instant,

indépendamment de la fréquence et du canal, se feront assigner une étiquette commune, les définissant comme un *subject level event* (SLE). Au moment de l'évaluation de la performance, un SLE sera considéré comme un seul événement.

Donc, le seuil de détection TH est une valeur arbitraire qui déterminera si l'augmentation d'énergie sur une bande de fréquences est suffisante pour considérer l'oscillation comme significative par rapport à l'activité de fond (von Ellenrieder et al. (2011)). La valeur suggérée par défaut est de 2.5, et ne varie pas temporellement, spectralement et spatialement. Cette méthode est très sensible avec généralement 90% des marqueurs de l'expert détectés, mais le taux de faux positifs par minute (FPR) risque d'être au-delà de 100. L'étape suivante sera donc une classification qui visera à séparer les vrais positifs des faux positifs.

1.8 Définition des seuils (von Ellenrieder et al. (2011))

Il existe deux types de fausses détections. Le premier provient des artéfacts musculaires. Au cours d'un mouvement, même pendant le sommeil, l'activité électrique des fibres musculaires sur le scalp près des électrodes produit cet artéfact à large bande spectrale. Il sera donc probable de voir une augmentation d'énergie significative sur plusieurs bandes étroites à cet instant. Le deuxième type de fausses-détections est dû aux oscillations de très basse amplitude qui sont trop faible pour être considéré comme une HFO. En se fiant aux experts, il y a un critère d'amplitude sur le signal filtré, si l'amplitude est trop basse il ne peut pas s'agir d'un HFO. Pour éliminer un maximum de ces fausses détections, trois valeurs seront calculées pour chaque détection sur la période définie;

- A- RMS(t) maximal sur la séquence de détection;
- B- amplitude absolue maximale de la broadband;
- C- amplitude absolue maximale de la narrowband.

Deux nouveaux seuils sont définis : THA et THB. Une détection sera retirée si elle vérifie la condition suivante :

$$A < \text{THA} \text{ ou } B > \text{THB} \times C$$

THA sert donc à retirer simplement les détections dont l'énergie est trop faible. THB tente de retirer les artéfacts musculaires. Dans le cas d'un déplacement d'électrode, l'énergie sur chaque bande étroite sera suffisante pour activer une détection, mais lorsque comparé à l'énergie de la bande large, le ratio d'amplitude sera élevé.

Tableau 1.1 Résultat du détecteur initial publié

THA et THB représente les seuils de classification utilisés pour chaque patient. N est le nombre de détection de l'expert ou du détecteur automatique. S représente la sensibilité et FPR est le taux de faux positif par minute. PP (positive predicted) est la proportion de vrais positifs des détections automatiques, et PA (positive agreement) représente l'accord entre l'expert et le détecteur

Tirée de von Ellenrieder et al. (2011)

Patient	TH A	TH B	N_{expert}	N_{auto}	S (%)	FPR	PP (%)	PA (%)
S86			1284	1755	85.6	1.6	57.3	69.2
S95			1284	2794	94.9	3.6	39.7	57.0
A	2.97	1.34	12	69	83.3	2.0	14.5	24.7
B	3.03	1.35	12	162	100	5.0	7.4	13.8
C	3.01	1.34	37	82	97.3	1.5	43.9	60.5
D	3.03	1.34	41	175	100	4.2	23.4	38.0
E	3.38	1.40	153	212	93.5	2.4	65.6	77.3
F	3.38	1.44	74	136	85.1	2.4	46.3	60.0
G	3.03	1.34	55	203	98	4.2	27.1	42.2
H	3.38	1.34	169	390	99	7.4	43.3	60.3
I	2.85	1.34	241	303	92.5	4.0	60.7	74.8
J	2.97	1.34	3	170	33.3	5.0	0.6	1.2
K	3.01	1.35	42	101	90.5	2.1	36.6	52.4
L	3.03	1.34	26	129	100	3.5	20.2	33.5
M	3.13	1.34	212	288	96.7	4.5	47.9	68.6
N	3.13	1.34	84	226	98.8	4.3	35.4	52.6
O	2.97	1.34	123	148	93.5	1.0	79.1	85.6

Les seuils THA et THB sont fixes indépendamment du temps, de la fréquence ou du canal de détection. Les données utilisées ne sont pas disponibles pour ce travail. Il est néanmoins possible de tester les résultats sur d'autres données marquées par un expert de la même institution en utilisant la même méthodologie. Notons que les nouvelles données ont été échantillonnées sur un système EEG Harmonie-Stellate en utilisant le même montage, mais avec une fréquence d'échantillonnage plus élevée (1000 Hz au lieu de 600 Hz), qui devrait être sans conséquence étant donné la fréquence d'intérêt inférieure à 200 Hz.

Notons que deux valeurs calculées dans le tableau 1.1, PP et PA, ne sont pas calculées pour le reste de ce travail. PP est la prédiction positive du détecteur, et est défini par le ratio des vrais positifs sur le nombre total de détections. PA est un calcul d'accord entre l'expert et le détecteur qui prend en considération qu'un marqueur de l'expert pourrait être trouvé plusieurs fois par le détecteur. Comme nous définissons toutes les détections superposées temporellement comme une seule détection (SLE) ce calcul n'est d'aucun intérêt pour nous.

Tableau 1.2 Résultats détecteur initial sur nouveaux patients

Détection	Classification
Sens = 89%	Sens = 60%
FPR = 116/min	FPR = 18/min

Les performances sur nos données sont inférieures à celles au tableau 1.1. Les seuils utilisés furent les valeurs moyennes sur les 8 patients testés, soit 2,67 pour THA et 1,35 pour THB. Comme le montre le tableau 1.2, la sensibilité diminue à 60% et le FPR passe à 18. Notons que la valeur des seuils pour obtenir les performances maximales varie significativement d'un patient à l'autre. Il serait donc préférable de fixer les seuils de manière à considérer l'activité électrophysiologie de chaque patient.

Considérant la nature variable des HFO d'un patient à l'autre, utiliser un détecteur automatique n'est peut-être pas la solution idéale. Ce travail propose donc un détecteur semi-automatique qui utilisera un échantillon des marqueurs de l'expert pour y extraire les caractéristiques d'amplitude des HFOs trouvés, et optimiser les seuils en conséquence. Ceci est très envisageable en pratique, où l'expert n'aura qu'à détecter visuellement une fraction de l'enregistrement pour permettre au détecteur de trouver les événements de même caractéristiques. Comme un enregistrement de 30 minutes sur 31 canaux prend approximativement 15 heures à l'expert, le gain potentiel en temps est considérable.

1.9 Conclusion

La revue des résultats avec un expert en détection des HFOs a permis de faire deux observations qui vont s'avérer au centre des changements proposés pour augmenter la performance du détecteur;

- 1- les seuils fixés par défaut peuvent être optimisés empiriquement. Ces valeurs varient d'un patient à l'autre;
- 2- l'amplitude des oscillations dans la bande ripple est plus basse que dans la bande gamma, mais leur rapport signal sur bruit est plus élevé.

La première observation nous dit que les paramètres ne devraient pas être fixes d'un patient à l'autre, tandis que la deuxième suggère qu'ils ne devraient pas être fixes en fréquences. Les changements proposés sont donc de faire un système d'apprentissage qui permettra au détecteur d'extraire les informations spectrales d'un sous-ensemble de marqueurs d'un patient en particulier, et de fixer les seuils de détection et de classification en conséquence. Avec un sous-ensemble de marqueurs gamma de marqueurs ripple, nous sommes en mesure d'affirmer que nous pouvons obtenir des seuils offrant des performances supérieures à ceux proposés par le détecteur initial. Ces résultats seront présentés dans le chapitre 5.

On doit maintenant s'interroger sur la quantité de marqueurs nécessaire pour bien représenter l'ensemble des HFOs. Dans cette perspective la stabilité temporelle, spectrale et spatiale des marqueurs est étudiée au chapitre 3. Le chapitre 2 présente d'abord la méthodologie de détection visuelle qui est utilisée dans le marquage des données de ce travail.

CHAPITRE 2

LA DÉTECTION VISUELLE

2.1 Introduction

Ce chapitre a pour objectif d'exposer la méthodologie dans la détection visuelle des HFOs. Cette méthode est celle utilisée par l'expert qui a fourni les données utilisées dans ce projet.

2.2 Méthodologie

L'intérêt sur les oscillations hautes-fréquences est relativement récent (Gotman *et al.* (2006)), l'hypothèse est que ces événements seront plus présents dans la zone identifiée comme le foyer épileptogène d'un patient que n'importe où ailleurs. Les méthodologies de détection visuelle sont encore rares pour les données d'EEG intracrânien et encore plus pour les données EEG de scalp. Il est clair que le bruit induit par les différentes couches physiques entre le cortex et le scalp constitue une grande difficulté pour un expert qui désire répertorier ces événements. La méthode proposée est celle utilisée par l'expert qui a fourni les marqueurs de ce projet. (F. Mari *et al.* (2011))

La méthode consiste à détecter deux types d'oscillations; un premier étiqueté «gamma» pour les événements d'au moins 4 oscillations contenues entre 40 et 80 Hz et le type «ripple» pour ceux oscillant au-delà de 80 Hz.

La détection est réalisée sur 30 minutes consécutives de sommeil profond (Mari *et al.* (2011), Melani *et al.* (2013)). Dans cette période les événements épileptiques (pointes et HFO) sont plus communs et les artefacts de mouvements le sont moins. Les pointes épileptiques sont également détectées sur la même période visuellement, mais ignorées lors de la détection des HFOs pour éviter d'en biaiser la détection. Un HFO sera défini

comme une oscillation d'au moins 4 périodes et d'amplitude significativement plus grande que l'activité de fond. L'affichage de tous les canaux superposés est fixé à 250mm/sec. L'écran est ensuite séparé en deux pour afficher simultanément le signal avec un filtre passe-bande 40-80 Hz et un passe-haut à 80Hz. Un évènement sera marqué gamma si il est présent dans la bande gamma et absent dans la bande ripple, et vice-versa pour les marqueurs ripple. Le processus est réalisé une deuxième fois par le même expert pour faire office de vérification.

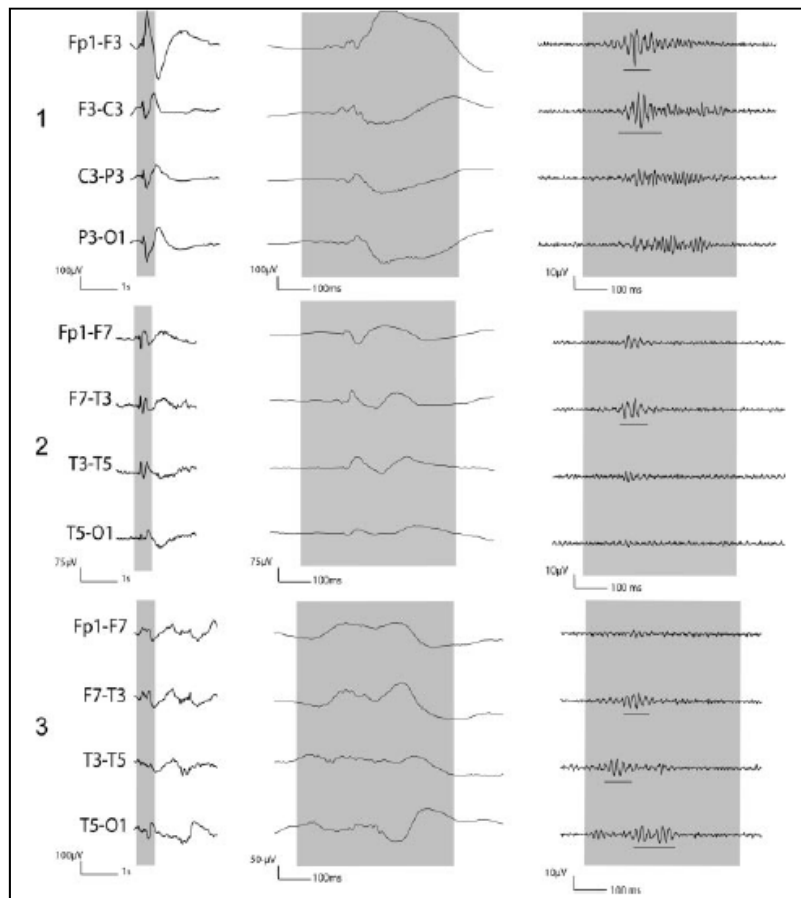


Figure 2.1 Exemple de HFO-Gamma

(1) HFO gamma visible simultanément à une pointe (deuxième colonne). (2) HFO-Gamma non visible simultanément à une pointe, mais visible après filtrage (troisième colonne). (3) HFO-Gamma indépendante d'une pointe

Tirée de Mari et al. (2011)

La figure 2.1 illustre trois exemples de HFO détectés par l'expert. La première colonne montre les données brutes, la seconde est la vision agrandie de la zone grise et la troisième est le filtrage passe-bande dans 40-80 Hz. Les HFO sont soulignés dans la troisième colonne. On constate que l'activité cérébrale est très faible dans cette bande de fréquence. Il est donc important que l'oscillation soit significative en amplitude par rapport à l'activité de fond pour considérer une détection.

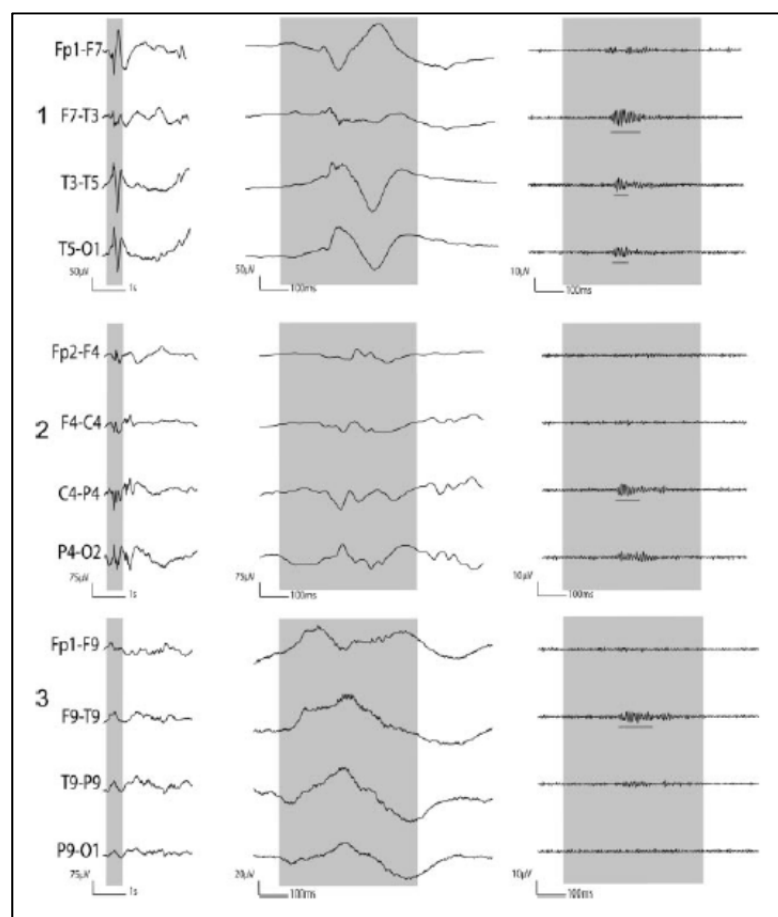


Figure 2.2: Exemple de HFO-Ripple
Légende voir figure 2.1
Tirée de Mari et al. (2011)

Dans la figure 2.2, on considère maintenant trois cas de HFO détectés dans la gamme ripple, donc dans la bande 80-200 Hz. On peut constater que l'activité de fond est plus

petite que dans les HFOs gamma, cela est dû au fait que l'activité cérébrale dans cette bande spectrale est plus rare. L'amplitude des HFOs ripple est toutefois environ deux fois plus petite. Cette caractéristique sera discutée dans le chapitre 4 sur l'exploitation de cette information dans la détection automatique.

Une fois les HFOs détectées, on s'intéresse au taux de chaque évènement par minute, si l'évènement est situé à l'intérieur du foyer épileptique (SOZ) ou non et si la détection du HFO est simultanée à une pointe épileptique. Cette zone est connue par l'expert dans le cadre de cette recherche.

Sensibilité = $[\text{canaux SOZ avec HFO} / (\text{canaux SOZ avec HFO} + \text{canaux SOZ sans HFO})] \times 100$

Spécificité = $[\text{canaux hors SOZ sans HFO} / (\text{canaux hors SOZ sans HFO} + \text{canaux hors SOZ avec HFO})] \times 100$

Précision = $[(\text{canaux SOZ avec HFO} + \text{canaux hors SOZ sans HFO}) / \text{nombre total de canaux}]$

Le risque de confondre un HFO avec un artéfact de mouvement est élevé puisque leur bande spectrale se superpose. Les évènements à morphologie variable ou d'amplitude très élevée sont donc vérifiés à 30mm/sec et retirés au besoin pour minimiser ces fausses détections.

Il y a également le risque de confondre une pointe épileptique filtrée avec un HFO (Bénar *et al.* (2009)), voir figure 2.3. En effet, ces évènements ne sont pas restreints dans une bande de fréquence étroite et leur énergie peut être distribuée jusque dans les bandes d'intérêt des HFOs en particulier. Les pointes épileptiques par exemple possèdent une oscillation centrale très rapide. Cela a pour effet de causer de fausses détections de HFOs puisqu'elles ont les mêmes caractéristiques temporelles, spectrales et d'amplitude définie pour la détection visuelle.

Pour éviter ce problème, il est conseillé de commencer par la détection de pointes par une méthode visuelle habituelle ou un détecteur automatique. La détection des HFOs devrait ensuite être faite en connaissant préalablement les instants contenant une pointe.

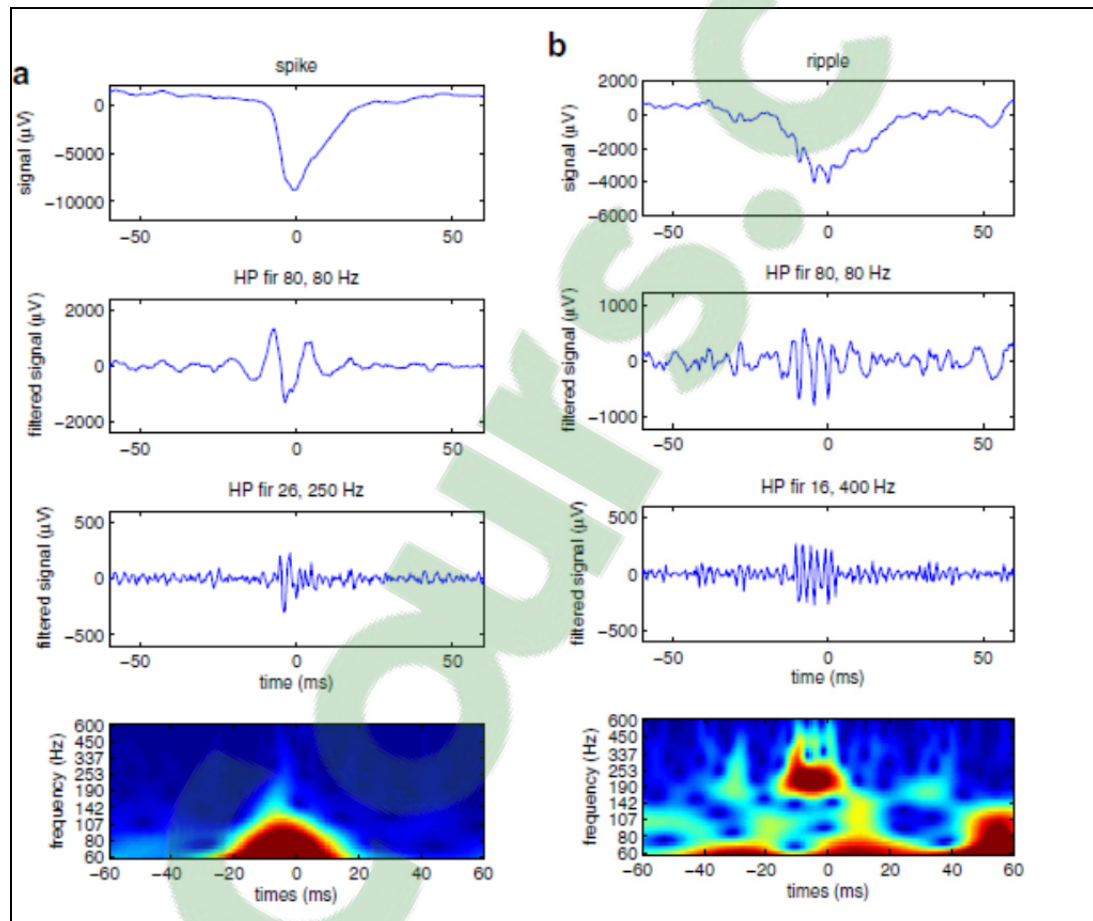


Figure 2.3 Pointe filtrée vs HFO filtré
 (a) Effet d'un filtre passe-bande sur une pointe
 (b) Effet du même filtre sur un HFO ripple
 Tirée de Bénar et al. (2009)

2.3 Exemple de marqueurs

Voici des exemples de vrais HFOs qui sont à la fois trouvés par l'expert et par le détecteur automatique dans sa forme actuelle. Notons que l'augmentation d'énergie à partir de la première ligne pointillée (début du marqueur) est visible.

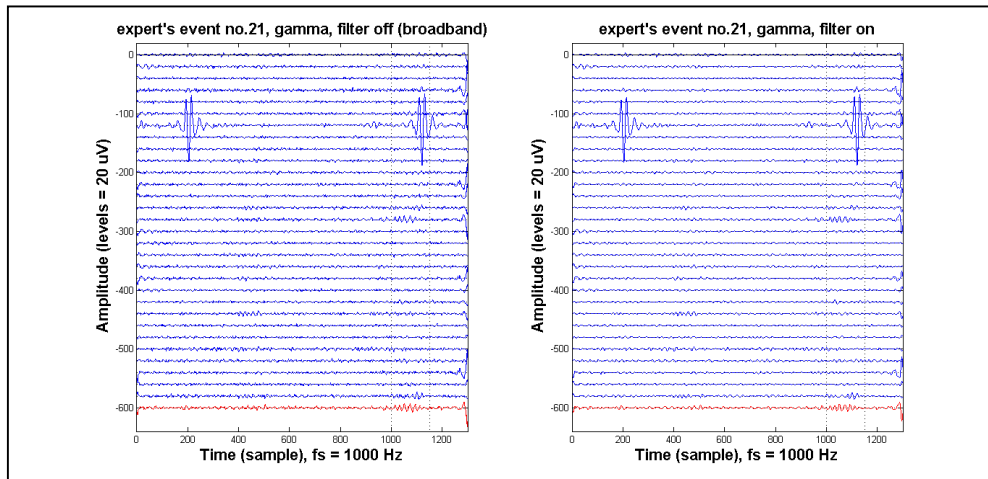


Figure 2.4 Vrai HFO-Gamma

Les canaux où l'expert a fait la détection du HFO gamma (début t=1000) sont indiqués en rouge. À gauche : Signaux filtrés dans la bande 40 à 200 Hz; à droite : Signaux filtrés dans la bande 40 à 80 Hz

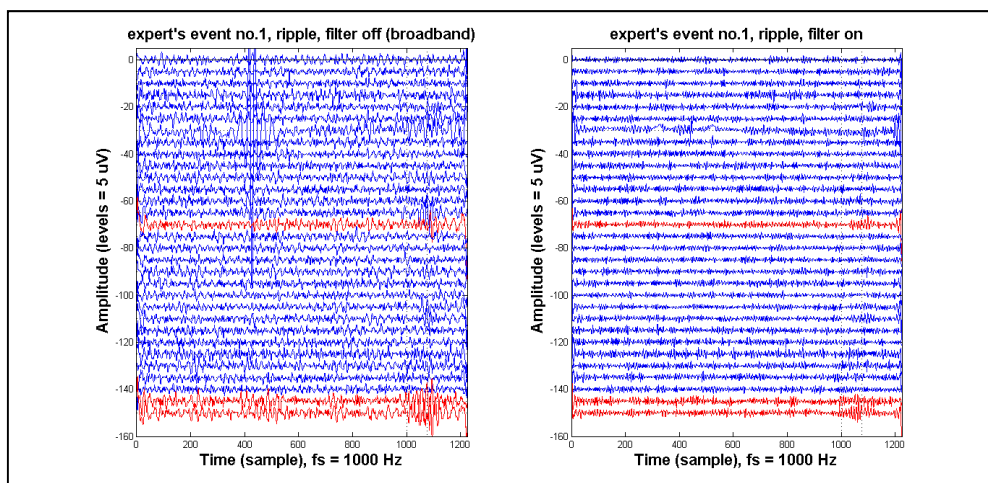


Figure 2.5 Vrai HFO-Ripple
Légende voir figure 2.4

Les figures suivantes montrent des HFOs trouvés par l'expert mais qui ne sont pas trouvés par le détecteur. Dans les deux cas, l'énergie dans la fenêtre de détection est visible, mais très faible. On peut également distinguer du bruit dans l'activité de fond, qui contribue à augmenter l'énergie dans l'activité de fond et rend la détection plus difficile pour le détecteur.

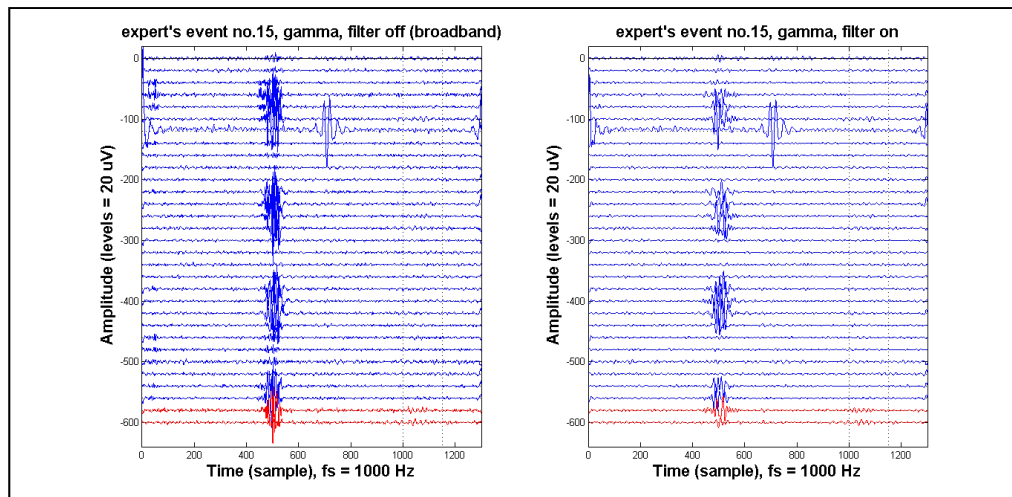


Figure 2.6 Faux négatif (gamma)
Légende voir figure 2.4

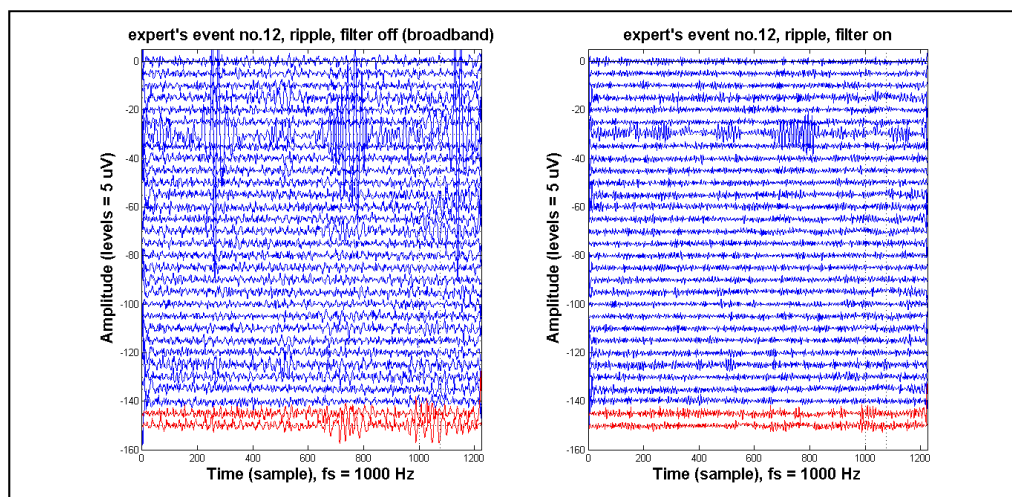


Figure 2.7 Faux négatif (ripple)
Légende voir figure 2.4

Ces marqueurs de l'expert ne semblent pas répondre aux critères établis dans notre définition d'un HFO. Ils n'ont pas été trouvés par le détecteur, mais après analyse, il est probable que la subjectivité de la détection visuelle explique ce type de marqueur par l'expert. Il nous est impossible de savoir pourquoi ces oscillations en particulier furent marquées, mais chacun de ces marqueurs aura un impact sur le calcul de sensibilité de notre détecteur, puisque les chances que le détecteur automatique les détecte sont très faibles.

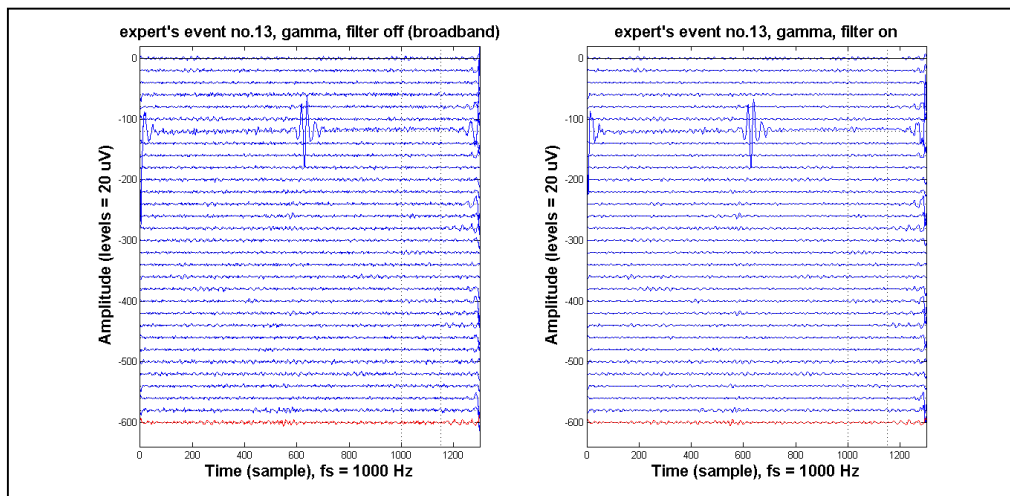


Figure 2.8 Exemple faux négatif (gamma) dont on peut douter de l'existence réelle
Légende voir figure 2.4

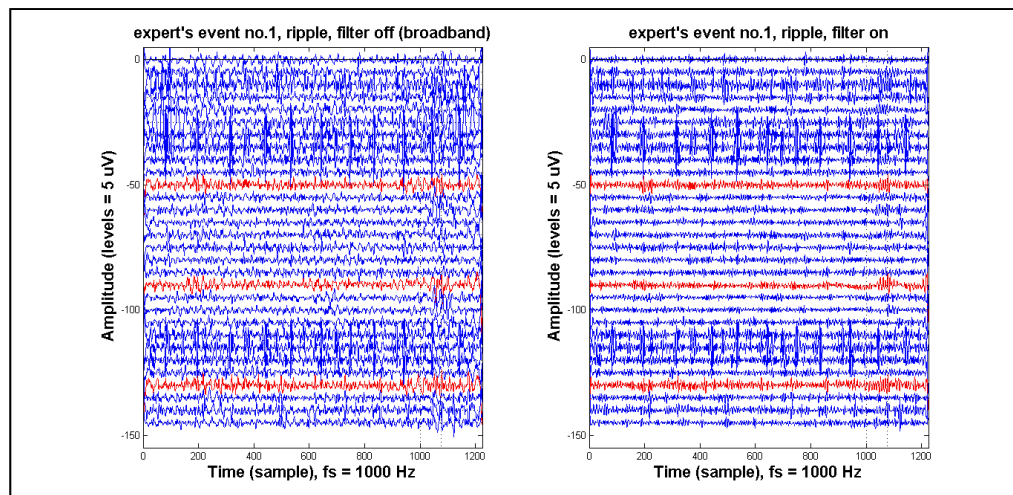


Figure 2.9 Exemple faux négatif (ripple) dont on peut douter de l'existence réelle
Légende voir figure 2.4

CHAPITRE 3

LE DÉTECTEUR SUPERVISÉ

3.1 Objectif

Une détection supervisée est une détection semi-automatique qui utilise quelques marqueurs de l'expert pour apprendre un modèle par optimisation. Afin d'assurer le bon fonctionnement du détecteur supervisé, un sous-ensemble de marqueurs doit être représentatif de l'ensemble des marqueurs d'un patient. Comme la sensibilité du détecteur et sa capacité à séparer les vrais positifs des fausses détections dépendent de paramètres optimisés à partir d'un sous-ensemble de marqueurs de l'expert, le nombre de marqueurs utilisés est donc un paramètre important. On peut se convaincre qu'un sous-ensemble plus grand de marqueurs permettra une détection plus performante; toutefois, l'intérêt de ce détecteur est de limiter l'effort requis de l'expert. On s'intéresse donc ici à la quantité de marqueurs juste nécessaire pour obtenir une représentation adéquate de l'ensemble des HFOs d'un sujet.

La première contribution de ce travail sera l'analyse des marqueurs de l'expert. Ce chapitre présentera l'impact du nombre de marqueurs sur le calcul des seuils de détection et de classification, qui sont basés sur des caractéristiques d'amplitude et spectrales, ainsi que sur des propriétés spatiales. Il sera également vérifié s'il est préférable de prendre les marqueurs sur une fenêtre temporelle donnée ou prendre un nombre fixe de marqueurs sélectionnés aléatoirement sur l'ensemble des données.

3.2 La base de données

Pour ce travail, nous avons accès à une base de données contenant les EEG de patients qui ont préalablement été marqués par un expert selon la méthodologie présentée au chapitre 2. Chaque patient fut marqué pendant une phase de sommeil profond pour limiter les artéfacts musculaires, sur une période d'environ 30 minutes consécutives. Nous avons omis les patients qui n'avaient pas ou très peu de marqueurs HFO sur l'ensemble de l'acquisition. Nous avons également retiré les canaux qui furent identifiés comme trop bruité par l'expert.

Tableau 3.1 Base de données des patients utilisés dans ce travail.
Les lignes sont le nombre de marqueurs gamma et ripple de chaque patient (colonne). La dernière colonne, MOY, représente la moyenne sur tous les patients

	CML	DR	JC	MA	PH	PJ	SBL	SG	MOY
Marqueurs Gamma	27	9	43	72	198	9	207	135	88
Marqueurs Ripple	7	6	5	25	47	6	77	33	26

3.3 Variabilité temporelle

L'objectif de cette étape est de vérifier que le taux de marqueurs ne varie pas dans le temps sur l'ensemble de l'acquisition (environ 30 minutes). Dans le cas contraire, la décision de prendre un nombre fixe de marqueurs sur toute la durée ou de prendre tous les marqueurs sur une période fixe pourrait être affectée. Il serait alors préférable de valider une stabilité puisqu'il est plus pratique de demander à un expert de simplement tout marquer sur une certaine période de temps plutôt que de lui demander de marquer jusqu'à l'obtention d'un certain nombre de marqueurs, la raison étant que le nombre de marqueurs total varie beaucoup d'un patient à l'autre.

Les figures 3.1 et 3.2 montrent la répartition des marqueurs pour deux patients avec un grand nombre de marqueurs. On peut constater que les détections sont régulières de la cinquième minute jusqu'à la fin de l'enregistrement dans les deux cas.

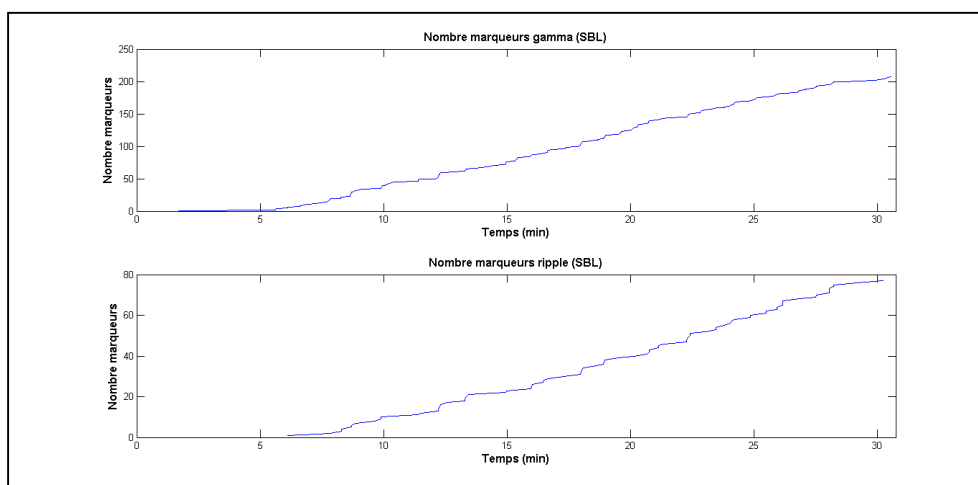


Figure 3.1 Stabilité temporelle SBL
En abscisse le temps et en ordonnée la somme cumulative du
nombre de marqueurs gamma (haut) et ripple (bas)

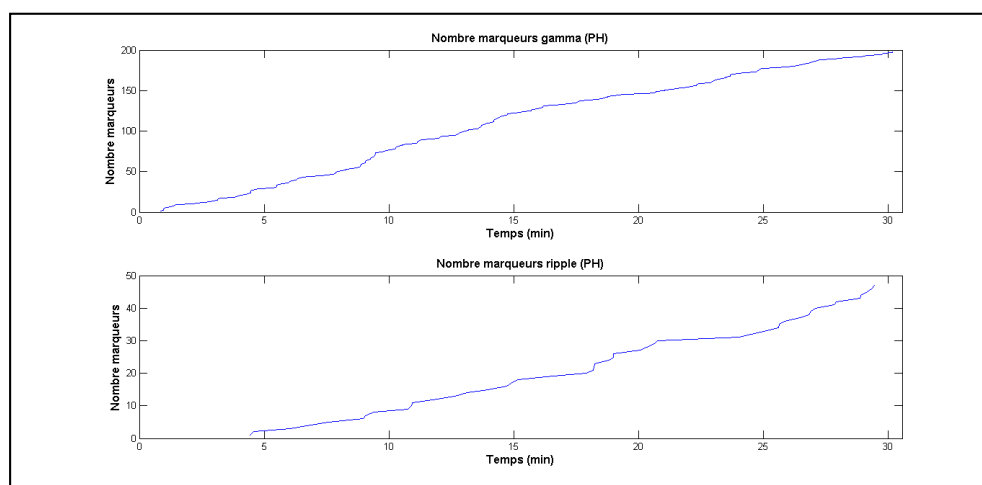


Figure 3.2 Stabilité temporelle PH
Voir figure 3.1 pour la légende

Pour ces deux cas, l'option de prendre une période de temps au lieu d'un nombre fixe de marqueurs serait valide. Il resterait à déterminer la durée de cette période pour assurer un taux de marqueurs représentatif de l'ensemble des données. On peut faire l'hypothèse que cette durée devrait être d'au moins 5 minutes, puisque dans le cas présenté, les cinq premières minutes ne sont pas représentatives de n'importe quel autre segment de même durée. La figure 3.3 et 3.4 montre la répartition totale des marqueurs sur deux patients avec un petit nombre de marqueurs.

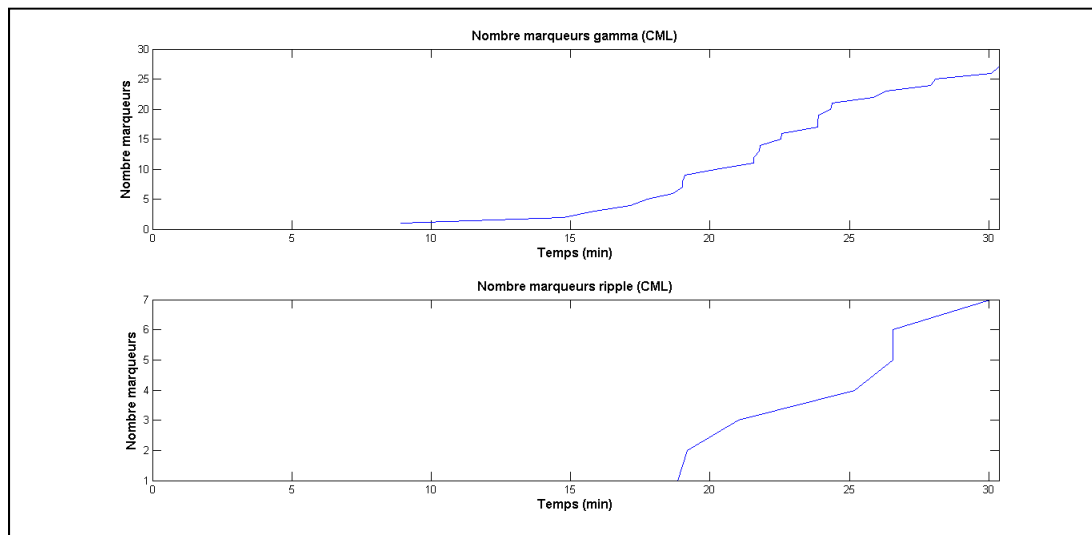


Figure 3.3 Stabilité temporelle (CML)
Voir figure 3.1 pour la légende

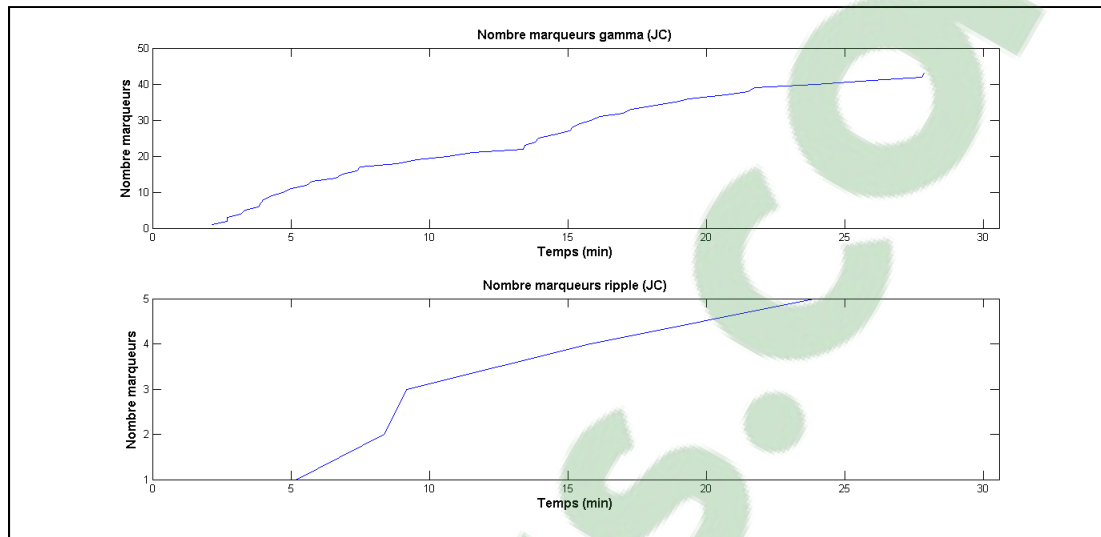


Figure 3.4 Stabilité temporelle (JC)
Voir figure 3.1 pour la légende

Dans le cas de la figure 3.3, une irrégularité temporelle est remarquée, les 7 marqueurs de la bande ripple étant trouvés dans les 10 dernières minutes, et presque aucun de la bande gamma dans les quinze premières minutes.

Dans le cas de la figure 3.4, malgré un nombre de marqueurs très limité en ripple (43 gamma et 5 ripple), ces marqueurs sont répartis également temporellement. En prenant tous les marqueurs contenus dans 10 minutes, il serait difficile d'obtenir plus de 2 marqueurs ripple. Par contre, si l'on demandait un nombre fixe de marqueurs ripple supérieur à 5, ce nombre ne serait jamais atteint avant d'avoir complété la détection visuellement.

Ce résultat réitère donc le besoin d'avoir un patient avec une forte activité de HFO pour assurer le bon fonctionnement de l'optimisation des paramètres de détection. Il n'est malheureusement pas possible de connaître cette activité à priori.

3.4 Stabilité d'amplitude

La méthodologie de détection de base est de comparer l'amplitude relative d'une oscillation à l'activité de fond dans une bande de fréquence de 10 Hz. Si le ratio entre l'amplitude de l'évènement détecté sur celle de l'activité de fond est significatif, l'évènement sera considéré comme un candidat HFO (von Ellenrieder *et al.* (2011)). Cette section vise à analyser ce ratio sur les marqueurs de l'expert, en utilisant la méthodologie suivante ;

- 1- pour un patient, filtrer les données dans une largeur de bande de 10 Hz, pour toutes les bandes entre 40 et 200 Hz;
- 2- calculer la valeur RMS $A(m,f)$ de tous les marqueurs m sur les canaux sur lesquels ils ont été détectés, sur la fenêtre complète de la détection, pour toutes les bandes de fréquences;
- 3- calculer également la valeur RMS $B(m,f)$ de la fenêtre d'une seconde qui précède chacun de ces marqueurs;
- 4- pour chaque marqueur, calculer le ratio $R(m,f) = A(m,f) / B(m,f)$. Conserver la valeur R la plus grande pour chaque marqueur, soit celui de la bande passante f où l'énergie du marqueur était maximale par rapport à l'activité de fond;
- 5- évaluer la moyenne R pour un patient en séparant les marqueurs définis comme Gamma et ceux définis comme Ripple;
- 6- vérifier si les valeurs R sont similaires entre les deux types de marqueurs;
- 7- vérifier la taille du sous-ensemble de marqueurs minimaux nécessaires pour être représentatifs de l'ensemble des marqueurs.

Le tableau suivant montre les valeurs des ratios R -gamma et R -ripple pour chaque patient. On montre également la différence entre les deux ratios, pour vérifier s'il y a une tendance à ce qu'un type de marqueurs ait une énergie relative supérieure à l'autre.

Tableau 3.2 Rapport d'énergie relative sur l'activité de background
(Première colonne) Chaque patient, la dernière ligne est la moyenne de chacun.
(colonne Marqueurs) le nombre de marqueurs de l'expert. (colonne R-Gamma)
Ratio d'énergie des marqueurs gamma sur leur background, filtré à 40-80 Hz.
(colonne R-ripple) Ratio d'énergie des marqueurs ripple sur leur background, filtré à
80-200 Hz. (colonne ripple-gamma) Différence entre la colonne 4 et 3

	Marqueurs	R-gamma	R-ripple	ripple – gamma
CML	27 gamma 7 ripple	1.55	2.26	0.71
DR	9 gamma 6 ripple	1.19	1.69	0.5
JC	43 gamma 5 ripple	2.56	2.36	-0.2
MA	72 gamma 25 ripple	2.21	2.42	0.21
PH	198 gamma 47 ripple	2.05	2.59	0.54
PJ	9 gamma 6 ripple	1.84	2.09	0.25
SBL	207 gamma 77 ripple	1.77	2.26	0.49
SG	135 gamma 33 ripple	2.09	1.98	-0.11
MOY	88 gamma 26 ripple	1.91	2.21	0.3

Les résultats du tableau 3.1 montrent que dans la majorité des cas (6 sur 8), le ratio R-gamma est inférieur au ratio R-ripple. Dans les deux cas où l'énergie relative est supérieure en gamma (JC et SG), les différences entre les deux ratios sont les plus près de 0 (-0.2 et -0.11).

Deux observations peuvent être tirées de ces résultats ;

- 1- il y a une tendance à ce que l'énergie relative des HFO en haute fréquence soit supérieure à ceux en basse fréquence;
- 2- cette énergie varie considérablement d'un patient à l'autre.

Ces observations supportent l'hypothèse qu'il faudrait rendre la détection variable en fréquence et adaptée pour chaque patient.

Les figures suivantes illustrent la distribution des ratios trouvés en n'utilisant qu'un sous-ensemble de marqueurs. Les sous-ensembles sont tous les marqueurs présents dans une fenêtre aléatoire de soit 2, 5, 7 ou 10 minutes, chacune répétée 20 fois avec une fenêtre différente.

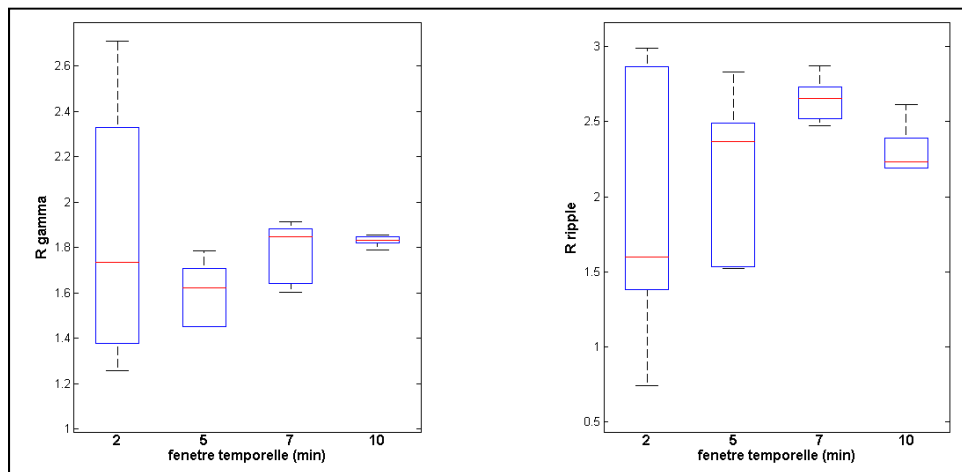


Figure 3.5 Énergie relative de sous-ensembles (SBL)
Variation d'énergie relative au background des marqueurs gamma (gauche)
et ripple (droite) en fonction du sous-ensemble utilisé

Le patient SBL a un R-gamma de 1.77 et un R-ripple de 2.26 en utilisant tous les marqueurs. Sur la figure 3.5, on y voit que l'utilisation d'une fenêtre de 2 ou 5 minutes ne permet pas de retrouver ces valeurs à chaque expérience, même si la valeur moyenne y est très proche. En utilisant une fenêtre de 7 minutes, la variabilité des valeurs est beaucoup diminuée. Avec une

fenêtre de 10 minutes, la variabilité est très faible, et les valeurs moyennes des deux ratios sont très près de ceux recherchés (1.74 et 2.23).

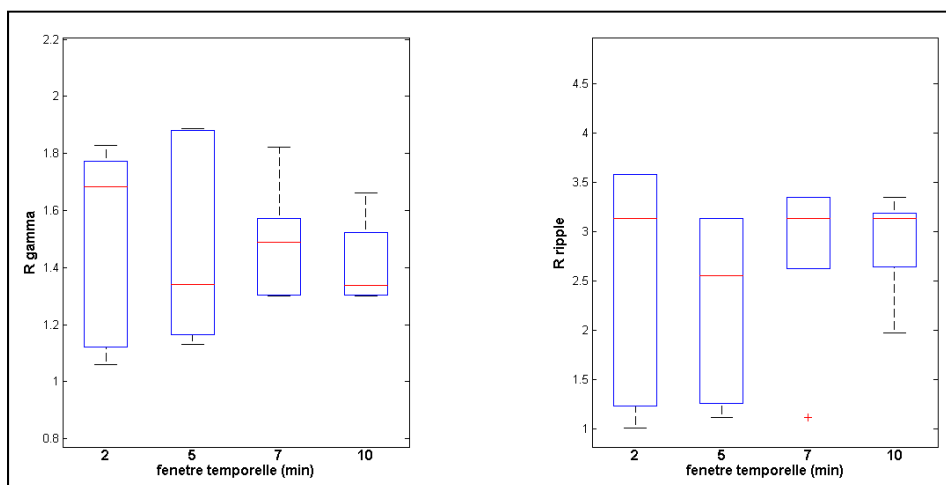


Figure 3.6 Énergie relative de sous-ensembles (CML)
Légende voir figure 3.5

La figure 3.6 représente les résultats pour le patient CML. Dans ce cas la moyenne du R-gamma est similaire peu importe la fenêtre utilisée, soit 1.52 pour la fenêtre de 10 minutes. Par contre, la variabilité des résultats augmente avec une fenêtre plus petite. Pour R-ripple, la variabilité est grande même avec une fenêtre de 10 minutes, avec une moyenne variant beaucoup selon la fenêtre utilisée.

Notons que pour la fenêtre de 10 minutes, le R-gamma (1.52) est très similaire à celui en utilisant tous les marqueurs (1.55), mais le R-ripple (2.76) l'est moins (2.26). Notons que CML ne possède que 7 marqueurs Ripple sur l'ensemble des données, ce qui peut expliquer la variabilité des résultats en fonction de la fenêtre choisie.

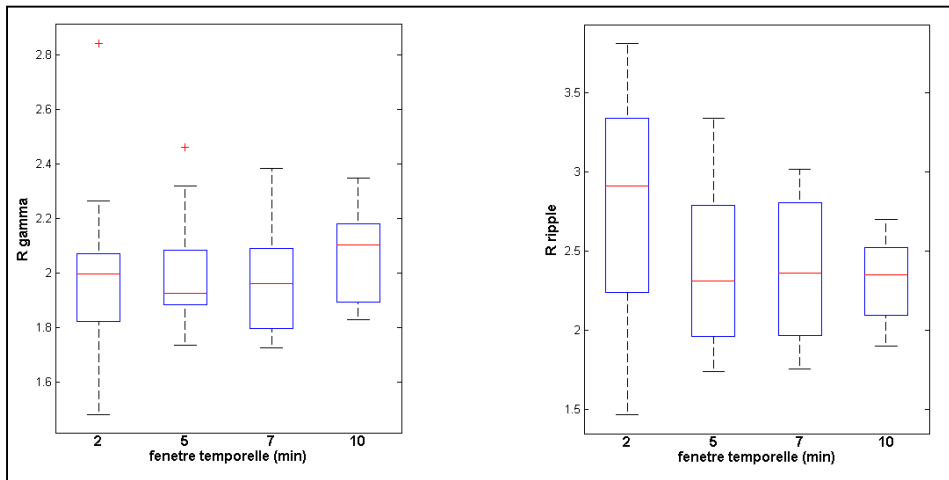


Figure 3.7 Énergie relative de sous-ensembles (PH)
Légende voir figure 3.5

Pour la patient PH, et le reste des patients, les résultats sont similaires. Le R-gamma moyen de la fenêtre de 10 minutes (2.07) est presque le même que celui avec tous les marqueurs (2.05). Le R-ripple (2.32) est un peu différent de celui avec tous les marqueurs (2.59), mais la variabilité est plus faible qu'avec les fenêtres plus petites.

Puisque le R-gamma et R-ripple sur 10 minute est similaire à ceux avec tous les marqueurs, nous concluons que les marqueurs de l'expert possèdent une énergie relative par rapport à leur activité de fond stable au cours du temps. Un sous-ensemble de marqueurs contenus sur un échantillonnage de 10 minutes permet de représenter cette énergie pour l'ensemble des marqueurs. Nous remarquons également une variabilité de cette énergie d'un patient à l'autre, ainsi qu'une variabilité en fonction de la fréquence d'oscillation des marqueurs.

3.5 Stabilité spatiale

Les caractéristiques spatiales des marqueurs de l'expert ne sont pas utilisées par le détecteur. Toutefois, on montre dans cette section que les marqueurs de l'expert sont spatialement corrélés, c'est-à-dire qu'ils sont presque tous trouvés sur les mêmes canaux. Ce résultat appuie l'hypothèse que les HFOs sont probablement plus présents dans la zone épileptogène.

Toutefois, à ce stade de la détection, il n'est pas souhaitable d'utiliser cette corrélation puisqu'elle ne fait pas partie de la définition actuelle des HFOs. En effet, l'expert faisant preuve de subjectivité dans sa détection, il est possible qu'il soit plus sensible à une oscillation trouvée sur un canal attribué à la zone épileptogène. Ce biais spatial serait alors transmis au détecteur puisqu'un sous-ensemble de marqueurs est utilisé.

Néanmoins, cette corrélation est observée et la section suivante visera à déterminer l'échantillon de marqueurs nécessaires pour refléter cette propriété. Elle analysera également l'impact de la méthode de sélection des marqueurs.

Nous cherchons donc à « dessiner » la carte des canaux détectés par l'expert dans un sous-ensemble de marqueurs. Si cette carte est la même pour plusieurs sous-ensembles des marqueurs de même dimension, nous considérerons cette dimension suffisante pour bien représenter les caractéristiques spatiales de tous les marqueurs. La méthodologie pour trouver ces cartes est la suivante ;

Nous utilisons tous les marqueurs contenus dans 10 minutes d'enregistrement, puisque la section précédente a suggéré que cela offrait une représentation adéquate de tous les marqueurs. Comme il est difficile de représenter la stabilité d'une expérience à l'autre sur une carte spatiale, seulement les résultats stables d'une expérience à l'autre seront montrés. La figure 3.9 montre les cartes spatiales utilisant 40 marqueurs gamma et 20 marqueurs ripple avec 3 échantillons aléatoires différents.

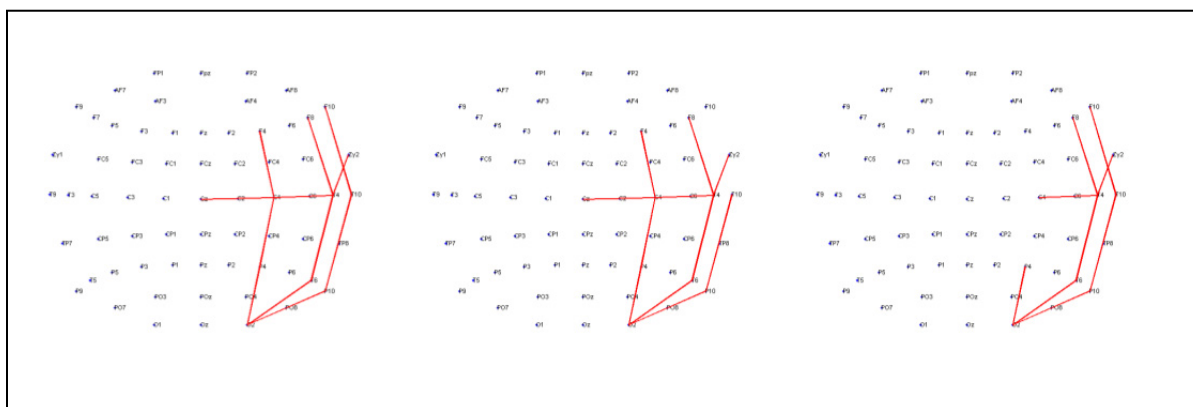


Figure 3.9 Cartes spatiales 40-20

Chaque ligne est un canal avec au moins un marqueur de l'expert. Chaque carte représente un sous-ensemble de 40 marqueurs gamma et 20 marqueurs ripple

On remarque que bien qu'elle soit presque complète par rapport à la référence de la figure 3.9, la carte varie légèrement d'une itération à l'autre. Elle est globalement correcte, mais pas suffisamment pour assurer que l'échantillonnage représente fidèlement la référence 3.8. La figure 30.10 refait l'expérience en utilisant tous les marqueurs sur 10 minutes de trois fenêtres différentes.

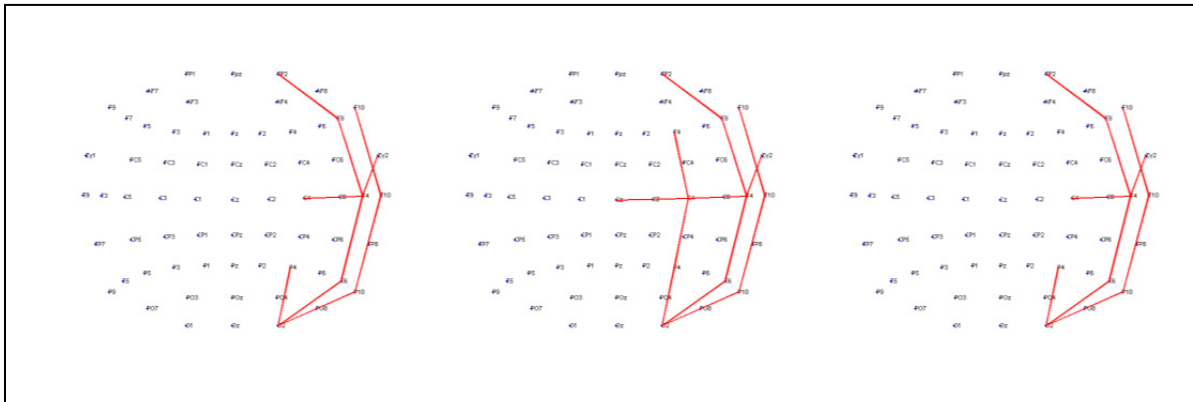


Figure 3.10 Cartes spatiales 10 min

Chaque ligne est un canal avec au moins un marqueur de l'expert. Chaque carte représente un sous-ensemble de tous les marqueurs contenus sur 10 minutes

Les cartes de la figure 3.10 sont très similaires, en plus de l'être avec la référence de la figure 3.8. Malgré les différences mineures, l'utilisation d'un sous-ensemble de tous les marqueurs contenus dans 10 minutes offre une meilleure représentation de l'ensemble des marqueurs qu'une sélection d'un nombre fixe. Puisque la carte de la figure 3.8 peut être reproduite avec un de ces sous-ensembles.

Les trois cartes de la figure 3.10 ne sont pas identiques. Une seule des itérations, la deuxième de la figure 3.10, est identique avec la référence de la figure 3.8. Dans tous les cas il est difficile d'arriver à exactement la même carte, puisque certaines connexions pourraient possiblement dépendre d'une seule détection sur l'ensemble des données. Nous sommes toutefois en mesure de valider que cette relation existe, et qu'elle est raisonnablement représentée avec un sous-ensemble de marqueurs.

3.6 Conclusion

Cette section a démontré qu'il existe des caractéristiques d'amplitudes, temporelles et spatiales dans les marqueurs de l'expert. Ces corrélations peuvent être extraites dans un sous-ensemble de marqueurs dans le but d'augmenter la performance du détecteur automatique.

L'utilisation de 10 minutes de marqueurs donne des résultats stables et satisfaisants, c'est la méthode qui sera utilisée pour ce travail. Cette méthode a l'avantage d'être une demande plus réaliste et pratique à un expert que de lui demander un nombre fixe de marqueurs. Sans information préalable, il est impossible de prédire si un nombre fixe de marqueurs sera à la fois raisonnable et représentatif de l'ensemble des HFO présents. En utilisant une période fixe de 10 minutes, il sera probable que l'échantillon sera suffisant, et la charge de travail sera diminuée par un facteur 3 pour un enregistrement de 30 minutes, soit environ 10 heures de travail sauvé.

Une corrélation spatiale dans les marqueurs de l'expert a également été observée. Néanmoins, cette caractéristique ne sera pas utilisée dans la détection puisque nous désirons éviter un trop grand biais afin d'être plus sensible. Par contre, le critère spatial sera utilisé dans le système de score présenté au chapitre 5, qui attribuera un poids à chaque détection. En réalité, une détection faite dans la même région que les marqueurs est plus probable d'être un vrai positif qu'une détection à l'extérieur. Il sera également intéressant de vérifier si les détections automatiques sont principalement localisées dans cette zone, ce qui sera fait dans le chapitre des résultats.

CHAPITRE 4

APPRENTISSAGE

4.1 Objectif et méthodologie

Le système d'apprentissage proposé est présenté sur la figure 4.1 ci-dessous. Rappelons que le détecteur initial possède un seuil de détection et deux seuils de classification. Cet algorithme vise à déterminer les valeurs des seuils les plus spécifiques en maintenant une grande sensibilité. Il consiste à créer des segments avec le sous-ensemble de marqueurs de l'expert sur 10 minutes, tel que validé dans la section précédente. Ces segments sont de 3 secondes, centrés sur le moment de détection. Le seuil de détection est fixé très bas, pour obtenir une sensibilité maximale, et cette sensibilité est enregistrée comme référence.

Le processus recommence ensuite avec un seuil légèrement plus élevé, et cette boucle continue tant que la sensibilité reste supérieure à 90%. Cette valeur de sensibilité est plus élevée que celle des objectifs de performance (75%), mais puisque l'apprentissage est fait sur une sous-ensemble de marqueurs, nous nous attendons à ce que la sensibilité réelle diminue en utilisant tous le signal.

Le résultat est enregistré pour les segments gamma et pour les segments ripple. Le processus pour les seuils de classification est très similaire.

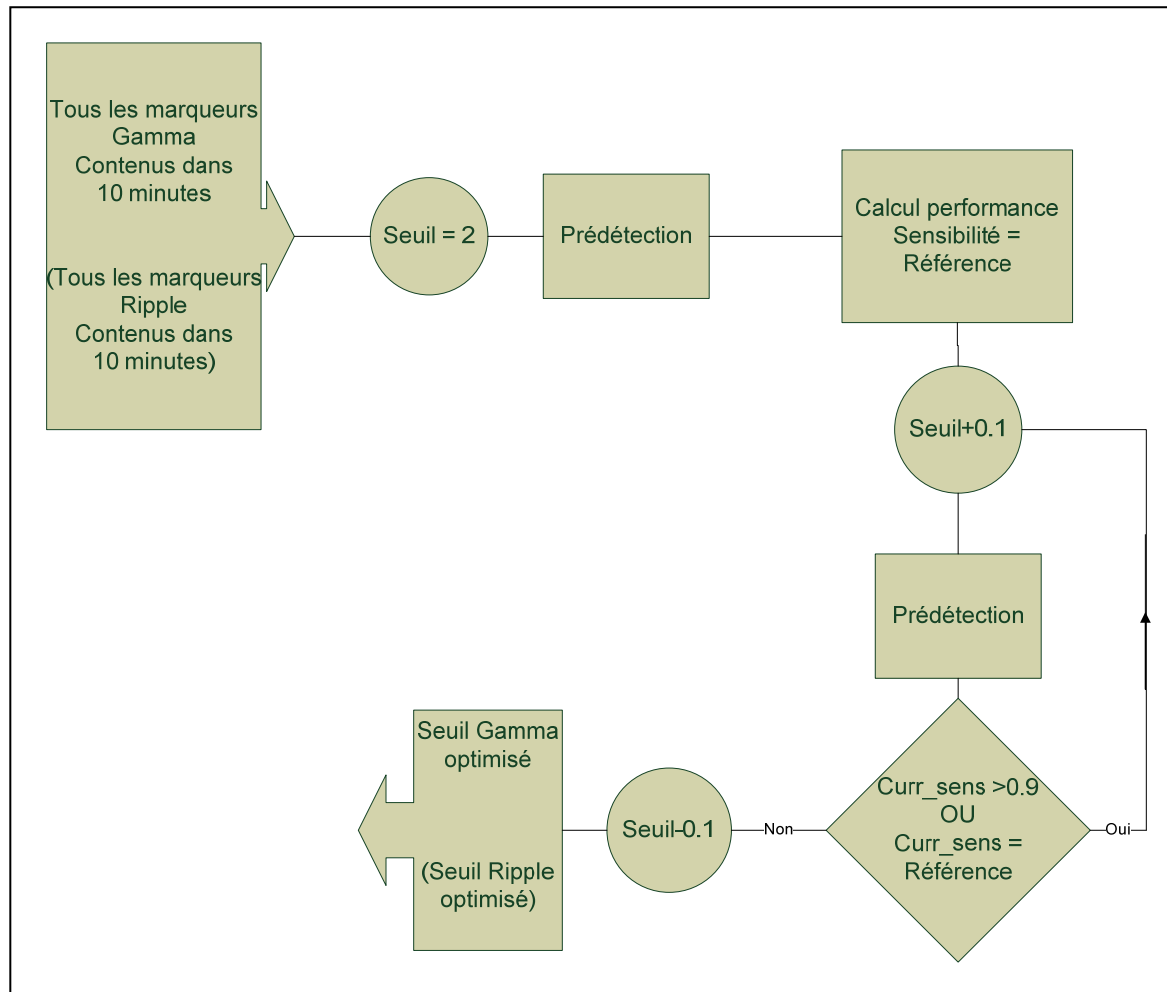


Figure 4.1 Diagramme d'apprentissage du détecteur

En commençant avec un seuil faible (2), la sensibilité sera élevée. La boucle augmente graduellement ce seuil pour obtenir la valeur la plus spécifique en conservant une sensibilité supérieure à 90% ou égale à la référence. Quand la sensibilité baisse, la valeur du seuil de l'itération précédente est conservée pour ce patient

À ce stade, nous obtenons 6 seuils (TH_gamma, TH_ripple, THA_gamma, THA_ripple, THB_gamma et THB_ripple), 3 pour la bande gamma et 3 pour la bande ripple. Pour obtenir le seuil optimal pour chaque bande étroite, les résultats sont interpolés de 40 à 200 Hz commençant du seuil en gamma vers celui en ripple. Le résultat donne un résultat différent pour chaque sujet, mais l'allure de chaque droite est similaire à la figure 4.2.

Les croix rouges correspondent aux valeurs trouvées lors de l'apprentissage. La première de chaque ligne, à 45 Hz, est la valeur trouvée avec le sous-ensemble de marqueurs gamma. La dernière, à 195 Hz, correspond à la valeur trouvée avec les marqueurs ripple. Les 14 autres valeurs sont trouvées par l'interpolation linéaire entre les valeurs intermédiaires de deux seuils. Ce modèle fut choisi pour sa simplicité, et parce rien ne laisse croire à une relation non-linéaire lorsque la fréquence augmente.

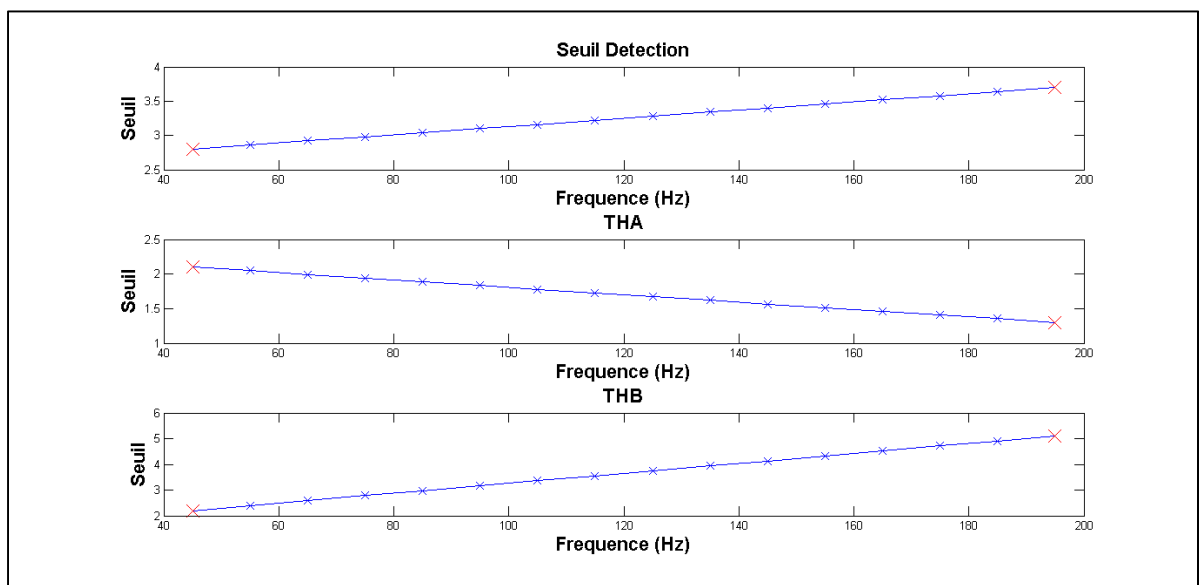


Figure 4.2 Seuils optimisés par apprentissage

En abscisse est la fréquence de chaque bande et en ordonnée le seuil associé. Les premières (45 Hz) et les dernières (195 Hz) valeurs sont celles trouvées par l'apprentissage. Les autres valeurs sont interpolées linéairement

4.2 Résultats actuels

Rappelons que la performance d'un détecteur est directement évaluée par sa capacité à trouver les événements marqués par l'expert, cette référence est appelée le « *gold-standard* ». Compte tenu de la subjectivité de ce gold-standard, une performance « parfaite » n'est donc pas possible. La détection visuelle par un humain, avec la méthodologie de la section précédente, possède une variabilité inhérente. Cette détection sera donc intrinsèquement biaisée par la subjectivité de l'expert.

Le détecteur présenté dans ce travail a l'avantage d'être déterministe, donc qu'il donne le même résultat chaque fois qu'il est lancé sur les mêmes données avec les mêmes paramètres. Compte tenu de la subjectivité de la détection visuelle ainsi que de l'expertise récente en la matière, nous souhaitons souligner qu'il est peu recommandé de rejeter sans considération les événements trouvés par le détecteur et qui n'ont pas été marqué par l'expert. Dans le même ordre d'idée, les marqueurs de l'expert qui n'ont pas été trouvés par le détecteur ne doivent pas nécessairement être classés comme des faux-négatifs.

Sur la base de huit patients, le tableau 4.1 compare les résultats de performance entre le détecteur initial et celui utilisant l'optimisation des paramètres du chapitre 3. Notons que la comparaison des deux détecteurs peut être faite puisque les résultats utilisent les mêmes marqueurs de l'expert. Au niveau du détecteur seulement (sans classification), le résultat du détecteur optimisé (sens = 84%, FPR = 57) offre une sensibilité similaire au détecteur initial (sens = 89%, FPR = 116) avec un gain important en spécificité. En effet, 116 détections par minute sont l'équivalent d'environ 1 par 500 msec, ce qui n'est pas satisfaisant. Avec optimisation, la sensibilité chute légèrement (5%), mais le nombre de détections totales diminue de moitié. Au niveau de la classification, le résultat du détecteur optimisé est moins intéressant, avec une perte de sensibilité importante (31%).

Tableau 4.1 Comparaison détecteur initial et optimisé
 Le premier encadré contient la performance du détecteur initial. Le deuxième contient celle du détecteur avec apprentissage. La sensibilité (sens) est en % et le taux de faux positifs (FPR), est par minute

		Détecteur initial		Détecteur optimisé		
	Marqueurs	Détection	Classification	Détection	Gain	Classification
CML	27 gamma 7 ripple	Sens = 91 FPR = 131	Sens = 50 FPR = 2	Sens = 97 FPR = 99	Sens = 6 FPR = -32	Sens = 71 FPR = 34
DR	9 gamma 6 ripple	Sens = 73 FPR = 94	Sens = 33 FPR = 1	Sens = 53 FPR = 35	Sens = -20 FPR = -59	Sens = 33 FPR = 11
JC	43 gamma 5 ripple	Sens = 89 FPR = 95	Sens = 88 FPR = 5	Sens = 88 FPR = 32	Sens = -1 FPR = -63	Sens = 50 FPR = 18
MA	72 gamma 25 ripple	Sens = 99 FPR = 135	Sens = 90 FPR = 54	Sens = 96 FPR = 58	Sens = -3 FPR = -77	Sens = 91 FPR = 40
PH	198 gamma 47 ripple	Sens = 82 FPR = 92	Sens = 50 FPR = 12	Sens = 87 FPR = 96	Sens = 5 FPR = 4	Sens = 32 FPR = 40
PJ	9 gamma 6 ripple	Sens = 88 FPR = 133	Sens = 55 FPR = 1	Sens = 78 FPR = 15	Sens = -10 FPR = -118	Sens = 6 FPR = 8
SBL	207 gamma 77 ripple	Sens = 97 FPR = 155	Sens = 81 FPR = 63	Sens = 89 FPR = 70	Sens = -8 FPR = -85	Sens = 89 FPR = 64
SG	135 gamma 33 ripple	Sens = 92 FPR = 90	Sens = 32 FPR = 6	Sens = 83 FPR = 37	Sens = -9 FPR = -53	Sens = 51 FPR = 22
MOY	88 gamma 26 ripple	Sens = 89 FPR = 116	Sens = 60 FPR = 18	Sens = 84 FPR = 57	Sens = -5 FPR = -59	Sens = 53 FPR = 30

On note que la sensibilité élevée est préférée à un faible FPR. La raison étant qu'avec une forte sensibilité, il est toujours possible d'augmenter la spécificité avec des étages de classification supplémentaires. L'inverse n'est pas vrai, la sensibilité ne pouvant que

décroître à chaque étape. Un FPR très élevé peut toutefois rendre la détection automatique inutile, il est évidemment souhaitable d'avoir un minimum de faux-positifs.

La figure 4.3 est une autre représentation des résultats du tableau 4.1. Les croix rouges représentent chaque patient en utilisant le détecteur initial, tandis que les cercles bleus représentent les mêmes patients en utilisant le détecteur optimisé. Le gain en spécificité au niveau de la détection est analysé par le mouvement des cercles bleus vers le bas, sans aller vers la gauche. Au niveau de la classification, on peut constater que l'optimisation est moins efficace pour retirer les faux positifs par la présence plus nombreuse de cercles bleus en hauteur.

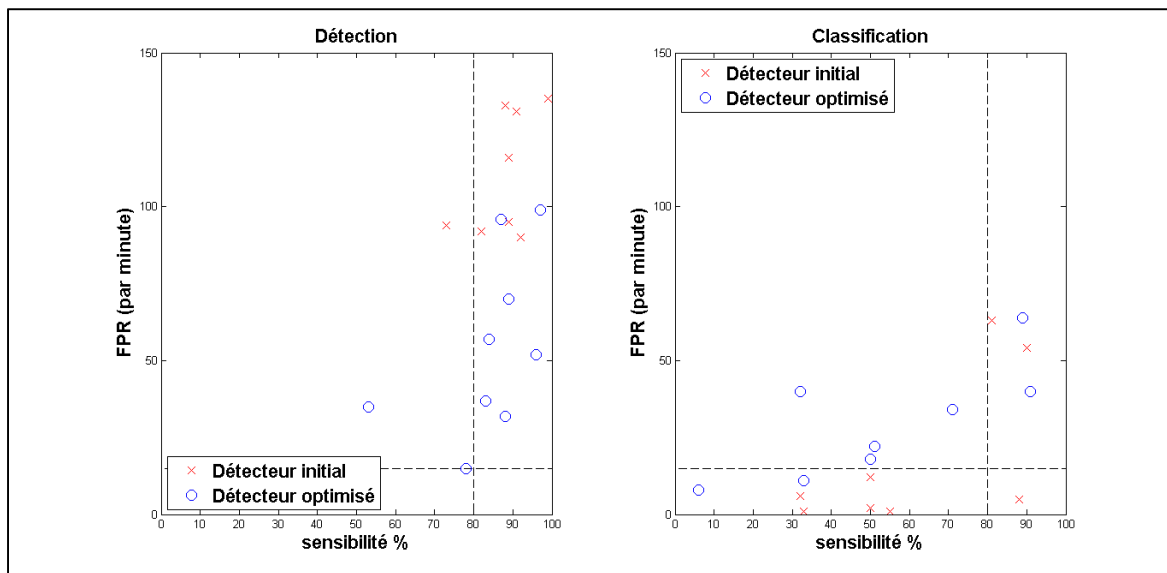


Figure 4.3 ROC apprentissage

Cercles bleus : Les 8 patients en utilisant le détecteur optimisé. Croix rouges : Les 8 mêmes patients en utilisant le détecteur initial. En abscisse est la sensibilité et en ordonnée est le taux de faux positifs

Notons également que les résultats de l'optimisation sont faibles pour les deux sujets ayant très peu de marqueurs (DR et PJ). Considérant que la méthodologie d'apprentissage repose sur un sous-échantillon de marqueurs contenus sur 10 minutes, la variabilité inhérente des

marqueurs causera un biais dans l'apprentissage plus important chez les patients ayant peu de marqueurs. Si l'on exclut ces sujets, les sensibilités moyennes sont améliorées pour l'ensemble des techniques, tel que présenté au tableau 4.2.

Tableau 4.2 Comparaison détecteur initial et optimisé sans PJ et DR
La première ligne est la performance de tous les patients. La deuxième est la performance en excluant les patients avec peu de marqueurs. La sensibilité (sens) est en % et le taux de faux positifs (FPR), est par minute

	Marqueurs	Détecteur original	Classification originale	Détecteur optimisé	Classification optimisée
MOY avec tous les sujets	88 gamma 26 ripple	Sens = 89 FPR = 116	Sens = 60 FPR = 18	Sens = 84 FPR = 57	Sens = 53 FPR = 30
MOY sans PJ et DR	114 gamma 32 ripple	Sens = 91 FPR = 116	Sens = 65 FPR = 24	Sens = 89 FPR = 64	Sens = 64 FPR = 36

Nous concluons des résultats des tableaux 4.1 et 4.2 que l'apprentissage est une méthode efficace pour améliorer la performance concernant la détection. Au niveau de la classification, l'apprentissage n'est pas efficace par rapport au modèle fixe proposé dans le détecteur initial. Même sans apprentissage, ce modèle de classification engendre une perte importante en sensibilité pour le détecteur initial (-26%). Il est donc souhaitable de trouver une autre méthode de classification qui viendrait remplacer le modèle de classification actuellement proposé.

CHAPITRE 5

CLASSIFICATION FINALE

5.1 Introduction au score

Pour permettre au détecteur de séparer les vrais positifs des faux positifs, nous proposons d'utiliser des caractéristiques qui ne sont pas incluses dans la définition actuelle d'un HFO, mais qui ont été observées chez les marqueurs de l'expert. Ces caractéristiques sont communes à la majorité des marqueurs, et peuvent être vérifiées avec le même sous-ensemble que celui utilisé pour l'apprentissage des seuils. Il est donc possible d'augmenter la spécificité du détecteur en mettant de côté les événements détectés qui ne possèdent pas ces caractéristiques, celles-ci seront expliquées plus en détail dans les prochaines sections.

Définissons les caractéristiques suivantes, pour chaque événement e ;

$\alpha(e)$ = proximité à une pointe épileptique;

$\beta(e)$ = caractéristique spatiale;

$\gamma(e)$ = durée maximale de la détection.

À partir de ces caractéristiques, nous proposons la définition d'un score qui sera attribué à chaque événement. Ce score représente la vraisemblance d'un événement détecté par rapport aux marqueurs de l'expert. Cette stratégie possède le risque de biaiser la détection pour ne trouver que les événements qui ressemblent à celui de l'expert. Or, tel que mentionné en 5.1, le marquage de l'expert n'est pas « parfait », et ce biais pourrait éliminer de vrais HFOs qui sortent du modèle établi par l'expert.

Le score S de chaque évènement est défini selon l'équation suivante :

$$S(e) = \alpha(e) + \beta(e) + \gamma(e) \quad (5.1)$$

S est calculé en fonction des caractéristiques définie plus haut. Un score élevé servira d'indicateur de vrai positif selon la définition de l'expert, un score faible indiquera un faux positif. Comme ces caractéristiques ne sont pas définies dans le modèle actuel d'un HFO, ce score ne servira pas à retirer des évènements, mais simplement à aider l'utilisateur à ignorer ou à prioriser certaines détections s'il le souhaite. Ainsi faisant, la spécificité devrait être augmentée en fonction de la tolérance à l'erreur. Si l'utilisateur ne recherche que les HFO « certains », seuls les scores élevés pourraient être conservés. Si à l'inverse le recensement complet des HFO potentiels est important, même les scores faibles peuvent être conservés.

5.2 La proximité des pointes épileptiques (Melani *et al.* (2013))

Le premier critère observé dans les marqueurs de l'expert est la cooccurrence entre les HFOs et les pointes épileptiques. Une pointe épileptique est une activité cérébrale spontanée, imprévisible, de courte durée, et est un des principaux biomarqueurs épileptogènes pour la neurologie actuelle (Gotman (1999)). Elle est mieux comprise des experts que les HFOs, puisque les neurologues s'y intéressent depuis plusieurs décennies, tandis qu'on s'intéresse aux HFOs seulement depuis à quelques années (Zelmann *et al.* (2009)).

On observe que les HFOs arrivent fréquemment dans la même fenêtre temporelle qu'une pointe (Melani *et al.* (2013)). Nous définissons la cooccurrence comme la proportion d'un évènement arrivant à l'intérieur de 50 msec d'un autre type d'évènement. Les résultats de l'étude sont les suivants :

Tableau 5.1 Statistique de cooccurrence
 La cooccurrence $A \in B$ est la probabilité d'avoir
 un évènement B à l'intérieur de 50 msec au
 moment d'un évènement A
 Tirée de Melanie et al. (2013)

Cooccurrence Pointe $\in$ Gamma	9.5%
Cooccurrence Pointe $\in$ Ripple	3.2%
Cooccurrence Gamma $\in$ Pointe	46.2%
Cooccurrence Ripple $\in$ Pointe	44.4%

La présence d'un HFO (gamma ou ripple) est donc un indicateur de la présence d'une pointe à proximité dans environ la moitié des cas. Par contre, la présence d'une pointe n'indique pas la présence d'un HFO. Notons également que les pointes sont beaucoup plus nombreuses que les HFOs gamma, donc encore plus que les HFOs ripple. Notons aussi que les patients avec plus de pointes détectées (groupe A) avaient proportionnellement plus de HFOs. Le groupe avec presque qu'aucune pointe (groupe D) n'avait également très peu de HFOs, tel que sur la figure suivante :

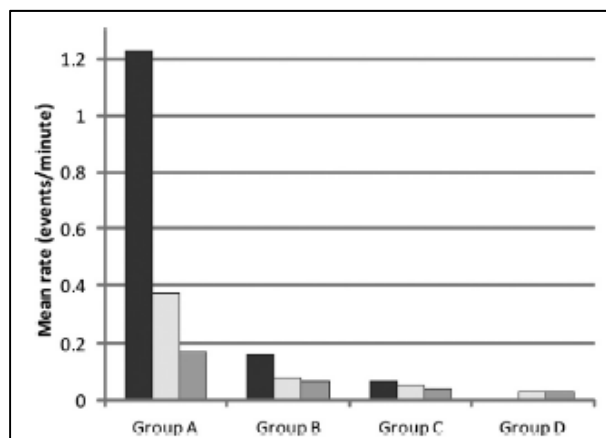


Figure 5.1 Taux des marqueurs de l'expert
 Colonne noire : Taux de pointes; Colonne grise pâle : Taux de
 HFO gamma; Colonne grise foncé : Taux de HFO ripple
 Tirée de Melani et al. (2013)

Considérons maintenant la valeur α dans le calcul du score. Étant donné que la cooccurrence HFO-pointe n'est valide que dans environ 50% des cas, α ne devrait pas avoir un impact trop grand sur le score S. Le score sera donc influencé par ce critère, mais les critères β et γ devront balancer le score de manière à ce qu'aucun critère ne soit trop dominant.

La détection des pointes n'est pas problématique étant donné la plus grande expertise pour ce type de détection que pour les HFOs. En effet, il existe plusieurs détecteurs automatiques de pointes qui permettent d'accomplir cette tâche avec de bons résultats (Dubarry (2010)). Nous pourrions donc donner un score supérieur à une détection HFO qui possède une pointe à l'intérieur de 50 msec sur n'importe quel canal, et un score inférieur dans le cas contraire. Nous attribuons un score initial S de 0 à chaque détection automatique e. Dans le cas où une pointe est marquée à proximité, ce score est augmenté de 0.2, ou diminué de 0.2 dans le cas inverse. Puisqu'un détecteur automatique de pointes est disponible, notons que l'utilisation de ce critère n'ajoute aucun coût supplémentaire à la détection.

5.3 La proximité spatiale

La même étude sur la cooccurrence HFO-pointes démontre également que les HFOs sont très restreints à une zone épileptogène identifiée SOZ (Melani *et al.* (2013)). Il est observé que les taux de pointes épileptiques et de HFOs sont significativement plus élevés sur les canaux de la SOZ. Cette correspondance spatiale indique que les générateurs de ces deux signaux partagent une partie de leurs réseaux neuronaux.

Définissons la précision P comme suit :

$$P = (\text{canaux SOZ avec HFO} + \text{canaux hors SOZ sans HFO}) / \text{nombre de canaux} \times 100 \quad (5.2)$$

P est donc un indicateur des HFOs qui ont été restreintes à la SOZ. Le calcul fut réalisé pour les trois type de marqueurs : Pointes, HFO gamma et HFO ripple. Sur 558 canaux analysés,

86 furent à l'intérieur du SOZ. La méthode qui permet de définir la SOZ n'est pas expliquée. La précision de chaque type de marqueur est présentée dans le tableau suivant :

Tableau 5.2 Précision des marqueurs

P pointe	P gamma	P ripple
54%	74%	77%

Ces résultats montrent que la majorité des marqueurs HFOs sont contenus dans une zone restreinte ainsi que principalement dans la zone épileptogène définie par l'expert. Ce résultat contribue à augmenter l'intérêt que nous portons aux HFOs comme biomarqueurs pour remplacer ou compléter les pointes dans la recherche du SOZ des patients épileptiques. Il contribue également à la décision d'inclure le critère spatial dans le calcul du score de nos détections automatiques.

Dans la section 2.4, nous avons montré que les marqueurs de l'expert sont restreints à certains canaux. Avec un sous-ensemble de marqueurs contenus sur 10 minutes, nous sommes en mesure de recenser ces canaux spécifiques. Une détection automatique contenue à l'intérieur de la SOZ est donc plus probable d'être un HFO que ceux à l'extérieur.

Nous ne connaissons pas les canaux attribués au SOZ par l'expert. Nous sommes toutefois en mesure de l'estimer avec notre sous-ensemble de marqueurs. Nous définissons donc la SOZ par tous les canaux contenant un HFO dans le sous-ensemble de marqueurs utilisés. Les détections contenues dans la SOZ auront plus de ressemblance aux marqueurs de l'expert que ceux hors SOZ.

Chaque détection est possiblement faite sur plus d'un canal. Le score de chaque détection sera augmenté de 0.1 pour chaque canal détecté à l'intérieur de la SOZ et diminué de 0.1 pour chaque canal détecté hors SOZ. Dans le cas où la détection est très étendue, le score peut potentiellement être très influencé par ce critère spatial β . Comme les canaux à

l'intérieur de la SOZ sont généralement moins nombreux que les canaux hors SOZ, un artefact qui causerait une détection sur beaucoup de canaux risque d'avoir son score grandement diminué. Dans le cas d'une HFO très localisée dans la SOZ, ce score sera élevé.

Ici aussi, nous désirons éviter que le critère β soit dominant. Les HFOs hors de la SOZ sont rares, mais possible. Dans le cas d'un HFO localisé sur un seul canal hors de la SOZ, le score sera peu diminué. Par rapport au critère précédent, s'il est à proximité d'une pointe, ce HFO hors SOZ conservera un score supérieur à 0.

5.4 La durée maximale de la détection

La définition utilisée d'un HFO requiert un minimum de quatre oscillations. Selon les fréquences vérifiées, soit de 40 Hz à 200 Hz, un HFO sera donc théoriquement d'au moins 20 msec à 100 msec dépendamment de la fréquence observée. Aucun critère n'est fixé sur la durée maximale, toutefois, en pratique, un HFO possède rarement plus de quatre oscillations. Nous avons observé la durée moyenne des marqueurs de l'expert, tel que rapportés dans le tableau suivant :

Tableau 5.3 Durée moyenne des marqueurs
La troisième colonne est la durée moyenne de tous les marqueurs de
chaque patient (ligne)

	Marqueurs	Durée moyenne (msec)
CML	27 gamma	Durée = 101
	7 ripple	Durée = 65
DR	9 gamma	Durée = 90
	6 ripple	Durée = 74
JC	43 gamma	Durée = 94
	5 ripple	Durée = 75
MA	72 gamma	Durée = 103
	25 ripple	Durée = 58
PH	198 gamma	Durée = 108
	47 ripple	Durée = 61
PJ	9 gamma	Durée = 110
	6 ripple	Durée = 82
SBL	207 gamma	Durée = 112
	77 ripple	Durée = 86
SG	135 gamma	Durée = 103
	33 ripple	Durée = 66
MOY	88 gamma	Durée = 103
	26 ripple	Durée = 71

Ces résultats démontrent que malgré l'étendue théorique de la durée des HFO (80 msec), les marqueurs ripple ont une durée moyenne (71 msec) similaire à celle des HFO gamma (103 msec). Comme la détection des HFOs devient de plus en plus difficile en augmentant la fréquence, ce résultat est attribué à la détection des HFO ripple plus fréquents près de 80 Hz que près de 200 Hz. Néanmoins, les cas d'HFOs de très longue durée sont très rares.

Certains artéfacts ont une durée supérieure aux HFOs les plus longs. Il est donc possible de pénaliser le score des détections très longues sans risquer de diminuer celui des HFOs potentiels. Le dernier critère qui influencera le score de chaque détection est donc sa durée. Dans le cas où la détection est faite sur plus de 150 msec, la valeur γ sera de -0.5. Contrairement aux autres critères, le score ne peut que diminuer. Rien ne porte à croire qu'une détection plus courte est plus probable d'être un HFO qu'une plus longue, tant qu'elle ne dépasse pas une certaine limite. Le score S sera donc calculé selon la formule suivante :

$$S(e) = \alpha(e) + \beta(e) + \gamma(e) \quad (5.3)$$

$\alpha = +/- 0.2$ si il y a cooccurrence d'une pointe ou pas

$\beta = +/- 0.1$ par canal détecté dans SOZ/hors-SOZ

$\gamma = -0.5$ si la durée est supérieure à 150 msec

Notons que le poids de chaque critère fut déterminé empiriquement en maximisant la performance sur deux sujets, SBL et PH. Les sujets choisis furent celui avec le FPR le plus faible (SBL) et celui avec le FPR le plus élevé (PH). L'objectif étant que les scores soit majoritairement positif sur le sujet avec peu de faux-positifs, et négatif avec l'autre. Si le modèle fonctionne bien avec les deux extrêmes de spécificité, il est probable qu'il soit applicable à n'importe quel sujet.

Pour bien illustrer le ratio de scores bas et de scores élevés, il est préférable d'évaluer la distribution des scores un sujet à la fois. La distribution typique des scores sur les détections d'un patient est la suivante :

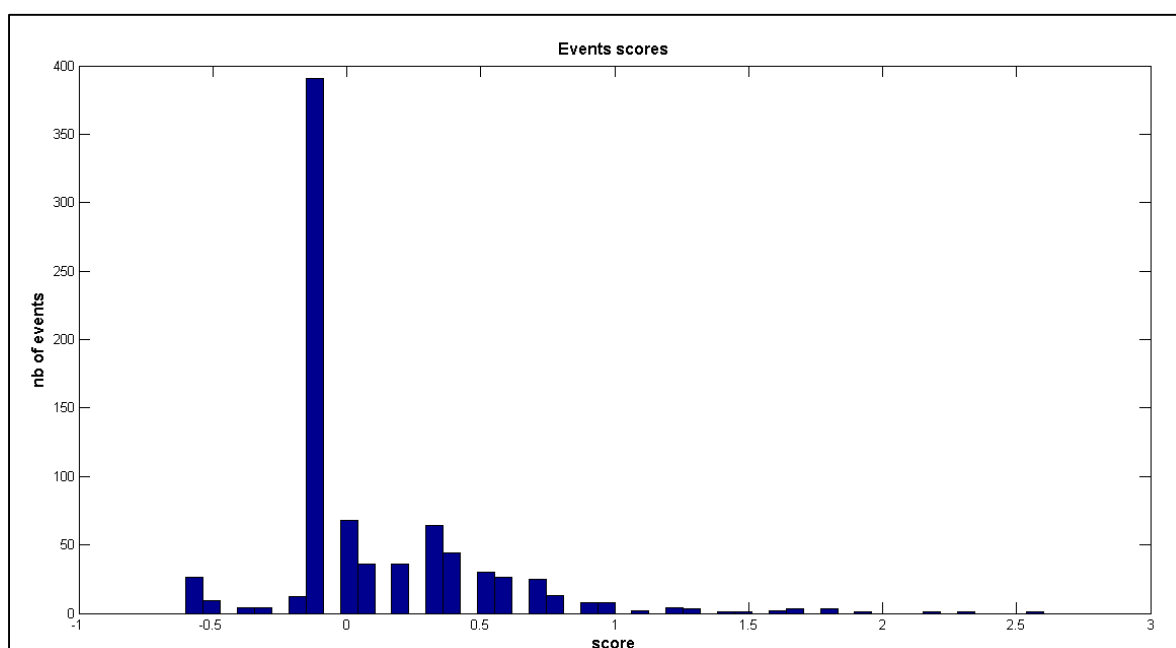


Figure 5.2 Distribution des scores (SBL)

Histogramme des scores des évènements détectés. En abscisse : Le score de chaque évènement; en ordonnée, le nombre d'évènements avec ce score. On remarque que la majorité des évènements ont un score inférieur à zéro

5.5 Résultats

Chaque évènement possède donc un score S qui représente sa vraisemblance aux marqueurs de l'expert en fonction des trois critères présentés dans les sections précédentes. L'objectif de ce score est d'ordonner les détections pour permettre d'isoler les faux positifs. La performance de ce système dépend de l'équilibre des trois critères. Cet équilibre peut être validé par un calcul de performance sur les détections en ne considérant que les scores au-dessus d'un certain seuil. Il est souhaitable que les premiers évènements à être retirés, soit les scores les plus faibles, soient des "vrais" faux positifs. Comme il est improbable qu'un vrai positif n'ait rencontré aucun des trois critères, leur score devrait être élevé.

Les figures suivantes analysent l'impact de rejeter les détections avec un score S en dessous d'un seuil.

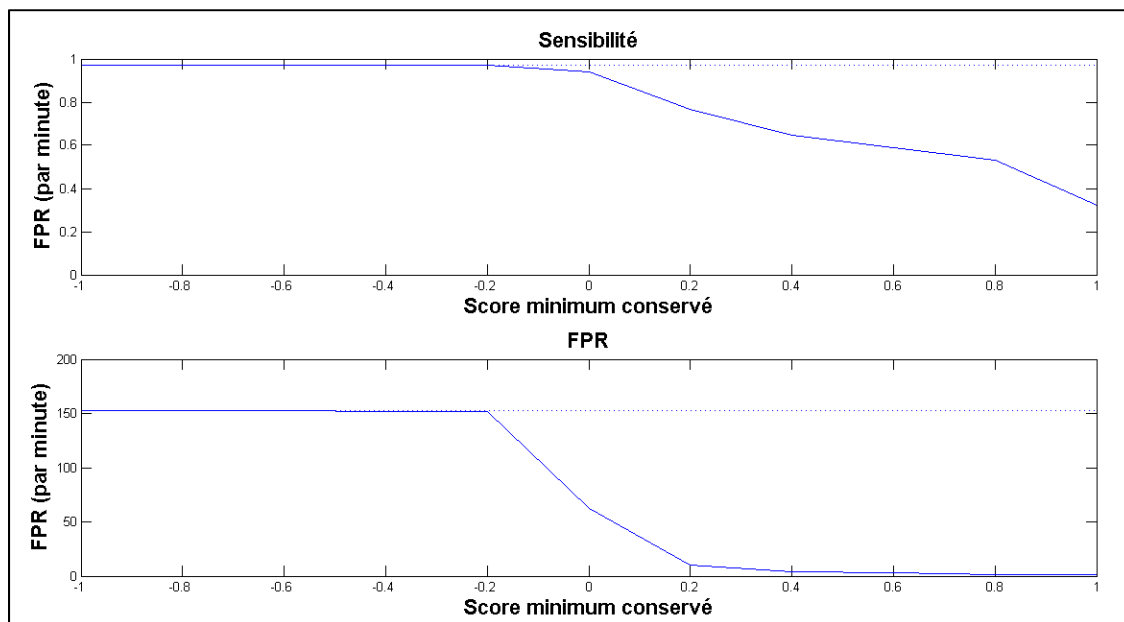


Figure 5.3 Impact du score (CML)

En abscisse, le score minimum conservé. En ordonnée est la sensibilité (haut) et le taux de faux positifs par minute (bas)

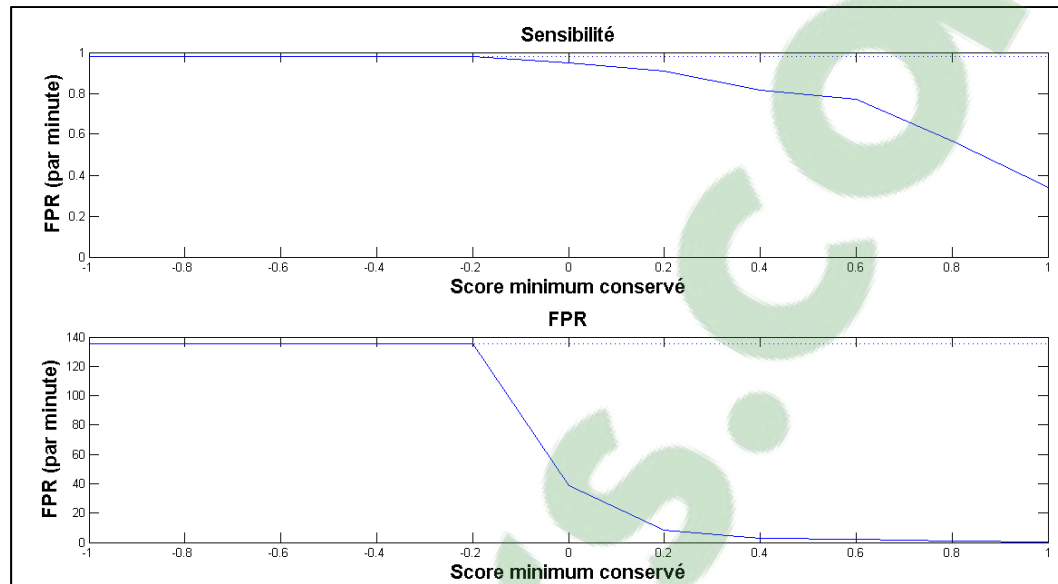


Figure 5.4 Impact du score (MA)
Voir figure 5.3

Les figures 5.5-5.6 démontrent une transition en spécificité en ne conservant que les évènements avec un score positif. La perte en sensibilité à ce niveau est très faible pour l'ensemble des patients. On peut donc en conclure que le système de score permet efficacement d'attribuer un score faible à des faux positifs et un score élevé aux vrais positifs. Certains artefacts se font attribuer un score élevé, principalement ceux qui ont été détectés sur plusieurs canaux dans la zone d'intérêt SOZ. Toutefois **il est préférable de donner un score élevé à un évènement non désiré qu'un score faible à un évènement désiré. En d'autres mots, nous favorisons d'éviter les erreurs de type II par rapport aux erreurs de type I.**

Bien que l'objectif de ce système ne soit pas de retirer des détections, mais simplement de les classer, les résultats sont suffisamment satisfaisants pour suggérer de systématiquement retirer les scores négatifs. Le tableau suivant compile les performances pour tous les patients dans le cas où les évènements sont scorés après une détection avec le système d'apprentissage, et où seulement les scores supérieurs à 0 sont conservés.

Tableau 4 Résultats classification finale

L'encadré contient la performance du détecteur avec apprentissage au niveau de la détection, et celle après avoir retiré les scores négatifs. La sensibilité (sens) est en % et le taux de faux positifs (FPR), est par minute). Les deux gains sont calculés par rapport au détecteur et classifieur initiaux

		Détecteur optimisé			
	Marqueurs	Détection	Gain Détection	Classification Finale (S < 0)	Gain Classification
CML	27 gamma 7 ripple	Sens = 97 FPR = 99	Sens = 6 FPR = -32	Sens = 71 FPR = 12	Sens = 21 FPR = 10
DR	9 gamma 6 ripple	Sens = 53 FPR = 35	Sens = -20 FPR = -59	Sens = 53 FPR = 14	Sens = 20 FPR = 13
JC	43 gamma 5 ripple	Sens = 88 FPR = 32	Sens = -1 FPR = -63	Sens = 88 FPR = 10	Sens = 0 FPR = 5
MA	72 gamma 25 ripple	Sens = 96 FPR = 58	Sens = -3 FPR = -77	Sens = 88 FPR = 6	Sens = -2 FPR = -48
PH	198 gamma 47 ripple	Sens = 87 FPR = 96	Sens = 5 FPR = 4	Sens = 78 FPR = 13	Sens = 28 FPR = 1
PJ	9 gamma 6 ripple	Sens = 78 FPR = 15	Sens = -10 FPR = -118	Sens = 78 FPR = 4	Sens = 23 FPR = 3
SBL	207 gamma 77 ripple	Sens = 89 FPR = 70	Sens = -8 FPR = -85	Sens = 86 FPR = 10	Sens = 5 FPR = -53
SG	135 gamma 33 ripple	Sens = 83 FPR = 37	Sens = -9 FPR = -53	Sens = 73 FPR = 9	Sens = 41 FPR = 3
MOY	88 gamma 26 ripple	Sens = 84 FPR = 57	Sens = -5 FPR = -59	Sens = 77 FPR = 10	Sens = 17 FPR = -8

La sensibilité moyenne est maintenant de 77% avec le taux de faux positifs le plus bas à 10/minute. Avec cette méthodologie, nous proposons donc par rapport au détecteur initial, soit un gain de 17% en sensibilité et de 8 faux positifs/minute de moins. Notons également que la sensibilité est supérieure à 70% pour 7 des 8 sujets, et le FPR est inférieur à 15 pour tous les sujets. Le cas où les performances sont plus faibles, le patient DR, seulement 15 HFOs furent marquées, et plusieurs de ces marqueurs devraient être remis en question.

CONCLUSION

Ce travail a eu pour objectif de faire la détection des oscillations hautes-fréquences sur le signal EEG de scalp de patients épileptiques. Un détecteur automatique de base, avec tous les paramètres fixes pour tous les patients, fut utilisé comme point de départ du projet.

La première contribution fut l'analyse des marqueurs de l'expert. Il fut observé puis démontré que les marqueurs de l'expert de chaque patient ont des caractéristiques d'amplitudes, spectrales et spatiales similaires entre eux. Ces caractéristiques sont toutefois variables d'un patient à l'autre. Nous avons conclu que le détecteur pourrait bénéficier d'une adaptation des paramètres pour chaque patient basée sur leurs caractéristiques intrinsèques observées sur les marqueurs.

La deuxième contribution fut un système d'apprentissage de paramètres à l'aide de marqueurs de l'expert. Ce système d'apprentissage fut le résultat de l'observation des marqueurs de la première contribution. Le système d'apprentissage assure que les paramètres du détecteur sont adaptés à chaque patient pour une détection optimale, au prix de la nécessité d'un sous-ensemble de marqueurs.

La troisième contribution fut un système de score qui aide à évaluer la vraisemblance d'une détection automatique par rapport aux marqueurs de l'expert, basé sur le même sous-ensemble que celui utilisé pour l'apprentissage. Certaines caractéristiques communes furent observées sur les marqueurs qui ne font pas partie de la définition des HFOs. Ces caractéristiques peuvent toutefois être utilisées pour attribuer à chaque détection automatique une valeur qui représente sa ressemblance au sous-ensemble de marqueurs. On note que la corrélation spatiale est utilisée dans l'attribution du score bien que nous ayons préalablement mentionné dans la section 3.5 que nous voulions l'éviter dans la détection pour éviter de recréer le biais spatial de l'expert. La justification est qu'à cette étape, aucune

détection n'est réellement retirée, elles sont simplement scorées, évitant ainsi d'être trop biaisé.

Le résultat du détecteur semi-automatique peut être comparé à celui du détecteur automatique initial et son classifieur, sur une base de données de 8 patients marqués par un expert. Le résultat de la détection en utilisant les paramètres optimisés par le système d'apprentissage offre une forte sensibilité (84%), malgré une perte de 5% par rapport au détecteur initial, et un gain important en spécificité, passant de 116 faux positifs par minutes à 57. En ne considérant ensuite que les détections avec un score positif, **la performance du détecteur proposé dans ce travail est supérieure en sensibilité et en spécificité par rapport au détecteur initial**. En effet, la sensibilité est de 77%, un gain de 16% par rapport au résultat final du détecteur et classifieur initial. La spécificité est quant à elle de 10 par minute, une amélioration de 8 par minute.

En conclusion, en utilisant le détecteur proposé dans cet ouvrage, la détection supervisée de HFO est améliorée en sensibilité et en spécificité, et offre une performance similaire à celle de l'expert en une fraction du temps. Pour y parvenir, l'expert doit fournir un sous-ensemble de marqueurs contenus dans 10 minutes d'enregistrement, ce qui reste un gain de temps considérable étant donné la détection visuelle difficile des HFOs sur le scalp.

En plus de l'information contenue dans ce mémoire est livré un logiciel MATLAB permettant la détection et la classification des HFOs pour un patient. Tous les algorithmes ayant permis l'analyse des marqueurs, la visualisation des résultats et le calcul de performance sont également disponibles sur demande.

RECOMMANDATIONS

Pour la suite de la recherche sur la détection des HFOs de scalp des patients épileptiques, il serait intéressant d'essayer de nouvelles avenues. À mesure où notre compréhension des HFOs va augmenter, et que leur définition officielle deviendra plus claire, le détecteur pourra augmenter en performance au même rythme. Les caractéristiques spatiales et la proximité aux pointes pourraient être incorporées à la détection pour une meilleure spécificité, au lieu d'être simplement au niveau du score.

Il serait également intéressant de repasser sur les détections automatiques avec un expert pour apprendre des détections différentes des marqueurs initiaux. Si certaines fausses détections s'avèrent être fréquentes, une étape supplémentaire de suppression de ce type de détection pourrait être ajoutée. Dans la même idée, il serait utile de se pencher sur les marqueurs qui ne sont pas actuellement trouvés par le détecteur, et réfléchir une méthodologie pour augmenter la sensibilité.

Au niveau du système de score, le poids de chaque critère pourrait être fixé de manière moins ad-hoc. Une approche Bayésienne pourrait permettre de trouver une valeur optimale pour chaque critère qui serait moins empirique et mieux validée.

Notons également que la nouvelle méthode a été testée sur les données avec lesquelles la méthode a été développée.

Dans tous les cas, comme le démontre ce travail, l'union des compétences entre l'ingénierie et la médecine peut permettre de bien meilleurs résultats que le travail individuel de ces deux expertises.

ANNEXE I

CODE MATLAB – CLASSE SUBJECT_HFO

```
% This class represents a subject, with all parameters extracted from the  
% Stellate file. It contains all labels and markers, and the general  
% recording parameters. It also contain a Detector object, which contains  
% the automatic detector.
```

```
classdef subject_HFO
```

```
    %import HFO_detector.*
```

```
    properties
```

```
        SigFileName ;    % Name of the file  
        MontageName ;    % Name of the montage  
        fs;              % Sampling frequency  
        ChnLabel;        % Name of each channels  
        n_total;         % Number of samples  
  
        nb_gamma;        % Number of HFO gamma markers  
        nb_ripple;       % Number of HFO ripple markers  
        nb_spike;        % Number of spike markers  
  
        PropHeader;      % Marker header (all markers)  
        StatusItems_Gamma; % HFO gamma markers  
        Channels_Gamma;   % HFO gamma marker channels (name)  
        Channels_Gamma_list; % HFO gamma marker channels (id)  
  
        StatusItems_Ripple; % HFO ripple markers  
        Channels_Ripple;    % HFO ripple marker channels (name)  
        Channels_Ripple_list; % HFO gamma marker channels (id)  
  
        StatusItems_Spike; % Spikes markers  
        Channels_Spike;    % Spikes marker channels (name)  
        Channels_Spike_list; % Spikes marker channels (id)  
  
        NumSecs;    % Number of seconds total  
        NumSamps;   % Number of samples total  
        eegdata;    % EEG data  
        Detector;   % Detector object
```

```
    end
```

```
    methods
```

```
        function obj = subject_HFO(FileName)  
            % This function receives the file to create a subject_HFO  
            % object. It could be a Stelatte file (requires Stelatte
```

```

% toolbox) or a matlab file from a previously saved subject.

% If the Filename is a string, it is expected to be the name
of
% the Stallate file
if isstr(Filename)
    % Assing values to the properties of the object.
    mFileOpen(Filename);
    obj.SigFileName = Filename;
    obj.MontageName = 'Zygo-Db-Temp.inf';
    obj.fs=mGetTrueSampFreq(Filename);
    obj.ChnLabel=mGetChanLabel(Filename,obj.MontageName);
    [AAA,obj.n_total]=mGetFileLength(Filename);
    [obj.PropHeader, obj.StatusItems_Ripple,
obj.Channels_Ripple] ...
        = mGetStatusItems(Filename, 'FM R');
    [obj.PropHeader, obj.StatusItems_Gamma,
obj.Channels_Gamma] ...
        = mGetStatusItems(Filename, 'FM Gamma');
    [obj.PropHeader, obj.StatusItems_Spike,
obj.Channels_Spike] ...
        = mGetStatusItems(Filename, 'FM spike');
    [obj.nb_gamma,B] = size(obj.StatusItems_Gamma);
    [obj.nb_ripple,B] = size(obj.StatusItems_Ripple);
    [obj.nb_spike,B] = size(obj.StatusItems_Spike);
    obj.Channels_Gamma_list = ...

CHNlabel2index(obj.nb_gamma,obj.Channels_Gamma,obj.ChnLabel);
    obj.Channels_Ripple_list = ...

CHNlabel2index(obj.nb_ripple,obj.Channels_Ripple,obj.ChnLabel);
    obj.Channels_Spike_list = ...

CHNlabel2index(obj.nb_spike,obj.Channels_Spike,obj.ChnLabel);
    [B, obj.NumSamps, obj.NumSecs] = mGetFileLength(Filename);
    obj.eegdata = extract_data(obj)
    obj.Detector = c_detector;
    obj.Detector.root = obj;

% If the Filename is a structure, it is expected to be from a
previoulsy saved subject (see save_subject below)
elseif isstruct(Filename)
    subject = Filename;
    obj.SigFileName = subject.SigFileName;
    obj.MontageName = subject.MontageName;
    obj.fs=subject.fs;
    obj.ChnLabel=subject.ChnLabel;
    obj.n_total=subject.n_total;
    obj.PropHeader = subject.PropHeader;
    obj.StatusItems_Ripple = subject.StatusItems_Ripple;
    obj.Channels_Ripple = subject.Channels_Ripple;
    obj.StatusItems_Gamma = subject.StatusItems_Gamma;
    obj.Channels_Gamma = subject.Channels_Gamma;

```

```

obj.StatusItems_Spike = subject.StatusItems_Spike
obj.Channels_Spike = subject.Channels_Spike
obj.nb_gamma = subject.nb_gamma;
obj.nb_ripple = subject.nb_ripple;
obj.nb_spike = subject.nb_spike;
obj.Channels_Gamma_list = subject.Channels_Gamma_list;
obj.Channels_Ripple_list = subject.Channels_Ripple_list;
obj.Channels_Spike_list = subject.Channels_Spike_list;
obj.NumSecs = subject.NumSecs;
obj.NumSamps = subject.NumSecs*subject.fs;
obj.eegdata = subject.eegdata;
obj.Detector = c_detector;
obj.Detector.root = obj;

obj.Detector.events_predetection = ...
    subject.events_predetection;
obj.Detector.channels_predetection = ...
    subject.channels_predetection;
obj.Detector.events_classification = ...
    subject.events_classification;
obj.Detector.sensitivity_predetection = ...
    subject.sensitivity_predetection;
obj.Detector.FPR_predetection = ...
    subject.FPR_predetection;
obj.Detector.sensitivity_classification = ...
    subject.sensitivity_classification;
obj.Detector.FPR_classification = ...
    subject.FPR_classification;
obj.Detector.umb = ...
    subject.umb;
obj.Detector.events_clustered = ...
    subject.events_clustered;
obj.Detector.optimized_thresholds = ...
    subject.optimized_thresholds;
obj.Detector.optimized_spatial = ...
    subject.optimized_spatial;
obj.Detector.cooccurrence = ...
    subject.cooccurrence;
obj.Detector.cooccurrence_header = ...
    ['spike-gamma spike_ripple gamma_spike ripple_spike'];
end

end

% This function save the object into a matlab file, which can be
% used a an input parameter for the previous function
function save_subject(obj);
    subject.SigFileName = obj.SigFileName;
    subject.MontageName = obj.MontageName ;
    subject.fs = obj.fs;
    subject.ChnLabel = obj.ChnLabel;
    subject.n_total = obj.n_total;

```

```

subject.nb_ripple = obj.nb_ripple;
subject.nb_gamma = obj.nb_gamma;
subject.nb_spike = obj.nb_spike;

subject.PropHeader = obj.PropHeader;
subject.StatusItems_Ripple = obj.StatusItems_Ripple;
subject.Channels_Ripple = obj.Channels_Ripple;
subject.StatusItems_Gamma = obj.StatusItems_Gamma;
subject.Channels_Gamma = obj.Channels_Gamma;
subject.StatusItems_Spike = obj.StatusItems_Spike;
subject.Channels_Spike = obj.Channels_Spike;

subject.Channels_Gamma_list = obj.Channels_Gamma_list;
subject.Channels_Ripple_list = obj.Channels_Ripple_list;
subject.Channels_Spike_list = obj.Channels_Spike_list;

subject.NumSecs = obj.NumSecs;
subject.NumSamps = subject.NumSecs*subject.fs;
subject.eegdata = obj.eegdata;

subject.events_predetection = ...
    obj.Detector.events_predetection;
subject.channels_predetection = ...
    obj.Detector.channels_predetection;
subject.events_classification = ...
    obj.Detector.events_classification;
subject.sensitivity_predetection = ...
    obj.Detector.sensitivity_predetection;
subject.FPR_predetection = ...
    obj.Detector.FPR_predetection;
subject.sensitivity_classification = ...
    obj.Detector.sensitivity_classification;
subject.FPR_classification = ...
    obj.Detector.FPR_classification;
subject.umb = ...
    obj.Detector.umb;
subject.events_clustered = ...
    obj.Detector.events_clustered;
subject.optimized_spatial = ...
    obj.Detector.optimized_spatial;
subject.optimized_thresholds = ...
    obj.Detector.optimized_thresholds;
subject.cooccurrence = ...
    obj.Detector.cooccurrence;

disp('saving...');
pause(0.5);
save(['struct_',obj.SigFileName,'_spike'],'subject');
disp('done');

end

```

```
        %end methods
    end

end
```


ANNEXE II

CODE MATLAB : CLASSE C_DETECTOR

```
% This class contains the detector parameters and results.
classdef c_detector

    properties
        % Algorithm parameters
        % Minimum separation between different events [seconds].
        Nove=.1;
        % Background moving window length [seconds].
        wl=1;
        % Detection Threshold (RMS of background times this value).
        umb=2.5;
        % Chunk size (file is processed by parts to avoid excessive
        % memory usage).
        n=200000;
        % Subject_HFO is saved in the root
        root;

        events_predetection; % Events detected
        channels_predetection; % Event's channels (label)
        sensitivity_predetection; % Sensitivity after detection
        FPR_predetection; % False positive rate after detection

        events_classification; % Events scored (low scores removed)
        sensitivity_classification; % Sensitivity after classification
        FPR_classification; % False positive rate after classification

        events_clustered; % Not currently used, saved for
                        % future clustering filtering method
        optimized_thresholds; % Contains optimized thresholds
        optimized_spatial; % not currently used, saved for
        cooccurrence_header = ...
            ['spike-gamma spike_ripple gamma_spike ripple_spike'];
        cooccurrence;
    end

    methods

        function obj = c_detector(varargin)
            % Assing values to the properties of the object.
            obj.events_predetection = [];
            obj.channels_predetection = [];
            obj.sensitivity_predetection = 0;
            obj.FPR_predetection = 0;
            obj.sensitivity_classification = 0;
        end
    end
end
```

```

obj.FPR_classification = 0;
obj.events_classification = [];
obj.events_clustered = [];
obj.cooccurrence = [];
obj.optimized_spatial = [];

end

% This is the code for this initial detector, and credits are given to
% Nicolas von Ellenrieder - 2011/01/19 (nellen@ieee.org). The list
% of modifications are the following:
% - Replaced the input-output with a HFDetector object
% - Changed most variable names to make easy to read
% - EEG data (root) is exempt of excluded channels
% - Detection threshold (umb) is a vector to reflect the optimized value
%   required for each narrowband. If the vector has a length of 1, this
%   value is used for each narrowband (no optimization)
% - Initial classification method is still used, but with a very sensitive
%   setting. False positives will be removed using a scoring method.
function obj = HFDetector(obj)

SigFileName = obj.root.SigFileName;
MontageName = obj.root.MontageName;
ExcludedChannels = [];
fs = obj.root.fs;
ChnLabel = obj.root.ChnLabel;
n_total = obj.root.n_total;

%
% First stage of automatic HFO detector for scalp recordings.
% This detector is designed for Fast Gamma and Ripple bands
% (40 - 200 Hz) and a 600 Hz sampling rate, for other uses it
% should be modified. For a detailed explanation of the algorithm
% see:
%
%
% Outputs:
% event: a list of HFO candidates. One row per HFO candidates.
%       Column 1: Channel number.
%       Column 2: Approximate frequency (10 Hz resolution).
%       Column 3: Start (in samples).
%       Column 4: Length (in samples).
%       Column 5: Maximum RMS value of the signal amplitude in the
%                 specified channel, frequency band, and time
%                 interval.
%       Column 6: Maximum narrowband (10 Hz bandwidth) signal
%                 amplitude in the specified channel, frequency
%                 band, and time interval.
%       Column 7: Maximum broadband (40 - 200 Hz) signal amplitude
%                 in the specified channel, frequency band, and
time

```

```

%           interval.
%       Column 8: Channel event identifier: all the events with the
%               same number correspond to a single event of the
%               channel level detector.
%       Column 9: Subject level identifier: all the events with the
%               same number correspond to a single event of the
%               subject level detector.
% ChnLabel: A cell array with the name of the channels.

% Algorithm parameters
% Minimum separation between different events [seconds]
% suggested value = (between 25 and 100ms).
Nove=obj.Nove;
% Background moving window length [seconds]
% suggested value = (between 10s y 60s).
wl=obj.wl;
% Threshold (RMS of background times this value)
% suggested value = (between 2.25 and 2.75).
umb2=obj.umb;
% Chunk size (file is processed by parts to avoid
% excessive memory usage).
n=obj.n;

% Classification parameters
% Since this classification method is considered not sensitive
% enough for our purposes, the parameters have been set to a very
% sentivite set of thresholds. Only the most obvious false
% positives will be removed with this method.
events_classification = [];
THA = 0.3;
THB = 6;

% Exclude the indicated channels (see remove_channels function)
nb_CHN = length(ChnLabel);
for ii = 1 : length(ChnLabel)
    if strcmp(ChnLabel{ii}, 'REMOVED')
        nb_CHN = nb_CHN-1;
    end
end
NumChn=1:nb_CHN;
if nargin==2, ExcludedChannels=[]; end
if ~isempty(ExcludedChannels)
if iscell(ExcludedChannels)
    a=zeros(length(Chnlabels),1)>0;
    for ii=1:length(ExcludedChannels)
        e=ExcludedChannels{ii};
        a=a|~isempty(strfind(e,ChnLabel));
    end
    ExcludedChannels=a;
end
if islogical(ExcludedChannels)
    if length(ExcludedChannels)==length(ChnLabel)

```

```

        ExcludedChannels=find(ExcludedChannels);
    else
        disp('Logical ExcludedChannels should be the same length
as the number of channels.');
```

as the number of channels.');

```

    end
end
if isnumeric(ExcludedChannels)
    NumChn(ExcludedChannels)=[]; ChnLabel(ExcludedChannels)=[];
else
    disp('ExcludedChannels should be a cell array, an integer
array or a logical array.');
```

array or a logical array.');

```

    return
end
end

% Number of valid channels
Nch=length(NumChn);

% Modify parameters
Nove=round(Nove*fs);
wl=round(wl*fs);
wb=waitbar(0, 'Processing...');

% Filters (one general bandpass filter and the 10Hz band filters)
% General bandpass filter
hbp=firpm(120,[0 25 35 205 215 fs/2]*2/fs,[0 0 1 1 0 0]);
% Narrowband filters order and central frequency
FO=300; f=(45:10:195)/fs; nf=length(f);
% Design narrowband filters
h=zeros(nf,FO+1);
for iter_freq=1:nf
    h(iter_freq,:)=firpm(FO,[0 f(iter_freq)-10/fs f(iter_freq)...
        -5/fs f(iter_freq)+5/fs f(iter_freq)+10/fs .5]*2,...
        [0 0 1 1 0 0],[10 1 1]);
end
% Minimum length of the events (narrowband filter effective
% duration plus four cycles of the band central frequency.
lmin=sqrt(sum(repmat((-FO/2:FO/2).^2,nf,1).*h.^2,2)./...
    sum(h.^2,2))+4./f';
% Initializations
event=zeros(50000,9);
LastSamp=n; SigOffset=FO; IniSamp=1; continue_flag=true;
evt_cnt=0; egr_cnt=0; sgr_cnt=0;
while continue_flag
    window_detection=LastSamp-IniSamp+1;
    raw_data = read_data(obj.root,IniSamp,window_detection);

    % Exclude indicated channels
    raw_data=raw_data(:,NumChn);

    %
    spatial_modifier = obj.events_variance;
    %
    spatial_modifier(8) = [];

```

```

% For each channel
for iter_chn=1:Nch
    % Bandpass filtering
    broadband=filter(hbp,1,[raw_data(:,iter_chn)' ...
        zeros(1,(length(hbp)-1)/2)]);
    broadband(1:(length(hbp)-1)/2)=[];
    % For each frequency band
    for iter_freq=1:nf
        % Varies threshold to be more specific on
        % higher frequencies
        if length(umb2) == 16
            umb = umb2(iter_freq);
        else
            umb = umb2;
        end

        % Narrow band signal
        narrowband=[filter(h(iter_freq,:),1,broadband)
zeros(1,FO/2)];

        narrowband(1:FO/2)=[];
        % Compute RMS (squared) value in 4 cycles window
        d=floor(2/f(iter_freq));
        do=ones(1,2*d+1)/(2*d+1);
        RMS_narrowband = sqrt([filter(do,1,narrowband.^2)
zeros(1,d)]);

        RMS_narrowband(1:d)=[];
        % Compute RMS of background (moving window)
        m=RMS_narrowband; m(1:wl)=mean(RMS_narrowband(1:wl));
        mm=RMS_narrowband;
        for jj=wl+1:window_detection
            mm(jj)=min(RMS_narrowband(jj-d),umb*m(jj-d));
            m(jj)=m(jj-1)+(mm(jj)-mm(jj-wl))/wl;
        end
        % First detections
        detection_flags=(RMS_narrowband>umb*m);
        %
        ratio_umb = RMS_narrowband./m;
        % Find begining and end of detections
        detection_flags(1)=0;
        detection_flags(window_detection-
1:window_detection)=0;
        start_flags=find(detection_flags(2:end)~=0&...
            detection_flags(1:end-1)==0)+1;
        end_flags=find(detection_flags(2:end)==0&...
            detection_flags(1:end-1)~=0);
        % Keep only detections long enough (and not already
        % detected in another data chunk)
        len=end_flags-start_flags+1;
        id=len<lmin(iter_freq)|start_flags<=SigOffset;
        start_flags(id)=[]; len(id)=[]; end_flags(id)=[];
        % Compute features
        ne=length(start_flags); o=ones(ne,1); val=[o o o];
        for jj=1:ne

```

```

        val(jj,:)=[max(RMS_narrowband(start_flags(jj):...
            end_flags(jj))) ...
            max(abs(narrowband(start_flags(jj):...
            end_flags(jj)))) max(abs(broadband...
            (start_flags(jj):end_flags(jj))))];
    end
    % Add detections to events
    e=[iter_chn*o (iter_freq*10+35)*o ...
        start_flags(:)+IniSamp-1 len(:) val 0*[o o]];
    event(evt_cnt+(1:ne),:)=e;
    %     event(evt_cnt+(1:ne),10) = ratio_umb(start_flags);
    evt_cnt=evt_cnt+ne;
end
% Combine events of different frequency and same channel

e=event(event(:,1)==iter_chn&event(:,3)>=IniSamp+SigOffset,:);
start_flags=e(:,3);
end_flags=e(:,3)+e(:,4)+Nove;
detection_flags=zeros(1,max(end_flags)+1)>0;
for jj=1:size(start_flags,1),
    detection_flags(start_flags(jj):end_flags(jj))=true;
end
start_flags = find(detection_flags(2:end)~=0&...
    detection_flags(1:end-1)==0)+1;
end_flags = find(detection_flags(2:end)==0&...
    detection_flags(1:end-1)~=0);
for iter_det=1:length(start_flags)
    egr_cnt=egr_cnt+1;
    event(event(:,3)>=start_flags(iter_det)&...
        event(:,3)<end_flags(iter_det)&event(:,1)...
        ==iter_chn,8)=egr_cnt;
end
waitbar((IniSamp-1)/n_total>window_detection/...
    n_total*iter_chn/Nch,wb);
end
% Decide if the end of the file has been reached, if not
select next
% data chunk
if LastSamp<n_total
    SigOffset=wl+FO;
    IniSamp=LastSamp-SigOffset+1;
    LastSamp=min(LastSamp+n,n_total);
else
    continue_flag=false;
end
end
event=event(1:evt_cnt,:);
% Combine events of different channels and frequencies
start_flags=event(:,3);
end_flags=event(:,3)+event(:,4)+Nove;
detection_flags=zeros(1,max(end_flags)+1)>0;
for jj=1:size(start_flags,1),
    detection_flags(start_flags(jj):end_flags(jj))=true;

```

```

end
start_flags=find(detection_flags(2:end)~=0&...
    detection_flags(1:end-1)==0)+1;
end_flags=find(detection_flags(2:end)==0&...
    detection_flags(1:end-1)~=0);
for iter_det=1:length(start_flags)
    sgr_cnt=sgr_cnt+1;
    event(event(:,3)>=start_flags(iter_det)&...
        event(:,3)<=end_flags(iter_det),9)=sgr_cnt;
end

% Remove the tstart offset of 2 detection windows
freq = event(:,2);
offset = round(2./freq.*fs);
event(:,3) = event(:,3) + offset;

% find if any offsetted events goes beyond data range
data_error = n_total - (event(:,3) + event(:,4));
id_error = find(data_error < 0);
events(id_error,3) = n_total - event(id_error,4) - 1;

obj.events_predetection = event;

% 1rst Classification
threshold = [THA THB];

% Select events with appropriate features values
in=event(:,5)>=threshold(1)&(event(:,7)./event(:,6))<=threshold(2);
event=event(in,:);
% Exclude all subject level events with at least one individual
% level with RMS amplitude of the broadband signal greater than
% 50 uV. This only deletes the most obvious artifacts and it will
% never delete a true positive. Used only for cosmetics of the
% results, not really to improve the performance, there are very
% few events in this category.
in=event(:,1)<0;
ix=unique(event(event(:,7)>=50,9));
for ii=1:length(ix)
    in=in|event(:,9)==ix(ii);
end
event(in,:)=[];
% Renumber channel level events
ix=unique(event(:,8)); ev=event;
for ii=1:length(ix), event(ev(:,8)==ix(ii),8)=ii; end
% Renumber subject level events
ix=unique(event(:,9)); ev=event;
for ii=1:length(ix), event(ev(:,9)==ix(ii),9)=ii; end

obj.events_classification = event;
obj.channels_predetection = ChnLabel;

```

```

obj = synch_detections(obj);
close(wb);
end

function obj= synch_detections(obj)
% This function assures that the channels tags for the markers
% and the detections are the same

channel_list = obj.channels_predetection;
events_predetection = obj.events_predetection;

iter_removed = 1;
for iter_chn = 1 : 32
    if strcmp('REMOVED',channel_list{iter_chn}) == 1
        id_removed(iter_removed) = iter_chn;
    end
end

nb_removed = iter_removed;
for iter_removed = 1 : nb_removed
    id_revision = find(events_predetection(:,1)>= ...
        id_removed(iter_removed));
    events_predetection(id_revision,1) = ...
        events_predetection(id_revision,1) +1;
end

obj.events_predetection = events_predetection;

end

end

end

```

ANNEXE III

CODE MATLAB – APPRENTISSAGE DU SEUIL DE DÉTECTION

```
% This function finds the optimized values of the detector threshold for
% a subject based on a subsample of markers (created in this function).
% Input:      obj = subject_HFO object
%            method = vector of length 2 for fix number of markers OR
%            a int value for the interval (in min)

function obj = optimize_umb(obj, method);

% Finds a random set of markers. The ammount chosen are based on the
% method input [num_gamma num_ripple]
if length(method) == 2
    Num_gamma = method(1);
    Num_ripple = method(2);
    Num_gamma = min([Num_gamma length(obj.Channels_Gamma)]);
    Num_ripple = min([Num_ripple length(obj.Channels_Ripple)]);

    Gamma_expert = obj.nb_gamma;
    Ripple_expert = obj.nb_ripple;

    id_event_Gamma = randperm(Gamma_expert);
    id_event_Gamma = id_event_Gamma(1:Num_gamma);
    id_event_Ripple = randperm(Ripple_expert);
    id_event_Ripple = id_event_Ripple(1:Num_ripple);

% Create the subsample of every marker contained in a time interval in
% minutes = method input. The window is randomized.
else
    fs = obj.fs;
    n_total = obj.n_total;
    window_length = method * fs * 60;
    id_event_Gamma = [];
    id_event_Ripple = [];
    while length(id_event_Gamma) == 0 || length(id_event_Ripple) == 0;
        window_min = round((n_total - window_length)*rand(1));
        window_max = window_min + window_length;

        id_event_Gamma1 = find(obj.StatusItems_Gamma(:,2) < window_max);
        id_event_Gamma2 = find(obj.StatusItems_Gamma(:,2) > window_min);
        id_event_Gamma = ismember(id_event_Gamma1, id_event_Gamma2);
        id_event_Gamma = id_event_Gamma1(id_event_Gamma);
        clear id_event_Gamma1
        clear id_event_Gamma2

        id_event_Ripple1 = find(obj.StatusItems_Ripple(:,2) < window_max);
        id_event_Ripple2 = find(obj.StatusItems_Ripple(:,2) > window_min);
```

```

    id_event_Ripple = ismember(id_event_Ripple1,id_event_Ripple2);
    id_event_Ripple = id_event_Ripple1(id_event_Ripple);
    clear id_event_Ripple1
    clear id_event_Ripple2

    Num_ripple = length(id_event_Ripple);
    Num_gamma = length(id_event_Gamma);
    sensitivity_buffer = 1/(Num_gamma + Num_ripple);
end

end

% These functions return the optimized value for both low and high
% frequencies. See below for details
umb_gamma = compute_umb(obj, id_event_Gamma, []);
umb_ripple = compute_umb(obj, [], id_event_Ripple);

% The detection threshold is interpolated between both thresholds for
% each narrowbands
obj.Detector.umb = linspace(umb_gamma,umb_ripple,16)

end

function umb = compute_umb(obj, id_event_Gamma, id_event_Ripple);
[Gamma_expert A]= size(obj.StatusItems_Gamma);
[Ripple_expert A]= size(obj.StatusItems_Ripple);
umb_threshold = 0.9;
fs = obj.fs;

% Filters (one general bandpass filter and the 10Hz band filters)
% General bandpass filter
hbp=firpm(120,[0 25 35 205 215 fs/2]*2/fs,[0 0 1 1 0 0]);
% Narrowband filters order and central frequency
FO=300; f=(45:10:195)/fs; nf=length(f);
% Design narrowband filters
h=zeros(nf,FO+1);
for ii=1:nf
    h(ii,:)=firpm(FO,[0 f(ii)-10/fs f(ii)-5/fs f(ii)+5/fs f(ii)+10/fs
    .5]*2,[0 0 1 1 0 0],[10 1 1]);
end
% Minimum length of the events (narrowband filter effective duration plus
% four cycles of the band central frequency.
lmin=sqrt(sum(repmat((-FO/2:FO/2).^2,nf,1).*h.^2,2)./sum(h.^2,2))+4./f';
det_filter.h = h;
det_filter.hbp = hbp;
det_filter.lmin = lmin;
det_filter.FO = FO;
det_filter.nf = nf;
det_filter.f = f;

umb = 2

```

```

    umb_jump = 0.1;
    subdata = create_subdata(obj, id_event_Gamma, id_event_Ripple);
    events_predetection = partial_detection(obj, subdata, umb, det_filter);
    nb_expert_gamma = length(subdata.id_event_Gamma);
    nb_expert_ripple = length(subdata.id_event_Ripple);
    nb_expert_total = nb_expert_gamma + nb_expert_ripple;
    [sens_reference freq] =
perfo_partial(obj, events_predetection, subdata);
    mysensitivity = sens_reference;
    while mysensitivity >= umb_threshold
        umb = umb+umb_jump
        events_predetection =
partial_detection(obj, subdata, umb, det_filter);
        mean_ratio = mean(events_predetection(:,8))
        [mysensitivity freq] =
perfo_partial(obj, events_predetection, subdata);
    end
    umb = umb-umb_jump;

end

% This function is very similar the the HFDetector function (see Annex II)
% the window of detection is restrained to the subsample. Again, some
% credit is given to Nicolas von Ellenrieder - 2011/01/19
% (nellen@ieee.org)
% for creating the initial detector this is based on.
function events_predetection =
partial_detection(obj, subdata, umb, det_filter);

    SigFileName = obj.SigFileName;
    MontageName = obj.MontageName;
    ExcludedChannels = [];
    fs = obj.fs;
    ChnLabel = obj.ChnLabel;
    n_total = length(subdata.data_samples);

    Nove=obj.Detector.Nove;
    wl=obj.Detector.wl;
    n=n_total;

    % Classification parameters
    events_classification = [];
    THA = 0.3;
    THB = 6;

    % Exclude the indicated channels
    nb_CHN = length(ChnLabel);
    for ii = 1 : length(ChnLabel)
        if strcmp(ChnLabel{ii}, 'REMOVED')
            nb_CHN = nb_CHN-1;
        end
    end
end

```

```

NumChn=1:nb_CHN;
%   NumChn=1:length(ChnLabel);
if nargin==2, ExcludedChannels=[]; end
if ~isempty(ExcludedChannels)
if iscell(ExcludedChannels)
    a=zeros(length(ChnLabels),1)>0;
    for ii=1:length(ExcludedChannels)
        e=ExcludedChannels{ii};
        a=a|~isempty(strfind(e,ChnLabel));
    end
    ExcludedChannels=a;
end
if islogical(ExcludedChannels)
    if length(ExcludedChannels)==length(ChnLabel)
        ExcludedChannels=find(ExcludedChannels);
    else
        disp('Logical ExcludedChannels should be the same length
as the number of channels. ');
    end
end
if isnumeric(ExcludedChannels)
    NumChn(ExcludedChannels)=[]; ChnLabel(ExcludedChannels)=[];
else
    disp('ExcludedChannels should be a cell array, an integer
array or a logical array. ');
    return
end
end
% Number of valid channels
Nch=length(NumChn);
% Modify parameters
Nove=round(Nove*fs);
wl=round(wl*fs);
%   wb=waitbar(0,'Processing...');

h    = det_filter.h;
hbp  = det_filter.hbp;
lmin = det_filter.lmin;
FO   = det_filter.FO;
nf   = det_filter.nf;
f    = det_filter.f;

% Initializations
event=zeros(50000,9);
LastSamp=n; SigOffset=FO; IniSamp=1; continue_flag=true;
evt_cnt=0; egr_cnt=0; sgr_cnt=0;
while continue_flag
    window_detection=LastSamp-IniSamp+1;
%
    X = subdata.signal;

    % For each channel

```

```

for nc=1:Nch
    % Bandpass filtering
    x=filter(hbp,1,[X(:,nc)' zeros(1,(length(hbp)-1)/2)]);
    x(1:(length(hbp)-1)/2)=[];
    % For each frequency band
    for ii=1:nf
        % Narrow band signal
        y=[filter(h(ii,:),1,x) zeros(1,FO/2)]; y(1:FO/2)=[];
        % Compute RMS (squared) value in 4 cycles window
        d=floor(2/f(ii));
        do=ones(1,2*d+1)/(2*d+1);
        z=sqrt([filter(do,1,y.^2) zeros(1,d)]); z(1:d)=[];
        % Compute RMS of background (moving window)
        m=z; m(1:wl)=mean(z(1:wl)); mm=z;
        for jj=wl+1:window_detection, mm(jj)=min(z(jj-d),...
            umb*m(jj-d)); m(jj)=m(jj-1)...
            +(mm(jj)-mm(jj-wl))/wl;
        end
        % First detections
        s=(z>umb*m);
        ratio_umb = z./m;
        % Find beginning and end of detections
        s(1)=0; s(window_detection-1:window_detection)=0;
        sta=find(s(2:end)~=0&s(1:end-1)==0)+1;
        fin=find(s(2:end)==0&s(1:end-1)~=0);
        % Keep only detections long enough (and not already
        % detected in another data chunk)
        len=fin-sta+1;
        id=len<lmin(ii)|sta<=SigOffset;
        sta(id)=[]; len(id)=[]; fin(id)=[];
        % Compute features
        ne=length(sta); o=ones(ne,1); val=[o o o];
        for jj=1:ne
            val(jj,:)=[max(z(sta(jj):fin(jj)))...
                max(abs(y(sta(jj):fin(jj)))) max(abs(x...
                    (sta(jj):fin(jj))))]);
        end
        % Add detections to events
        e=[nc*o (ii*10+35)*o sta(:)+IniSamp-1 len(:) ...
            val 0*[o o]];
        event(evt_cnt+(1:ne),:)=e;
        event(evt_cnt+(1:ne),8) = ratio_umb(sta);
        evt_cnt=evt_cnt+ne;
    end
    % Combine events of different frequency and same channel
    e=event(event(:,1)==nc&event(:,3)>=IniSamp+SigOffset,:);
    sta=e(:,3);
    fin=e(:,3)+e(:,4)+Nove;
    s=zeros(1,max(fin)+1)>0;
    for jj=1:size(sta,1), s(sta(jj):fin(jj))=true; end
    sta=find(s(2:end)~=0&s(1:end-1)==0)+1;
    fin=find(s(2:end)==0&s(1:end-1)~=0);
    for ii=1:length(sta)

```

```

        egr_cnt=egr_cnt+1;
        event(event(:,3)>=sta(ii)&event(:,3)<fin(ii)&...
            event(:,1)==nc,8)=egr_cnt;
    end
end
% Decide if the end of the file has been reached, if not
% select next data chunk
if LastSamp<n_total
    SigOffset=wl+FO;
    IniSamp=LastSamp-SigOffset+1;
    LastSamp=min(LastSamp+n,n_total);
else
    continue_flag=false;
end
end
event=event(1:evt_cnt,:);
% Combine events of different channels and frequencies
sta=event(:,3);
fin=event(:,3)+event(:,4)+Nove;
s=zeros(1,max(fin)+1)>0;
for jj=1:size(sta,1), s(sta(jj):fin(jj))=true; end
sta=find(s(2:end)~=0&s(1:end-1)==0)+1;
fin=find(s(2:end)==0&s(1:end-1)~=0);
for ii=1:length(sta)
    sgr_cnt=sgr_cnt+1;
    event(event(:,3)>=sta(ii)&event(:,3)<=fin(ii),9)=sgr_cnt;
end
events_predetection = event;

% 1rst Classification
threshold = [THA THB];

% Select events with appropriate features values

in=event(:,5)>=threshold(1)&(event(:,7)./event(:,6))<=threshold(2);
event=event(in,:);
% Exclude all subject level events with at least one individual
% level with RMS amplitude of the broadband signal greater than
% 50 uV. This only deletes the most obvious artifacts and it will
% never delete a true positive. Used only for cosmetics of the
% results, not really to improve the performance, there are
% very few events in this category.
in=event(:,1)<0;
ix=unique(event(event(:,7)>=50,9));
for ii=1:length(ix)
    in=in|event(:,9)==ix(ii);
end
event(in,:)=[];
% Renumber channel level events
ix=unique(event(:,8)); ev=event;
for ii=1:length(ix), event(ev(:,8)==ix(ii),8)=ii; end
% Renumber subject level events
ix=unique(event(:,9)); ev=event;

```

```

        for ii=1:length(ix), event(ev(:,9)==ix(ii),9)=ii; end

end

% This functions compute the performance (sensitivity and FPR) for the
% detection of the subsample. When the sensitivity drops too low, the
% optimization ends.
function [sens_reference freq] = ...
    perfo_partial(obj,events_predetection,subdata);
Nd = 0;
FN = 0;
freq = 0;
id_event_Gamma = subdata.id_event_Gamma;
id_event_Ripple = subdata.id_event_Ripple;
stop_buffer = subdata.stop_buffer;
start_buffer = subdata.start_buffer;
duration = stop_buffer + start_buffer;
events = events_predetection;
event_start = subdata.data_samples(events(:,3));
[nb_detection A] = size(events);

for iter_expert=1:length(id_event_Ripple)+length(id_event_Gamma)
    current_detection = subdata.data_samples(1+(iter_expert -
1)*duration:duration+(iter_expert -1)*duration);
    current_detection(1:start_buffer)=[];
    current_detection(251:end)=[];
    iter_events = 1;
    flag_detection = 0;
    while iter_events <= nb_detection && flag_detection == 0
        current_event =
event_start(iter_events):(event_start(iter_events)+events(iter_events,4));
        flag_detection = sum(ismember(current_detection,current_event));
        iter_events = iter_events+1;
    end

    if flag_detection == 0
        FN = FN+1;
    end
    if flag_detection > 0
        Nd = Nd+1;
        SLE = find(events(iter_events-1,9) == events(:,9));
        freq(iter_expert) = mode(events(SLE,2));
    end
end

sens_reference = Nd / (length(id_event_Ripple) + length(id_event_Gamma));
end

```



```

% This function adjust the score based on the spatial criteria
function obj = evaluate_good_location(obj,scoring_spatial);
events = obj.Detector.events_classification;
nb_min = 10;

fs = obj.fs;
n_total = obj.n_total;
window_length = nb_min * fs * 60;
id_event_Gamma = [];

while length(id_event_Gamma) == 0 || length(id_event_Ripple) == 0;
    window_min = round((n_total - window_length)*rand(1));
    window_max = window_min + window_length;

    id_event_Gamma1 = find(obj.StatusItems_Gamma(:,2) < window_max);
    id_event_Gamma2 = find(obj.StatusItems_Gamma(:,2) > window_min);
    id_event_Gamma = ismember(id_event_Gamma1,id_event_Gamma2);
    id_event_Gamma = id_event_Gamma1(id_event_Gamma);
    clear id_event_Gamma1
    clear id_event_Gamma2

    id_event_Ripple1 = find(obj.StatusItems_Ripple(:,2) < window_max);
    id_event_Ripple2 = find(obj.StatusItems_Ripple(:,2) > window_min);
    id_event_Ripple = ismember(id_event_Ripple1,id_event_Ripple2);
    id_event_Ripple = id_event_Ripple1(id_event_Ripple);
    clear id_event_Ripple1
    clear id_event_Ripple2

    Num_ripple = length(id_event_Ripple);
    Num_gamma = length(id_event_Gamma);
end

chn_gamma = [];
for iter_gamma = 1:Num_gamma
    chn_gamma = [chn_gamma
obj.Channels_Gamma_list{id_event_Gamma(iter_gamma)}];
end
chn_gamma = unique(chn_gamma);

chn_ripple = [];
for iter_ripple = 1:Num_ripple
    chn_ripple = [chn_ripple
obj.Channels_Ripple_list{id_event_Ripple(iter_ripple)}];
end
chn_ripple = unique(chn_ripple);

chn_SOZ = unique([chn_gamma chn_ripple]);

mat_voisin = abs(neighbors_mat(obj,2));

```

```

voisins = sum(mat_voisin(chn_SOZ,:),1)>0;
voisins = find(voisins == 1);
nb_voisin = length(voisins);

nb_SLE = max(events(:,9));

for iter_SLE = 1: nb_SLE
    curr_SLE = find(events(:,9) == iter_SLE);
    chn_SLE = unique(events(curr_SLE,1));
    nb_chn = length(chn_SLE);
    good_chn = length(ismember(chn_SLE,voisins));
    bad_chn = nb_chn - good_chn;
    events(curr_SLE,8) = events(curr_SLE,8) + scoring_spatial*good_chn -
scoring_spatial*bad_chn;
end

obj.Detector.events_classification = events;

end

```

```

% This function adjust the score based on the temporal criteria
function obj = evaluate_duration(obj,scoring_temporal);
events = obj.Detector.events_classification;
nb_SLE = max(events(:,9));

for iter_SLE = 1: nb_SLE
    curr_SLE = find(events(:,9) == iter_SLE);
    duration_SLE = mean(events(curr_SLE,4));
    if duration_SLE > 150
        events(curr_SLE,8) = events(curr_SLE,8) - scoring_temporal;
    end
end

obj.Detector.events_classification = events;

end

```

```

% This function adjust the score based on spike-HFO cooccurrence.
% Input:      obj = subject_HFO object
%             scoring_cooccurrence = Cooccurrence scoring factor
%             time_buffer = Delay allowed between cooccurring markers (samp)
function obj = evaluate_cooccurrence(obj,scoring_cooccurrence,time_buffer)
obj.Detector.cooccurrence = zeros(3,4);
obj = combine_markers(obj);

% This function compute the cooccurrence between expert's spike markers and
% HFO markers. This is used as a statistical information without affecting
% the detection
occurrence = expert_cooccurrence(obj,time_buffer);
obj.Detector.cooccurrence(1,:) = occurrence;
obj.Detector.cooccurrence(2,:) = occurrence;

% This function compute the cooccurrence between expert's spike markers and
% automatic detections. It also adjust the score of each event based on
% the result, according the the scoring factor in input.
[occurrence event_detector] =
detector_cooccurrence(obj,'classification',scoring_cooccurrence);
obj.Detector.cooccurrence(3,:) = occurrence;

obj.Detector.events_classification= event_detector;

end

```

```

% This function compute the cooccurrence between expert's spike markers and
% HFO markers. This is used as a statistical information without affecting
% the detection
function occurrence = expert_cooccurrence(obj,time_buffer);

time_gamma = 150;
time_ripple = 75;
time_spike = 200;

nb_gamma = obj.nb_gamma;
nb_ripple = obj.nb_ripple;
nb_spike = obj.nb_spike;
nb_sample = obj.NumSamps;

event_gamma = obj.StatusItems_Gamma;
event_ripple = obj.StatusItems_Ripple;
event_spike = obj.StatusItems_Spike;

gamma_start = event_gamma(:,2);
ripple_start = event_ripple(:,2);
spike_start = event_spike(:,2);

% The time window of each gamma markers are adjusted before and after by
% the time buffer used in input.
for iter_gamma = 1:nb_gamma
    sample_start = gamma_start(iter_gamma) - time_buffer;
    sample_stop = gamma_start(iter_gamma) + time_buffer + time_gamma -1;
    if sample_start < 0
        sample_stop = sample_stop - sample_start;
        sample_start = 0;
    elseif sample_stop > nb_sample
        sample_start = nb_sample - (2*time_buffer) - time_gamma;
        sample_stop = nb_sample;
    end
    gamma_samples(iter_gamma,:) = sample_start:sample_stop;
end

% Same with ripple HFOs
for iter_ripple = 1:nb_ripple
    sample_start = ripple_start(iter_ripple) - time_buffer;
    sample_stop = ripple_start(iter_ripple) + time_buffer + time_ripple -
1;
    if sample_start < 0
        sample_stop = sample_stop - sample_start;
        sample_start = 0;
    elseif sample_stop > nb_sample
        sample_start = nb_sample - (2*time_buffer) - time_ripple;
        sample_stop = nb_sample;
    end
    ripple_samples(iter_ripple,:) = sample_start:sample_stop;
end

```

```

% Same with spikes
for iter_spike = 1:nb_spike
    sample_start = spike_start(iter_spike) - time_buffer;
    sample_stop = spike_start(iter_spike) + time_buffer + time_spike -1;
    if sample_start < 0
        sample_stop = sample_stop - sample_start;
        sample_start = 0;
    elseif sample_stop > nb_sample
        sample_start = nb_sample - (2*time_buffer) - time_spike;
        sample_stop = nb_sample;
    end

    spike_samples(iter_spike,:) = sample_start:sample_stop;
end

% Computes gamma-spikes cooccurrence
for iter_gamma = 1:nb_gamma
    current_gamma = gamma_samples(iter_gamma,:);
    flag = 0;
    iter_spike = 1;
    while flag == 0 && iter_spike < nb_spike+1
        current_spike = spike_samples(iter_spike,:);
        flag = sum(ismember(current_gamma,current_spike));
        cooccurrence_gamma_IED(iter_gamma) = flag;
        iter_spike = iter_spike+1;
    end
end

% Computes ripple-spikes cooccurrence
for iter_ripple = 1:nb_ripple
    current_ripple = ripple_samples(iter_ripple,:);
    flag = 0;
    iter_spike = 1;
    while flag == 0 && iter_spike < nb_spike+1
        current_spike = spike_samples(iter_spike,:);
        flag = sum(ismember(current_ripple,current_spike));
        cooccurrence_ripple_IED(iter_ripple) = flag;
        iter_spike = iter_spike+1;
    end
end

% Computes spikes-HFO cooccurrence
for iter_spike = 1:nb_spike
    current_spike = spike_samples(iter_spike,:);
    flag = 0;
    iter_gamma = 1;
    while flag == 0 && iter_gamma < nb_gamma+1
        current_gamma = gamma_samples(iter_gamma,:);
        flag = sum(ismember(current_spike,current_gamma));
        cooccurrence_IED_gamma(iter_spike) = flag;
        iter_gamma = iter_gamma+1;
    end
end

```

```

flag = 0;
iter_ripple = 1;
while flag == 0 && iter_ripple < nb_ripple+1
    current_ripple = ripple_samples(iter_ripple,:);
    flag = sum(ismember(current_spike,current_ripple));
    cooccurrence_IED_ripple(iter_spike) = flag;
    iter_ripple = iter_ripple+1;
end
end

cooccurrence_IED_gamma = cooccurrence_IED_gamma>0;
cooccurrence_IED_ripple = cooccurrence_IED_ripple>0;
cooccurrence_gamma_IED = cooccurrence_gamma_IED>0;
cooccurrence_ripple_IED = cooccurrence_ripple_IED>0;

cooccurrence_IED_gamma = sum(cooccurrence_IED_gamma)/nb_spike;
cooccurrence_IED_ripple = sum(cooccurrence_IED_ripple)/nb_spike;
cooccurrence_gamma_IED = sum(cooccurrence_gamma_IED)/nb_gamma;
cooccurrence_ripple_IED = sum(cooccurrence_ripple_IED)/nb_ripple;

occurrence= [cooccurrence_IED_gamma cooccurrence_IED_ripple
cooccurrence_gamma_IED cooccurrence_ripple_IED];
end

```

```
% This function compute the cooccurrence between expert's spike markers and
% automatic detections. It also adjust the score of each event based on
% the result, according the the scoring factor in input.
```

```
function [occurrence event_detector] =
detector_cooccurrence(obj,method,scoring_cooccurrence);
time_buffer = 50;
time_gamma = 150;
time_ripple = 75;
time_spike = 200;

if strcmp(method,'classification')
    event_detector = obj.Detector.events_classification;
    event_detector(:,8) = 0;
    flag_classification = 1;
else
    event_detector = obj.Detector.events_predetection;
    flag_classification = 0;
end
event_spike = obj.StatusItems_Spike;

nb_detector = max(event_detector(:,9));
nb_spike = obj.nb_spike;
nb_sample = obj.NumSamps;

detector_start = event_detector(:,3);
spike_start = event_spike(:,2);

iter_gamma = 1;
iter_ripple = 1;

% Computes event-spikes cooccurrence
for iter_detector = 1:nb_detector
    id_SLE = find(event_detector(:,9) == iter_detector);
    detector_start = min(event_detector(id_SLE,3));

    sample_start = detector_start - time_buffer;
    sample_stop_gamma = detector_start + time_buffer + time_gamma -1;
    sample_stop_ripple = detector_start + time_buffer + time_ripple -1;

    if sample_start < 0
        sample_stop_gamma = sample_stop_gamma - sample_start;
        sample_stop_ripple = sample_stop_ripple - sample_start;
        sample_start = 0;
    elseif sample_stop_gamma > nb_sample
        sample_start = nb_sample - (2*time_buffer) - time_gamma;
        sample_stop = nb_sample;
    elseif sample_stop_ripple > nb_sample
        sample_start = nb_sample - (2*time_buffer) - time_ripple;
        sample_stop = nb_sample;
    end
end
```

```

    if min(event_detector(id_SLE,2)) < 200
        gamma_samples(iter_gamma,:) = sample_start:sample_stop_gamma;
        iter_gamma = iter_gamma +1;
    elseif max(event_detector(id_SLE,2)) > 200
        ripple_samples(iter_ripple,:) = sample_start:sample_stop_ripple;
        iter_ripple = iter_ripple +1;
    end

end

[nb_gamma whatever] = size(gamma_samples);
nb_ripple = 0;

for iter_spike = 1:nb_spike
    sample_start = spike_start(iter_spike) - time_buffer;
    sample_stop = spike_start(iter_spike) + time_buffer + time_spike -1;
    if sample_start < 0
        sample_stop = sample_stop - sample_start;
        sample_start = 0;
    elseif sample_stop > nb_sample
        sample_start = nb_sample - (2*time_buffer) - time_spike;
        sample_stop = nb_sample;
    end

    spike_samples(iter_spike,:) = sample_start:sample_stop;
end

% Adjusts the score for each Gamma events based on the cooccurrence factor
for iter_gamma = 1:nb_gamma
    current_gamma = gamma_samples(iter_gamma,:);
    flag = 0;
    iter_spike = 1;
    while flag == 0 && iter_spike < nb_spike+1
        current_spike = spike_samples(iter_spike,:);
        flag = sum(ismember(current_gamma,current_spike));
        cooccurrence_gamma_IED(iter_gamma) = flag;
        iter_spike = iter_spike+1;
    end
    if flag_classification ==1
        id_SLE = find(event_detector(:,9) == iter_gamma);
        if cooccurrence_gamma_IED(iter_gamma)
            event_detector(id_SLE,8) = event_detector(id_SLE,8) ...
                + scoring_cooccurrence;
        else
            event_detector(id_SLE,8) = event_detector(id_SLE,8) ...
                - scoring_cooccurrence;
        end
    end
end

% Adjusts the score for each Ripple events based on the cooccurrence factor
for iter_ripple = 1:nb_ripple

```

```

current_ripple = ripple_samples(iter_ripple,:);
flag = 0;
iter_spike = 1;
while flag == 0 && iter_spike < nb_spike+1
    current_spike = spike_samples(iter_spike,:);
    flag = sum(ismember(current_ripple,current_spike));
    cooccurrence_ripple_IED(iter_ripple) = flag;
    iter_spike = iter_spike+1;
end
if flag_classification ==1
    id_SLE = find(event_detector(:,9) == iter_ripple);
    if cooccurrence_gamma_IED(iter_gamma)
        event_detector(id_SLE,8) = event_detector(id_SLE,8) ...
            + scoring_cooccurrence;
    else
        event_detector(id_SLE,8) = event_detector(id_SLE,8) ...
            - scoring_cooccurrence;
    end
end
end

for iter_spike = 1:nb_spike
    current_spike = spike_samples(iter_spike,:);
    flag = 0;
    iter_gamma = 1;
    while flag == 0 && iter_gamma < nb_gamma+1
        current_gamma = gamma_samples(iter_gamma,:);
        flag = sum(ismember(current_spike,current_gamma));
        cooccurrence_IED_gamma(iter_spike) = flag;
        iter_gamma = iter_gamma+1;
    end

    flag = 0;
    iter_ripple = 1;
    while flag == 0 && iter_ripple < nb_ripple+1
        current_ripple = ripple_samples(iter_ripple,:);
        flag = sum(ismember(current_spike,current_ripple));
        cooccurrence_IED_ripple(iter_spike) = flag;
        iter_ripple = iter_ripple+1;
    end
end

cooccurrence_IED_gamma = cooccurrence_IED_gamma>0;
cooccurrence_IED_ripple = 0;
cooccurrence_gamma_IED = cooccurrence_gamma_IED>0;
cooccurrence_ripple_IED = 0;

cooccurrence_IED_gamma = sum(cooccurrence_IED_gamma)/nb_spike;
cooccurrence_IED_ripple = 0;
cooccurrence_gamma_IED = sum(cooccurrence_gamma_IED)/nb_gamma;
cooccurrence_ripple_IED = 0;

```

```
occurrence = [cooccurrence_IED_gamma cooccurrence_IED_ripple  
cooccurrence_gamma_IED cooccurrence_ripple_IED];  
  
end  
  
% This function removes every events with a score inferior to the  
% threshold in input.  
function obj = remove_scoring(obj,threshold)  
  
events = obj.Detector.events_classification;  
id_SLE = find(events(:,8)<threshold);  
events(id_SLE,:) = [];  
  
% Renumber subject level events  
ix=unique(events(:,9)); ev=events;  
for ii=1:length(ix), events(ev(:,9)==ix(ii),9)=ii; end  
  
obj.Detector.events_classification = events;  
  
end
```


ANNEXE IV

CODE MATLAB – PRÉPARATION DES DONNÉES (PREPROCESSING)

```
% This function makes sure that every marker that overlaps in time but on
% different channels are given the same tag to identify them as the
% "same event". Channels on which they were marked are recorded
function obj = combine_markers(obj)
if sum(obj.StatusItems_Gamma(:,4) == 0);
    obj = create_SLE(obj);
    obj = update_info(obj);
else
    disp('Expert''s markers are already combined')
end
end

function obj = create_SLE(obj)
% Give same id to simultaneous markers in both Gamma and Ripple

    % [index Begin_sample Duration DetType]
    new_gamma = obj.StatusItems_Gamma;
    new_ripple = obj.StatusItems_Ripple;
    new_spike = obj.StatusItems_Spike;

    % number of original markers
    nb_gamma = obj.nb_gamma;
    nb_ripple = obj.nb_ripple;
    nb_spike = obj.nb_spike;

% Create markers, matrix of sample id
for iter_gamma = 1:nb_gamma
    markers_gamma{iter_gamma} = new_gamma(iter_gamma,2):...
        new_gamma(iter_gamma,2)+new_gamma(iter_gamma,3);
end
for iter_ripple = 1:nb_ripple
    markers_ripple{iter_ripple} = new_ripple(iter_ripple,2):...
        new_ripple(iter_ripple,2)+new_ripple(iter_ripple,3);
end
for iter_spike = 1:nb_spike
    markers_spike{iter_spike} = new_spike(iter_spike,2):...
        new_spike(iter_spike,2)+new_spike(iter_spike,3);
end

% Give same id to simultaneous markers in Gamma
flag = 1;
iter_marker = 1;
counter = 1;
new_gamma(:,4) = 0;
revised_nb_gamma = nb_gamma;
```

```

while revised_nb_gamma > 0
    current_marker = markers_gamma{1};
    for iter_gamma = 1:revised_nb_gamma
        next_marker = markers_gamma{iter_gamma};
        flag_detection = sum(ismember(current_marker,next_marker));
        if flag_detection > 0
            new_gamma(counter,4) = iter_marker;
            counter = counter +1;
            markers_gamma{iter_gamma} = [];
        end
    end
    markers_gamma = markers_gamma(~cellfun('isempty',markers_gamma));
    iter_marker = iter_marker+1;
    [flag revised_nb_gamma] = size(markers_gamma);
end

% Same for Ripple
flag = 1;
iter_marker = 1;
counter = 1;
new_ripple(:,4) = 0;
revised_nb_ripple = nb_ripple;

while revised_nb_ripple > 0
    current_marker = markers_ripple{1};
    for iter_ripple = 1:revised_nb_ripple
        next_marker = markers_ripple{iter_ripple};
        flag_detection = sum(ismember(current_marker,next_marker));
        if flag_detection > 0
            new_ripple(counter,4) = iter_marker;
            counter = counter +1;
            markers_ripple{iter_ripple} = [];
        end
    end
    markers_ripple = markers_ripple(~cellfun('isempty',markers_ripple));
    iter_marker = iter_marker+1;
    [flag revised_nb_ripple] = size(markers_ripple);
end

% Same for Spike
flag = 1;
iter_marker = 1;
counter = 1;
new_spike(:,4) = 0;
revised_nb_spike = nb_spike;

while revised_nb_spike > 0
    current_marker = markers_spike{1};
    for iter_spike = 1:revised_nb_spike
        next_marker = markers_spike{iter_spike};
        flag_detection = sum(ismember(current_marker,next_marker));

```

```

        if flag_detection > 0
            new_spike(counter,4) = iter_marker;
            counter = counter +1;
            markers_spike{iter_spike} = [];
        end
    end
    markers_spike = markers_spike(~cellfun('isempty',markers_spike));
    iter_marker = iter_marker+1;
    [flag revised_nb_spike] = size(markers_spike);
end

obj.nb_gamma = new_gamma(end,4);
obj.nb_ripple = new_ripple(end,4);
obj.nb_spike = new_spike(end,4);

obj.StatusItems_Gamma = new_gamma;
obj.StatusItems_Ripple = new_ripple;
obj.StatusItems_Spike = new_spike;
end

function obj = update_info(obj)
% Update channel labels and lists and remove redundant markers

markers_gamma = obj.StatusItems_Gamma;
markers_ripple = obj.StatusItems_Ripple;
markers_spike = obj.StatusItems_Spike;

% number of revised markers
nb_gamma = obj.nb_gamma;
nb_ripple = obj.nb_ripple;
nb_spike = obj.nb_spike;

for iter_gamma = 1: nb_gamma
    id_SLE = find(markers_gamma(:,4) == iter_gamma);
    current_labels = obj.Channels_Gamma(id_SLE);
    current_lists = obj.Channels_Gamma_list(id_SLE);
    new_labels{iter_gamma} = current_labels;
    new_lists{iter_gamma} = find_chn_tag(new_labels{iter_gamma}...
        ,1,iter_gamma);
    if length(id_SLE)>1
        id_SLE(1) = [];
        markers_gamma(id_SLE,:) = [];
    end
end

obj.StatusItems_Gamma = markers_gamma;
obj.Channels_Gamma_list = new_lists;
obj.Channels_Gamma = new_labels;

clear new_lists
clear new_labels

```

```

for iter_ripple = 1: nb_ripple
    id_SLE = find(markers_ripple(:,4) == iter_ripple);
    current_labels = obj.Channels_Ripple(id_SLE);
    current_lists = obj.Channels_Ripple_list(id_SLE);
    new_labels{iter_ripple} = current_labels;
    new_lists{iter_ripple} = find_chn_tag(new_labels{iter_ripple}...
        ,2,iter_ripple);
    if length(id_SLE)>1
        id_SLE(1) = [];
        markers_ripple(id_SLE,:) = [];
    end
end

obj.StatusItems_Ripple = markers_ripple;
obj.Channels_Ripple_list = new_lists;
obj.Channels_Ripple = new_labels;

clear new_lists
clear new_labels

for iter_spike = 1: nb_spike
    id_SLE = find(markers_spike(:,4) == iter_spike);
    current_labels = obj.Channels_Spike(id_SLE);
    current_lists = obj.Channels_Spike_list(id_SLE);
    new_labels{iter_spike} = current_labels;
    new_lists{iter_spike} = find_chn_tag(new_labels{iter_spike}...
        ,3,iter_spike);
    if length(id_SLE)>1
        id_SLE(1) = [];
        markers_spike(id_SLE,:) = [];
    end
end

obj.StatusItems_Spike = markers_spike;
obj.Channels_Spike_list = new_lists;
obj.Channels_Spike = new_labels;
end

```

```

% This function completely removes a channel from the EEG recording.
% Channel 8 is always removed, as it is the ECG channel, but additionnal
% channels could be removed basd on the expert's opinion about those
% he finds are unusable.
% The channel to be removed can be an input, otherwise a list of channels
% will appear and the user the select which one to remove
function obj = remove_channel(obj,chn_to_remove);
clc
labels = obj.ChnLabel;
nb_chn = length(labels);
iter = 1;
disp('0- none')

%Display all the channels
for ii =1:nb_chn
    disp([int2str(iter),'- ',labels{ii}])
    iter = iter+1;
end

%
if nargin == 1
    chn_to_remove = input('\n Select the channel to remove please: ');
end

if chn_to_remove == 0;
    disp('No channel was removed');
    return
elseif strcmp('REMOVED',labels{chn_to_remove}) == 1
    disp('This channel was already removed');
    return
end

% The label is changed to REMOVED
obj.ChnLabel{chn_to_remove} = 'REMOVED';

removed_flag = strcmp('REMOVED',obj.ChnLabel);
chn_to_remove = chn_to_remove - sum(removed_flag(1:chn_to_remove-1));
obj.Channels_Gamma_list
obj.Channels_Ripple_list

% The EEG recording of that channel is erased
nb_eegdata = length(obj.eegdata);
for ii = 1:nb_eegdata
    eegdata = obj.eegdata{ii};
    eegdata(:,chn_to_remove) = [];
    obj.eegdata{ii} = eegdata;
end

% The channel index of all expert's marker are adjusted
for iter_gamma = 1:obj.nb_gamma
    curr_chn = obj.Channels_Gamma_list{iter_gamma};
    chn_id = find(curr_chn >= chn_to_remove);

```

```
    curr_chn(chn_id) = curr_chn(chn_id)-1;
    obj.Channels_Gamma_list{iter_gamma} = curr_chn;
end

% Same for ripple
for iter_ripple = 1:obj.nb_ripple
    curr_chn = obj.Channels_Ripple_list{iter_ripple};
    chn_id = find(curr_chn >= chn_to_remove);
    curr_chn(chn_id) = curr_chn(chn_id)-1;
    obj.Channels_Ripple_list{iter_ripple} = curr_chn;
end

obj.Detector.root = obj;

end
```

```

% This function allocated a tag based to each events based on subject
level
% events. In other words, events occuring at the same time, regardless of
% channels, are given the same SLE tag.
% input: events = events to be allocated SLE
%         tstart = first sample of the interval (usually 0)
%         tstop  = last sample of the interval (usually subject.NumSamps)
function new_events = allocate_SLE(events,tstart,tstop)

duration = tstop-tstart;
detection_flag = zeros(1,duration);
[nb_events whatever] = size(events);

for iter_events = 1 : nb_events
    event_start = events(iter_events,3);
    event_stop  = event_start + events(iter_events,4);
    detection_flag(event_start:event_stop) = 1;
end

dsig = diff([detection_flag]);
startIndex = find(dsig > 0)+1;
endIndex = find(dsig < 0)-1;
durationIndex = endIndex-startIndex+1;

nb_SLE = length(startIndex);
for iter_SLE = 1:nb_SLE
    detection_flag(startIndex(iter_SLE):endIndex(iter_SLE)) = iter_SLE;
end

for iter_events = 1 : nb_events
    event_start = events(iter_events,3)+1;
    events(iter_events,9) = detection_flag(event_start);
end

new_events = events;

end

```


ANNEXE V

CODE MATLAB – EXEMPLE COMPLET

```
clear all
close all
clc

% load the .mat file of the subject
load 'struct_PJ 11-2391-1000 HZ_30 MINUTES SLEEP_spike'
% Create the Subject object
subject = subject_HFO(subject);
% Combine expert's markers occuring at the same time
subject = combine_markers(subject)
% Remove ECG (channel 8) and optionnaly other channels
subject = remove_channel(subject,8)

% optimize detection threshold using a subsample of 10 minutes
subject = optimize_umb(subject,10);
% Detection
subject.Detector = subject.Detector.HFDDetector;
% Allocate SLE to all events
subject.Detector.events_predetection =
allocate_SLE(subject.Detector.events_predetection, 1,subject.NumSamps);
subject.Detector.events_classification =
subject.Detector.events_predetection;
% Score each event using the factors of your choice, see chapter 5
subject = classify_events(subject,0.2,0.1,0.5);
% remove events with a low score (optional)
subject = remove_scoring(subject,iter_score);
```


BIBLIOGRAPHIE

- Anne-Sophie Dubarry. 2010. « Détections de pointes épileptiques à partir de signaux EEG ». mémoire ETS
- C.G. Bénar, L. Chauvière, F. Bartolomei, F. Wendling. 2009. « Pitfalls of high-pass filtering for detecting epileptic oscillations: a technical note on “false” ripples ». *Clinical Neurology*, doi;10.1016/j.clinph.2009.10.019
- Dimitri M Kullmann. 2011. « Interneuron networks in the hippocampus ». *Neurobiology*, 21 :709-716
- Erin Mann and I. Mody, *Nat Neurosci*. 2001. « Control of hippocampal gamma oscillation frequency by tonic inhibition and excitation of interneurons ». 13(2): 205-212. Doi 10.1038/nn.2464
- Federico Melani, Rina Zelmann, François Dubeau. 2013. « Occurrence of scalp-fast oscillations among patients with different spiking rate and their role as epileptogenicity marker ». *epilepsy research*, 106, 345-356
- Felix Rosenow, Hans Luders. 2001. « Presurgical evaluation of epilepsy ». *Brain*, 124,1683-1700
- Hans O. Luders, Imad Najm, Dileep Nair, Peter Widdess-Walsh. William Bingman, 2006. « The epileptogenic zone: general principles ». *Epileptic Disord*. 8 (suppl. 2): S1-9
- Ivan Cohen, Vincent Navarro, Stéphane Clemenceau, Michel Baulac, Richard Miles. 2002. « On the origin of interictal activity in human temporal lobe epilepsy in vitro ». *Science*, Vol 298.
- J. Gotman. 1999. « Automatic Detection of seizures and spikes, *Journal of clinical neurophysiology* ». vol 16 – issue 2 – pp 130-140
- J.D. Jirsh, E. Urrestarazu, P. LeVan, A. Olivier, F. Dubeau, J. Gotman. 2006. « High-Frequency oscillation during human focal seizure ». *Brain*. 129, 1593-1608.
- L.P. Andrade-Valence, F. Dubeau, F. Mari et al. 2011. « Interictal scalp fast oscillations as a marker of the seizure onset zone ». *Neurology*; 77;523; Published online before print July 13, 2011; DOI 10.1212/WNL

- M. Esclapez, J.C Hirsch, R. Khazipov, Y. Ben-Ari, C. Bernard. 1997. « Operative GABAergic inhibition in hippocampal CA1 pyramidal neurons in experimental epilepsy ». National Academy of Sciences, 0027-8424/97/9412151-682.200
- Nicolas von Ellenrieder, Luciana P. Andrade-Valença, François Dubeau, Jean Gotman. 2011. « Automatic detection of fast oscillations (40-200 Hz) in scalp EEG recording ». *Clinical Neurophysiology*, doi : 10.1016/j.clinph.2011.03.050
- Pascale P. Quilichini, Michel Le Van Quyen, Anton Ivanov, Dennis A. Turner, Aurelie Carabalona, Henro Gozlan, Monique Exclapez, Christophe Bernard. 2012. « Hub GABA neurons Mediate Gamma-Frequency oscillations at ictal-like Event Onset in the immature hippocampus ». *Neuron* 74, 47-64
- Rina Zelman, Maeike Zijlman, Julia Jacobs, Claude-E. Chatillon, Jean Gotman. 2009. « Improving the identification of high frequency oscillations ». *Clinical neurophysiology*, 120, 1457- 1464
- R. Zelman, F. Mari, J. Jacobs, M. Ziklmans, J. Gotman. 2012. « A comparison between detectors of high frequency oscillations ». *Clinical Neurophysiology*, 106-116
- Rosa Cossart, Christophe Bernard, Yehezkel Ben-Ari, 2005, « Multiple facets of GABAergic neurons and synapses : multiple fates of GABA signalling in epilepsies ». *Neuroscience*, Vol 28. No.8
- T.F. Freund and G. Buzsaki. 1996. « Interneurons of the hippocampus ». *Wiley-Liss, INC*, 6:347-470

