الجمـــهوريـــة الجـــزائريـــة الديـمقـراطيـــــــة الشـــعبيـــــة
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
وزارة التعـلـيـم الـعـالـــي و الـبـحـث الـعـلـمي
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
جـامـعـة عمّـــار ثـلـيـــجـــي بالأغـــواط
UNIVERSITE AMAR TELIDJI LAGHOUAT
كـلـيـــــة الـعـلـــــوم
**FACULTE DES SCIENCES**
قـســم الإعـــلام الآلـــي
DEPARTEMENT D'INFORMATIQUE

# *Mémoire de MASTER*

| | |
|---|---|
| **Domaine :** | Mathématiques et Informatique |
| **Filière :** | Informatiques |
| **Option :** | Réseaux, Systèmes et Applications Réparties |

**Par**:

- **BRAHIM Mohamed**
- **MILOUDIA Kamel**

**THEME**

# Evaluation du Parallélisme de la Méthode De Segmentation à Energie Moyenne Continue et du Calcul des MFCC

Soutenu publiquement le 01-06-2017 devant le jury composé de:

| | | |
|---|---|---|
| Mme TAABA Kheira | M.C.(A) | Président |
| Mr REGGAB Mourad | M.C.(A) | Examinateur |
| Mr MAAMAR Ahfir | M.C.(A) | Encadreur |

**Année Universitaire 2016/2017**

# Dédicace

Je dédie ce travail à tous ceux qui m'ont soutenu durant ce labeur, et tout spécialement mes parents, mon encadreur Mr. Ahfir, Tous mes enseignants sans lesquels je n'aurai pas eu le savoir nécessaire pour effectuer ce travail.

**Mohamed**

# Dedicace

Je tiens A dédicacer ce travail à mes parents qui m'ont toujours pris la main et guide dans tous mes choix et qui ont toujours veille à mon bonheur.

Je tiens aussi à faire une dédicace à mes frères et sœurs qui ont su me supporter durant les moments de stresse et de fatigue

Sans oublier mon encadreur Mr. Ahfir, qui a été là durant tout ce travail jours et nuits.

Kamel

***Remerciements***

***C'est avec un grand plaisir qu'on réserve cette page en signe de gratitude et de profonde reconnaissance à tous ceux qui nous ont aidés de près ou de loin à la réalisation de ce travail.***

***On tient tout d'abord à exprimer nos sincères remerciements et respects à notre encadreur Mr : AHFIR Maamar maître de conférences à l'université de Laghouat, pour son encouragement et les précieux conseils qu'il n'a cessé de nous prodiguer et qui nous ont permis d'achever ce travail.***

***On souhaite également exprimer notre profonde gratitude à nos enseignants durant le cursus universitaire pour leurs conseils et leurs contributions.***

***Enfin, nos meilleurs et vifs remerciements s'adressent aux membres du jury pour l'honneur qu'ils nous font en acceptant d'examiner et d'évaluer ce modeste travail durant lequel nous avons tant appris.***

# Sommaire

# Liste des Figures

# Liste des Tableaux

# Liste des Abréviations

**LFO/VC:** Longueur du fichier original / Vitesse de Conversion.

**TFO/TMMFCC:** la taille du fichier original / la taille de la matrice MFCC.

**RIFF:** Resource Interchange File Format.

**LPC:** Linear Predictive Codes.

**PLP:** Perceptual Linear Prediction.

**MFCC:** Mel Frequency Cepstrum Coefficient.

**Résumé :**

Dans ce mémoire, nous avons décrit quelques notions du traitement du signal de parole comme la numérisation du signal analogique, la segmentation du signal de parole et le calcul de paramètre décrivant ce dernier. Nous avons ensuite parallélisé l'algorithme de segmentation à énergie moyenne continue et nous avons décrit un algorithme de calcul des MFCC et son inverse. Pour finalement, les tester sur plusieurs fichiers Wave de différentes catégories et les résultats montrent que la segmentation par la méthode décrite peut être accélérée grâce au parallélisme, mais aussi que le nombre de coefficients MFCC optimal pour la transmission et la reconstruction soit de 20.

**Abstract :**

In this memoire, we have described some notions of signal processing such as digitization of the analog signal, the segmentation of the vocal signal and calculation of parameters describing the latter. Then, we parallelized the continuous average energy segmentation algorithm and we have described an algorithm for computing MFCC and its inverse. Finally, we tested them on several Wave files of different categories and the results show that this segmentation method can be accelerated by parallelism, also the optimal number of MFCC coefficients for the transmission and the reconstruction is 20.

**ملخص:**

في هذه المذكرة، وصفنا بعض مفاهيم معالجة الإشارات مثل رقمنة الإشارة التناظرية، وتجزئة إشارة الصوتية وحساب المعلمات التي تصف هذه الأخيرة. ثم جعلنا خوارزمية التجزئة بالطاقة المتوسطة المتواصلة تتم بشكل موازٍ ووصفنا خوارزمية لحساب MFCC وعكسها. وأخيرا، اختبرناهما على عدة ملفات WAVE من فئات مختلفة، وتبين النتائج أن هذا الأسلوب للتجزئة يمكن تسريعه بالتوازي، وأيضا أن العدد الأمثل لمعاملات MFCC للإرسال وإعادة بناء الإشارة الأصلية هو 20.

# Introduction Générale

# Introduction Générale

La reconnaissance automatique de la parole/locuteur permet à la machine de comprendre et de traiter des informations fournies oralement par l'utilisateur humain. Elle consiste à employer des techniques afin de comparer une onde sonore à un ensemble d'échantillons, composés généralement de mots mais aussi, des phonèmes (unité sonore minimale).

La reconnaissance de la parole, fait appel à plusieurs disciplines telles que : l'anatomie (les fonctions de l'appareil phonatoire et de l'oreille), la phonétique, le traitement du signal, la linguistique, l'informatique, l'intelligence artificielle et les statistiques, etc.

Un système de reconnaissance automatique de la parole SRAP se décompose en plusieurs phases :la production de la parole, la numérisation du signal, la segmentation, la para métrisation, la reconnaissance.

La segmentation désigne est une étape indispensable dans les systèmes SRAP (Reconnaissance vocale, Synthèse vocale, etc.) ; c'est le processus de la division d'une entité, généralement continue, en petites entités appelées segments. Chaque segment possède des propriétés propres qui permettent de le différencier des autres.

Dans notre travail, on s'intéresse à la segmentation (analyse et traitement) du signal de paroles, qui fonctionnent d'une manière automatique et/ou en temps réel et nous mettons l'accent sur le parallélisme de celle-ci ainsi que le calcule de paramètre MFCC.

Dans un premier temps, nous allons essayer d'appliquer un algorithme sur des signaux comportant des mots isolés, et par la suite, on le généralise sur les signaux de la parole continue.

La technique employée dans notre travail est la détection d'activité vocale VAD. Le VAD est un module largement utilisé dans une grande gamme d'applications du traitement automatique de la parole. Il est utilisé pour localiser le début et la fin des régions (segments) à reconnaître. La précision du VAD utilisé se matérialise dans une amélioration du taux de reconnaissance et l'accélération du traitement en question.

Ensuite nous allons évaluer les performances du calcul des paramètres MFCC et la reconstruction de signaux de parole depuis ceux-ci.

Ce mémoire s'articule autour de trois chapitres :

- Le premier chapitre, est un aperçu général du domaine de traitement de signal numérique, de la segmentation et des différents paramètres qui peuvent caractériser un signal de la parole.

- Le deuxième chapitre présent la conversion d'un algorithme de segmentation automatique, basé sur l'énergie moyenne continue du signal de la parole depuis MATLAB vers Delphi ainsi que son parallélisme, pour ensuite décrire le code de calcul des paramètres MFCC et leur reconstruction en signal de parole.

- Le troisième chapitre, présente le test et validation des deux codes ainsi qu'une mesure des performances sur la base de plusieurs fichiers audio de différentes catégories.

# Chapitre I

# Outils de Signal de Parole

## 1. Introduction

Dans ce chapitre, Nous allons rappeler quelques généralités sur la théorie du signal et plus spécifiquement le signal de parole. Ensuite nous allons présenter les différentes méthodes de segmentation automatique des signaux audio, utiles aussi bien pour la synthèse que pour la reconnaissance de la parole continue.

Et afin d'alléger le signal segmenté, il est nécessaire d'utiliser une méthode de modélisation de celui-ci pour qui' il sera prêt pour l'entrainement d'un système de reconnaissance vocal, ou tout simplement pour l'envoyer vers un récepteur afin d'être reconstruit.

## 2. Numérisation du Signal Analogique

Un signal est la représentation physique d'une information qui est transportée avec ou sans transformation, de la source jusqu'au destinataire. Il en existe deux catégories :

- Les signaux analogiques, qui varient de façon continue dans le temps (intensité sonore, intensité lumineuse, pression, tension), c'est-à-dire qu'ils peuvent prendre une infinité de valeurs différentes.
- Les signaux numériques qui transportent une information sous la forme de nombres.

Le signal analogique à convertir est une tension électrique variable issue d'un capteur (microphone par exemple) ou d'un circuit électrique.

On obtient alors la courbe à la Figure I-1 représentant le signal analogique :

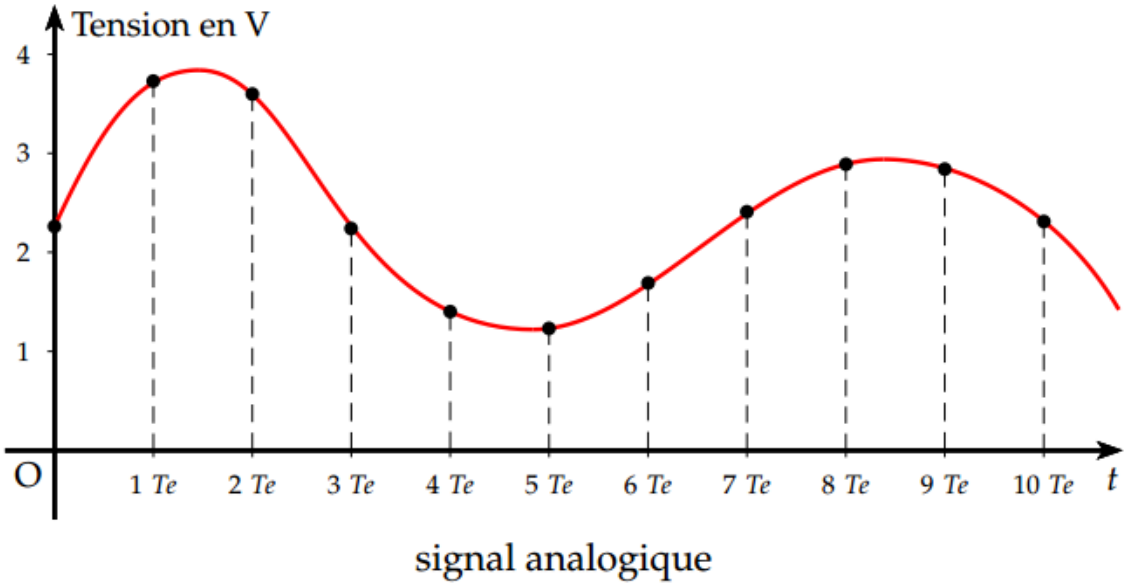


Figure I-1 Echantillonnage du signal analogique

Numériser un signal analogique consiste à transformer les grandeurs continues dans le temps en des grandeurs discontinues qui varient par palier en prenant des valeurs à intervalle de temps régulier : période d'échantillonnage $T_e$

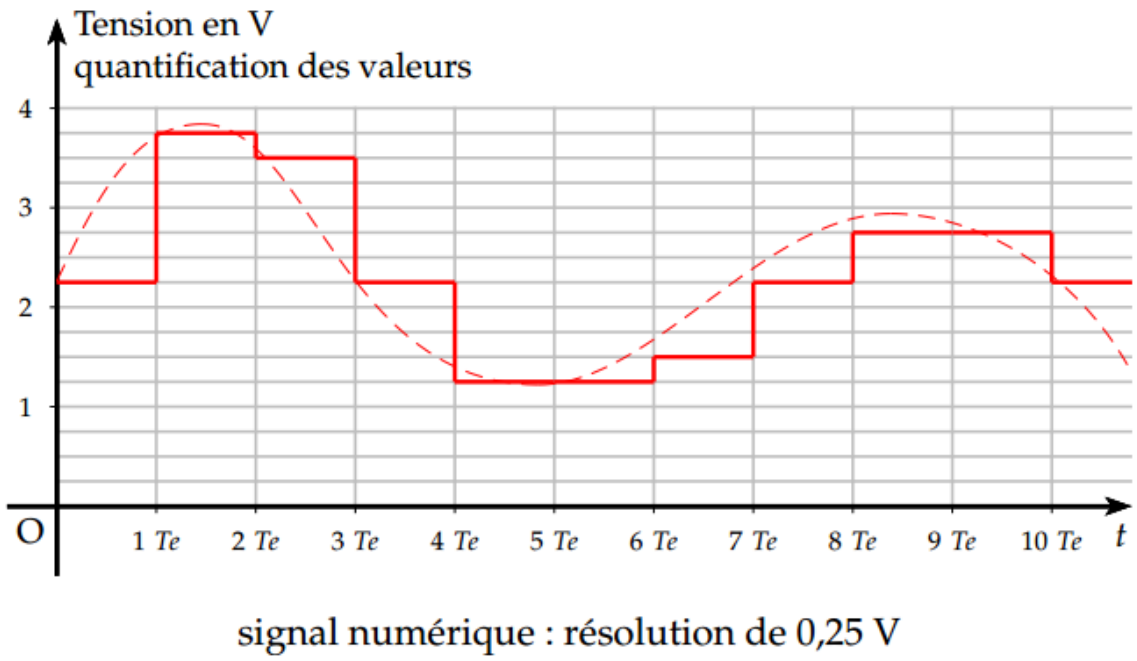


Figure I-2 Quantification du signal analogique.

La numérisation est d'autant meilleure que le signal numérique se rapproche du signal analogique initial.

La numérisation d'un signal nécessite trois étapes :

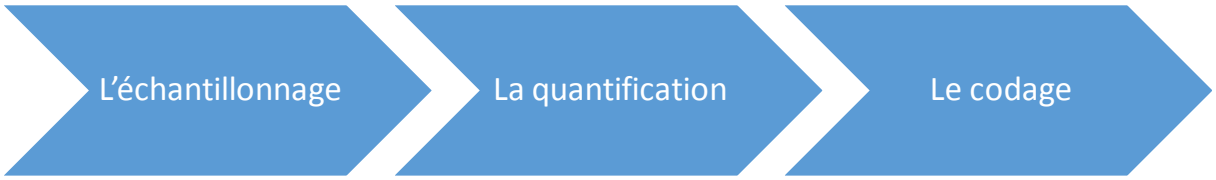


Figure I-3 Etapes de la numérisation du signal analogique.

### 2.1.L'échantillonnage

On appelle période d'échantillonnage Te (en s), le temps entre deux mesures successives.

La fréquence d'échantillonnage$f_e$, correspond au nombre de mesures effectuées par seconde.

On a : $f_e = \frac{1}{T_e}$ Equation I-1

Le choix de la fréquence d'échantillonnage est crucial afin de reproduire fidèlement le signal étudié. En effet si le signal analogique varie trop vite par rapport à la fréquence d'échantillonnage, la numérisation donnera un rendu incorrect.

*Théorème de Shannon : Pour un signal périodique (comme un son) la fréquence d'échantionnage fe doit être au moins le double de la fréquence maximale $f_{max}$ du signal : $f_e \geq 2f_{max}$*

Les fichiers audio sont couramment échantillonnés à 44,1 kHz, car cela permet de restituer des sons dont la fréquence peut aller jusqu'à 22,05 kHz, c'est-à-dire un peu au-delà de la fréquence maximale audible par l'Homme (20 kHz).

### 2.2.La quantification

Un signal numérique ne peut prendre que certaines valeurs : c'est la quantification. Elle s'exprime en bits. Cette quantification est assurée par un convertisseur (CAN). Chaque valeur est arrondie à la valeur permise la plus proche par défaut.

On appelle alors résolution ou pas l'écart (constant) entre deux valeurs permises successives.

### 2.3.Le codage

On appelle codage la transformation des différentes valeurs quantifiées en langage binaire.

Pour calculer la taille N en octet d'un signal audio numérisé, Nous appliquons l'équation suivante :

$$N = F \times \frac{Q}{8} \times n \qquad \text{Equation I-2}$$

Ou F : fréquence échantillonnage en Hz, Q : quantification en bits et n : nombre de voies (si le son est stéréo, n = 2 ; en mono : n = 1).

## 3. Le format RIFF

Afin de pouvoir manipuler des données, il est nécessaire de connaitre l'encodage de ceux-ci.

Nous allons donc expliquer les rudiments de l'encodage utilisé dans ce projet, qu'est le format Wave qui est un sous ensemble de la spécification RIFF de Microsoft. Le fichier commence par un entête suivi par une séquence de morceau de donnée ceci est appelé la forme canonique.

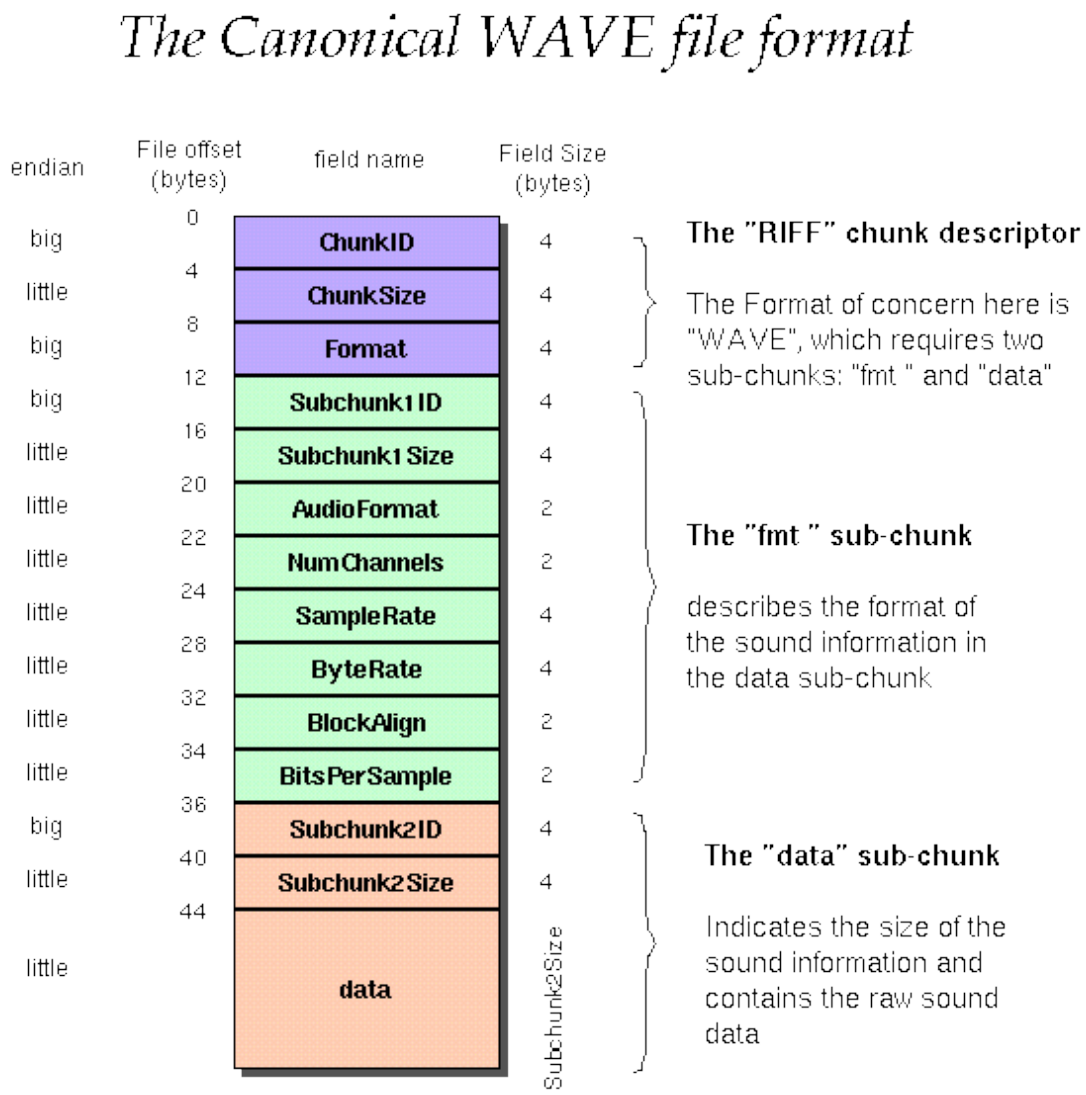


Figure I-4 Schéma de la spécification du format RIFF

Comme un exemple, voici les 72 octet de donné d'un fichier WAVE montré en nombre hexadécimal.

```
52 49 46 46 24 08 00 00 57 41 56 45 66 6d 74 20 10 00 00 00 01 00 02
00 22 56 00 00 88 58 01 00 04 00 10 00 64 61 74 61 00 08 00 00 00 00 00 00
24 17 1e f3 3c 13 3c 14 16 f9 18 f9 34 e7 23 a6 3c f2 24 f2 11 ce 1a 0d
```

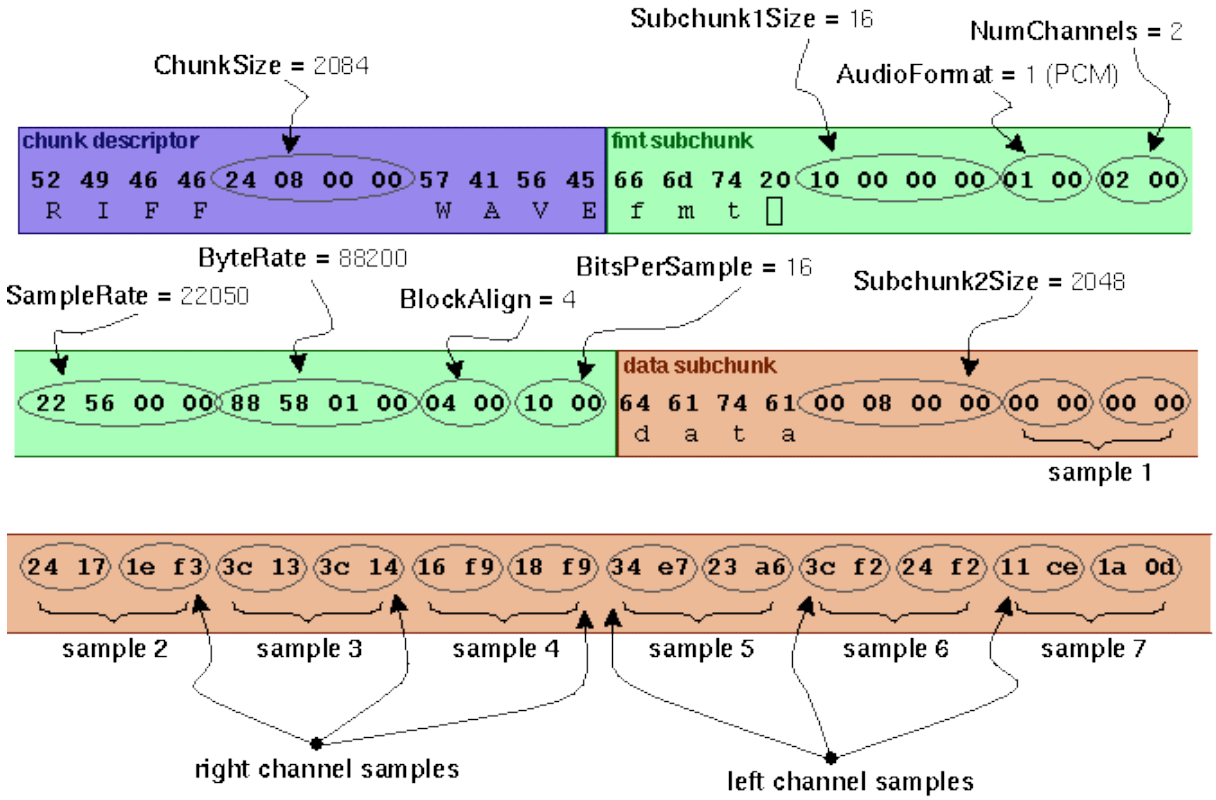


Figure I-5 Exemple du début d'un fichier Wave

Par défaut l'ordre d'octet dans un fichier WAVE est en little-endian.

Les échantillons à 8 bits son stocké en entier non signé et les échantillons à 16 bits sont stocké en complément à deux d'entiers signé.

Les applications multimédia nécessitent le stockage et le traitement d'une large variété de donnée. Le RIFF permet de le faire simplement. Des exemples de donnée qui peuvent être stocké dans un fichier RIFF sont :

- Audio/visual interleaved data (.AVI)
- Waveform data (.WAV)
- Bitmapped data (.RDI)
- MIDI information (.RMI)
- Color palette (.PAL)
- Multimedia movie (.RMN)
- Animated cursor (.ANI)

Après la lecture du format RIFF nous aurons un vecteur de donnée à cette fréquence d'échantillonnage. Le traitement sera bien simplifié de cette manière. Ce vecteur sera l'entrée de l'algorithme de segmentation

## 4. Segmentation

### 4.1.Définition de la segmentation :

La segmentation est le processus de division d'un discours continue en unités de base ayant un sens acoustique. C'est une étape importante dans la reconnaissance vocale, et elle joue également un rôle important dans certaines applications. (1)

### 4.2.Les types de segmentation

La segmentation manuelle : elle est effectuée manuellement par l'examen de la forme d'onde du signal de parole avec un spectrogramme. Mais ce processus est très fastidieux, consommateur du temps, d'erreurs et les résultats ne peuvent pas être reproduits. (2)

La segmentation automatique : elle est plus pratique mais difficile à réaliser. Le discours est segmenté automatiquement en unités de mots qui sont définis avec un sens acoustique (3).Elle est jugée meilleure que la segmentation manuelle. Les discours peuvent être efficacement segmentés en ses unités qui sont des mots, des phonèmes ou des syllabes, à l'aide de segmentation automatique. (4)

### 4.3.Les méthodes de segmentation automatique de la parole :

Les méthodes de segmentation automatique de la parole se divisent en deux grandes classes des méthodes :

La première classe englobe toutes les méthodes qui permettent de segmenter un signal de parole sans connaissance a priori du contenu linguistique de ce dernier. Ces méthodes produisent des segments d'un signal de parole en zones spectralement homogènes, qui correspondent généralement à des segments sub-phonétiques.

La deuxième classe englobe toutes les méthodes qui permettent de segmenter un signal de la parole étant donné une description linguistique de ce signal. Ces méthodes de segmentation sont dites avec contraintes linguistiques.

### 4.4.La segmentation sans contraintes linguistiques :

Ces méthodes ont pour objectif de produire une segmentation acoustique du signal de parole sans lien a priori avec le contenu linguistique de ce signal.

Autrement dit, ces méthodes ne fournissent pas un étiquetage linguistique des segments acoustiques qu'elles délimitent. Ces segments acoustiques reflètent cependant la réalité physique du signal car chacun de ces segments représente une zone d'homogénéité (stabilité) spectrale du signal. C'est pourquoi, chacune des méthodes de segmentation de cette classe utilise une mesure de distance entre vecteurs acoustiques ou entre modèles statistiques, permettant de détecter et de délimiter les

segments acoustiques constituant le signal de parole (5). Pour cette classe, nous pouvons citer brièvement quelques méthodes qui existent à nos jours :

### 4.5.Word Chopper technique :

Nishi Sharma a discuté dans son article la technique de la segmentation de la parole en syllabes en utilisant la méthode de Word Chopper (1).Cette technique implique l'extraction des caractéristiques du signal comme la taille, le nombre de canaux, la fréquence d'échantillonnage, la longueur de données et les bits par échantillon. Avec cette technique, les régions de silence peuvent être détectés et coupés et les syllabes sont détectées dans le signal de parole.

### 4.6.Energie à court terme :

La technique de l'énergie à court terme du signal de parole est l'énergie du signal à un instant particulier de temps. Cette méthode différencie entre les parties voisés, non voix et le silence de la parole. L'énergie à court terme est élevée pour la parole voisée, faible pour la parole non voisée et zéro pour le silence (6)

Amanpreet Kaur a décrit une méthode de segmentation de la parole en syllabes en utilisant l'énergie à court terme de discours. Dans cette méthode un certain seuil est sélectionné. Le point de départ de syllabe est détecté avec la valeur ayant plus de valeur que le seuil. Les valeurs inférieures à la valeur de seuil sont comptées comme des zéros et une constante de zéros représentent fin de syllabe. (3)

Runshen Cai a présenté une technique de segmentation des discours en syllabes basée sur des transcriptions phonétiques et des caractéristiques tels que : la moyenne de l'énergie à court terme, le taux de passage par zéro à court terme, le produit de ces 2 derniers, le rapport du premier sur la dernière caractéristique et enfin le rapport de l'énergie moyenne des basses fréquences sur l'énergie moyenne totale (7).

## 5. Modélisation du signal de parole

Le signal de parole est très compliqué en lui-même pour être analyser par un algorithme directement. C'est pour cela qu'il est essentiel de passer par un modèle qui transcrit ce signal en paramètres plus simples qui peuvent être pris comme entré d'un algorithme d'analyse de la paroles. Ce modèle ainsi généré peut aussi être utilisé en sens contraire pour synthétiser de la parole.

Plusieurs algorithmes existent permettant d'effectuer cette tache de modélisation, nous allons voir quelque un des plus connues à savoir, LPC, PLP et MFCC.

### 5.1.LPC

Supposons qu'un signal d'intérêt ait été produit par une source qui excite un filtre linéaire. En radar, la source est une forme d'onde d'impulsion transmise. Ainsi, le filtre peut être modélisé comme une ligne retardée pondérée qui renvoie le signal réfléchi et atténué. Dans l'analyse sismique, une explosion fournit la source, et diverses structures géophysiques agissent comme des filtres. Pour la modélisation de la parole, la source peut représenter des bouffées périodiques d'air traversant la glote (espace entre les cordes vocales), ou une forme d'onde bruyante produite à une étroite constriction dans le tractus vocal et le filtre correspond au tractus vocal supérieur (la langue, les levres, les dents … etc).

La modélisation LPC en utilisant la méthode du moindre carrée induit une perte d'information de phase, comme en témoigne l'utilisation de données d'autocorrélation (qui élimine la phase) pour représenter un signal, qui est acceptable dans la plupart des applications. Par exemple, dans la reconnaissance de la parole, les coefficients LPC captent la forme du tube vocal, ce qui est important pour distinguer les différents sons; La phase rejetée représente principalement l'information temporelle dans l'excitation glottale, qui est peu utile pour discriminer les phonèmes. Cependant, la phase peut être importante pour le discours resynthèses à partir des paramètres LPC; Le langage LPC semble synthétisé ou robotique, tandis que le discours ADPCM (qui conserve la phase en codant le signal d'erreur) ne le fait pas. Les circuits intégrés qui font L'analyse LPC sont suffisamment complexe pour nécessiter des puces de traitement de signal numérique, par exemple, le NEC 7720 ou le TI TMS320, qui font des multiplications en un seul cycle (contrairement aux microprocesseurs standard) (8).

### 5.2.PLP

Afin de visualiser cette méthode un schéma est proposé à la Figure I-6 par (9) et est numéroté à chaque étape, qui est ensuite décrite ci-dessous :

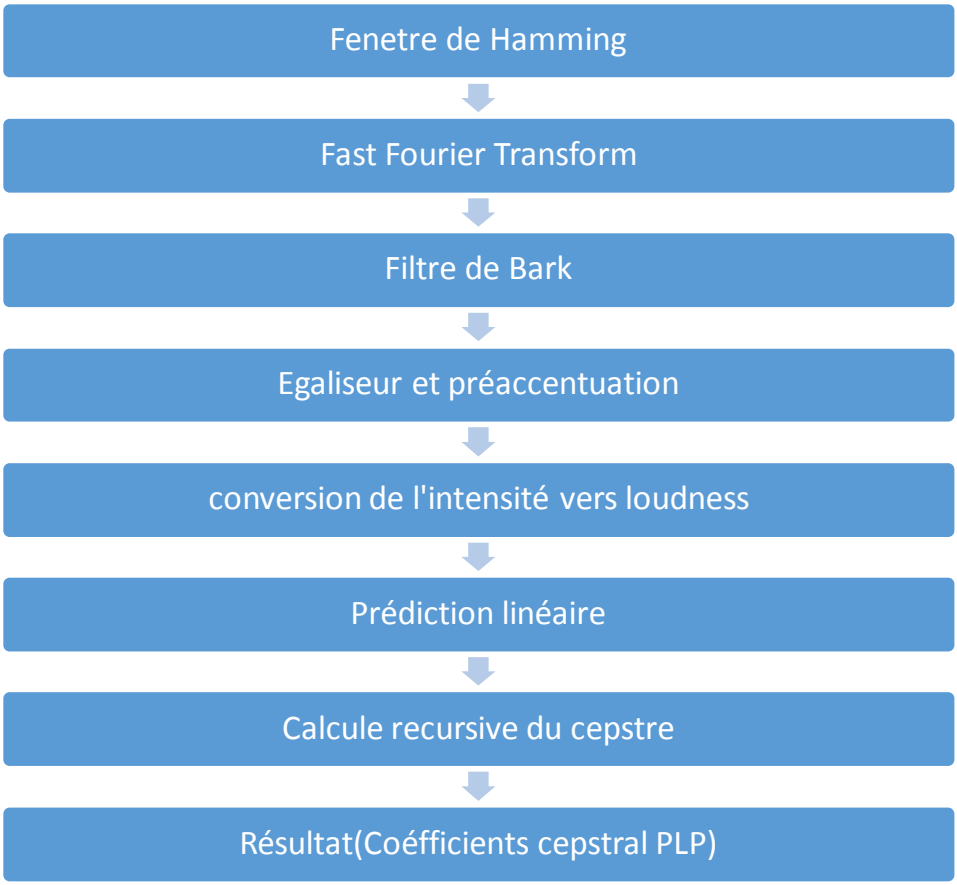


Figure I-6 Les étapes de calcul des coéfficients PLP

(i) Le spectre de puissance est calculé à partir d'une fenêtre appliquée au signal vocal.

(ii) Une déformation de fréquence dans l'échelle Bark est appliquée.

(iii) Le spectre ondulé est convolution avec le spectre de puissance de la courbe de masquage de bande critique simulée pour simuler l'intégration de la bande critique de l'ouïe humaine.

(iv) Le spectre lissé est échantillonné à intervalles réguliers de 1 Bark. La déformation, le lissage et l'échantillonnage de la fréquence de trois étapes (ii-iv) sont intégrés dans une seule banque de filtres appelée banque de filtre Bark.

(v) Une préaccentuation égalisé à l'intensité pondère des sorties de la banque de filtres pour simuler la sensibilité de l'ouïe.

(vi) Les valeurs égalisées sont transformées selon la loi de puissance de Stevens en élevant chacun à la puissance de 0,33.

(vii) Prévision linéaire (LP), l'application de LP au spectre de ligne audible déformé signifie que nous calculons les coefficients de prédiction d'un signal (hypothétique) qui a ce spectre déformé comme spectre de puissance.

(viii) les coefficients cépstrales sont obtenus à partir des coefficients prédictifs par une récurrence qui est équivalente au logarithme du spectre modèle suivi d'une transformée inverse de Fourier. (9)

Le principe sous-jacent de l'analyse PLP consiste à rapprocher le spectre auditif de la parole par un modèle à tous les pôles. Dans cette section, nous avons décrit un moyen calculé d'une manière raisonnablement efficace d'obtenir l'estimation du spectre auditif: convolution du spectre FFT avec la fonction de la bande critique, en la multipliant par une courbe d'intensité égale et en compriment son amplitude par une fonction cube-racine. Les approches techniques des lois psychophysiques ont été les choix personnels des auteurs, souvent dirigés en premier lieu par l'efficacité du calcul. Un certain nombre de phénomènes connus ont été ignoré, par exemple la dépendance de la forme de la bande critique ou de la courbe d'intensité égale sur l'intensité du son qui par leur expérience ne posait pas de différence majeure. (10)

### 5.3.MFCC

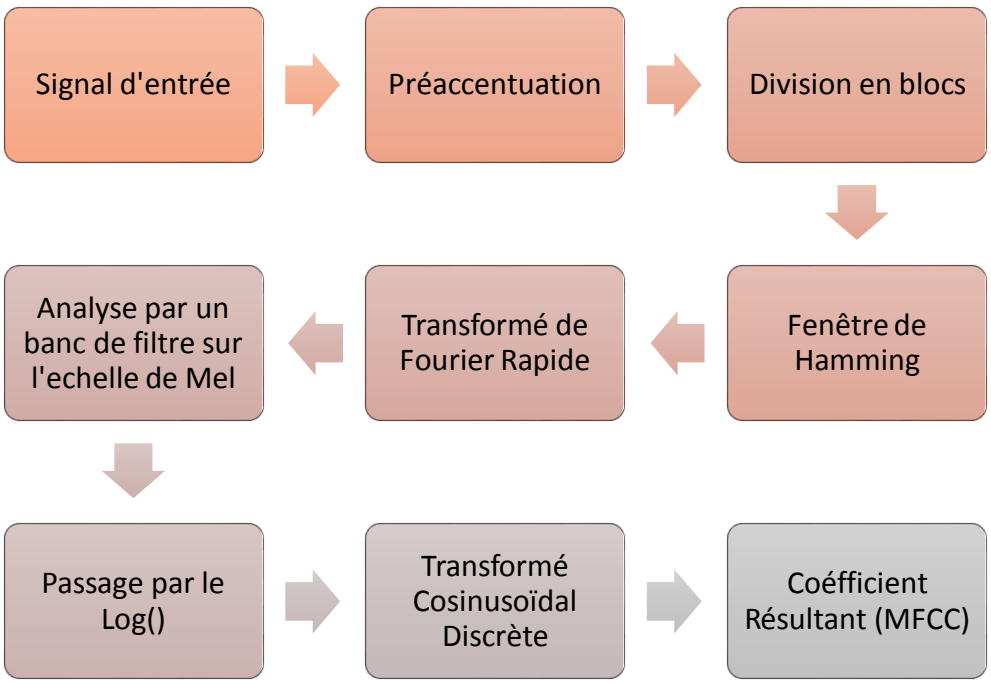


Figure I-7 Schéma de principe du calcul des MFCC. (11)

L'analyse des MFCCs commence par l'application de la Transformée de Fourier Rapide (FFT) sur la séquence de trame afin d'obtenir certains paramètres, en convertissant le spectre de puissance en un spectre de fréquence de Mel, en prenant le logarithme de ce spectre et en calculant sa transformée de Fourier inverse comme le montre la Figure I-7.

Les MFCC ont été standardisé dès les années 2000 pour l'utilisation dans la téléphonie mobile afin d'extraire les paramètres pour la reconnaissance vocale (12).

# Chapitre II

# Implémentations de la Méthode de Segmentation et Calcul des coefficients MFCC

## 1. Introduction

Dans ce chapitre, nous allons décrire l'implémentation de la méthode de segmentation à énergie continue décrite par (13). Le choix de cette algorithme est du à sa simplicité et à sa rapidité grâce à l'utilisation d'opération atomique pour la plateforme cible (x86), c.-à-d. des opérations simple et rapide comme l'addition et la multiplication.

Cet algorithme a été à l'origine en MATLAB que nous l'avons traduit en Pascal Orienté Objet pour Delphi et justifier ce choix de langage et compilateur. Et afin de rendre cet algorithme plus rapide, nous avons choisis de lui appliquer le parallélisme par Thread.

Une fois la segmentation automatique effectuée, nous allons calculer les paramètres MFCC de l'enregistrement vocal ainsi allégé des périodes de silence.

Le calcul des paramètres MFCC sera effectué grâce à l'algorithme de Gang Min et al (14) implémenté en MATLAB, que nous n'avons pas réussi à traduire vu l'abstraction faite dans ce langage pour certaine fonction et que nous avons utilisé dans ce même langage avec des modifications légères afin de convenir aux besoins de notre travail.

## 2. Pascal Orienté Objet (Delphi)

Le choix de ce langage de programmation est non seulement motivé par notre maîtrise de celui-ci, mais aussi par sa performance. En effet, le compilateur de Delphi s'adapte parfaitement à notre plateforme destination qui est Windows x86 et permet donc de gérer les threads efficacement sous celle-ci.

Le code écrit peut inclure des instructions en assembleur afin de rendre certaine opérations plus rapide. Et grâce à la librairie VCL, nous pouvons créer des graphes aisément.

Et avec des modifications mineures (abstraction de la plateforme cible) le code peut être compilé pour des plateformes mobiles afin de s'exécuter sur Android ou IOS.

## 3. Lecture d'un Fichier RIFF Wave

La lecture d'un fichier RIFF commence par la lecture de son entête de 44 octets afin de comprendre sa structure interne et de permettre la lecture de sa partie donnée (DATA).

Le fichier est d'abord chargé dans un flux fichier(TFileStream) pour ensuite être lu octet par octet, comme a été décrit précédemment les données dans un fichier RIFF sont codé en Little-Endian, la première tâche sera donc de convertir les données lu en Big-Endian pour ensuite les stocker dans leurs variables correspondantes.

Les données de la partie

## 4. La Segmentation par la Méthode à Energie Moyenne Continue

Comme décrit précédemment cette méthode est sensible au bruit ce qui nous a motivé pour utiliser des enregistrements non-bruités.

Il est cependant à noter que la plupart des smartphone actuels sont doté de deux microphones afin de supprimer le bruit environnant des enregistrements. Donc cette méthode peut également être utilisée pour des enregistrements vocaux faites dans des milieux bruités et filtrés grâce à la technologie disponible dans ces appareils.

### 4.1.L'Algorithme de Segmentation par la Méthode à Energie Moyenne Continue

Le signal de départ est tout d'abord normalisé puis mis au carrée afin de faire remonter les front négative dans un vecteur A de longeur N, ensuite, une fenêtre de taille T est passé par le signal résultant (vecteur A) et la moyenne de cette fenêtre est mise dans le premier element d'un deuxiemme vecteur B, la fenêtre est ensuit coulissé d'un element sur le vecteur A, et le deuxiemme element du vecteur B est calculé de la même manière jusqu'à l'element à l'indice N-T.

L'ecart type du vecteur B est alors calculé et la moyenne est soustraite à chaque valeur de B pour enfin la diviser par l'ecart type dans un vecteur C.

Chaque valeur de C est ensuite comparée à la moyenne qui est établi comme seuil et aux indices auquel correspondent des valeurs supérieurs, il est déduit que c'est un sous ensemble voisé.

Il est évidents que plusieurs opérations peuvent être effectuées sur le même vecteur, nous les avons séparé et nommé différemment par souci de clarté.

Nous présentons un organigramme de l'algorithme dans la Figure II-1

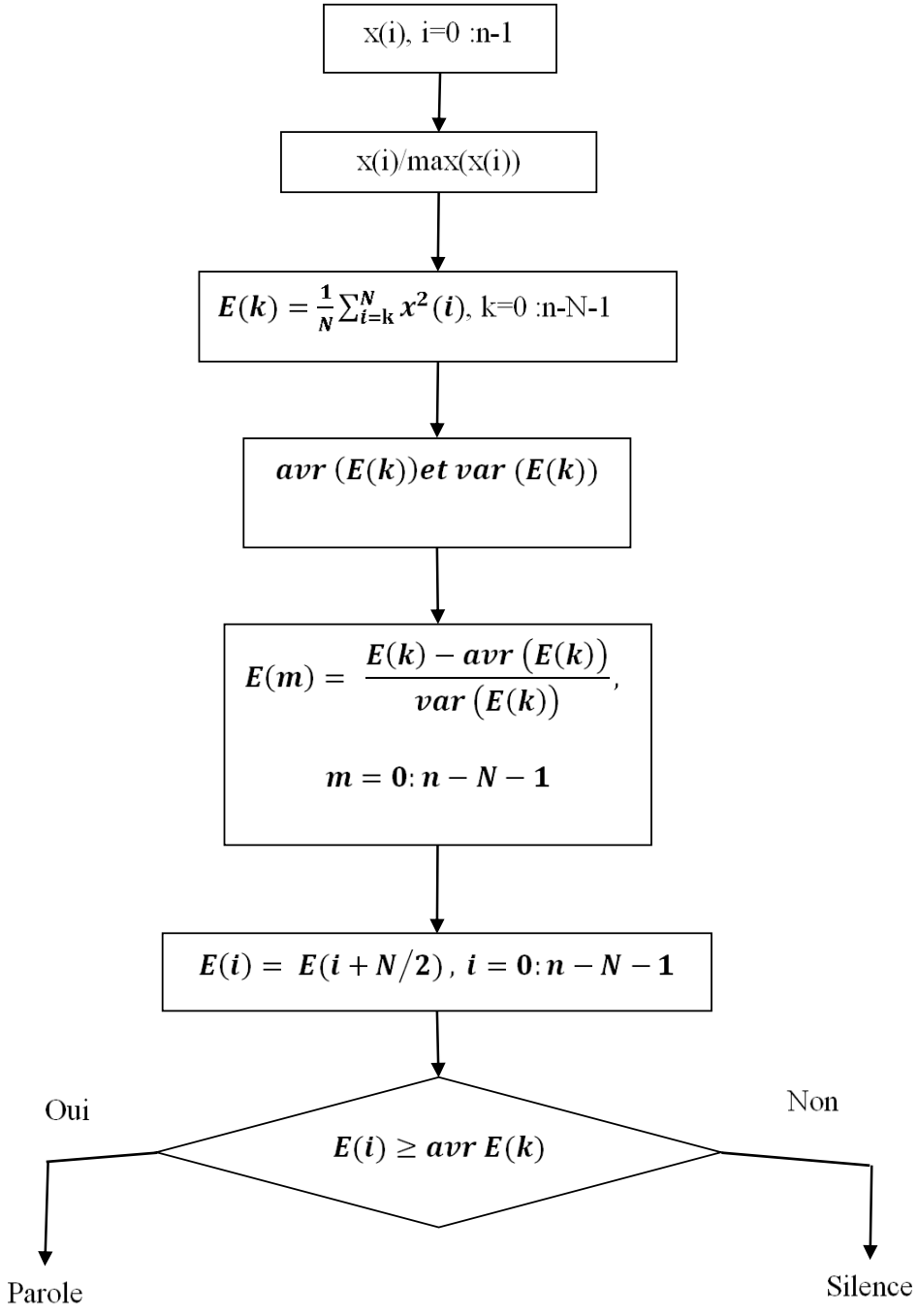


Figure II-1 Organigramme de la méthode à énergie moyenne continue

### 4.2.Parallélisme de l'Algorithme de Segmentation à Energie Moyenne Continue

Il est évident que la tâche la plus lourde durant le déroulement de cet algorithme est le calcul du vecteur B qui calcule n-N somme de N éléments donc (n-N)*N opération élémentaire. Pour donner un exemple en nombre, un enregistrement de 10 secondes à une fréquence d'échantillonnage de 48000 KHz et avec chaque échantillon codé sur 2 octets, et une fenêtre de 2400 échantillon nous aurons :

$$48000 \times 2 \times 10 \times 2400 = 1{,}843{,}200{,}000$$

Donc un peu plus de 1.8 Giga Opération qui est énorme.

Il devient donc assez logique de distribuer cette tâche sur plusieurs threads pour ensuite évaluer la performance.

Le vecteur B est donc divisé également entre les différents threads, il est à rappeler que la taille du vecteur B n'est pas de n, mais de n-N, Il est également à noter que l'élément de départ d'une fenêtre chevauche N-1 élément de fin du thread qui traite la partie en amont du thread en question et vice versa avec le thread qui traite la partie en avale. Cela est visible dans la Figure II-2

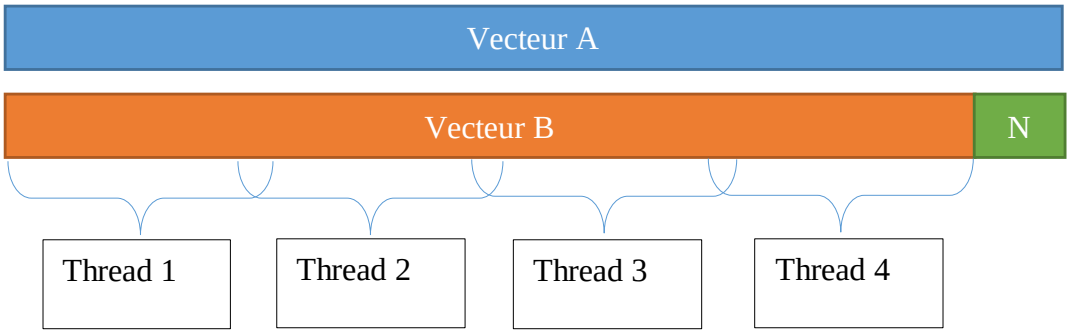


Figure II-2 Division des tâches inter-thread.

#### 4.2.1. Analyse de la Dépendance pour le Parallélisme

Afin d'éviter les conflits inter-thread il est essentiel de vérifier la dépendance des différentes variables en lecture mais surtout en écriture et définir si besoin des points de synchronisation.

La variable B est partagée en lecture seul entre les threads, et comme la plateforme cible (x86) supporte la lecture en parallèle d'une même variable, il n'y a pas d'inter blocage inter thread.

Une fois la somme calculée chaque thread dépose le résultat dans un autre vecteur V cette fois bien qu'il y est une écriture, chaque thread écrit une valeur dans une case mémoire indépendante des autres, là encore, il n'y a pas de problème d'inter blocage inter thread.

La majorité du travail de notre code se déroule en série dans le thread principal du programme, donc il est essentiel d'attendre que toutes les threads se termine avant de continuer le déroulement du code, pour cela l'ajout d'une barrière de synchronisation est indispensable après la création de toutes les threads dans le thread principal.

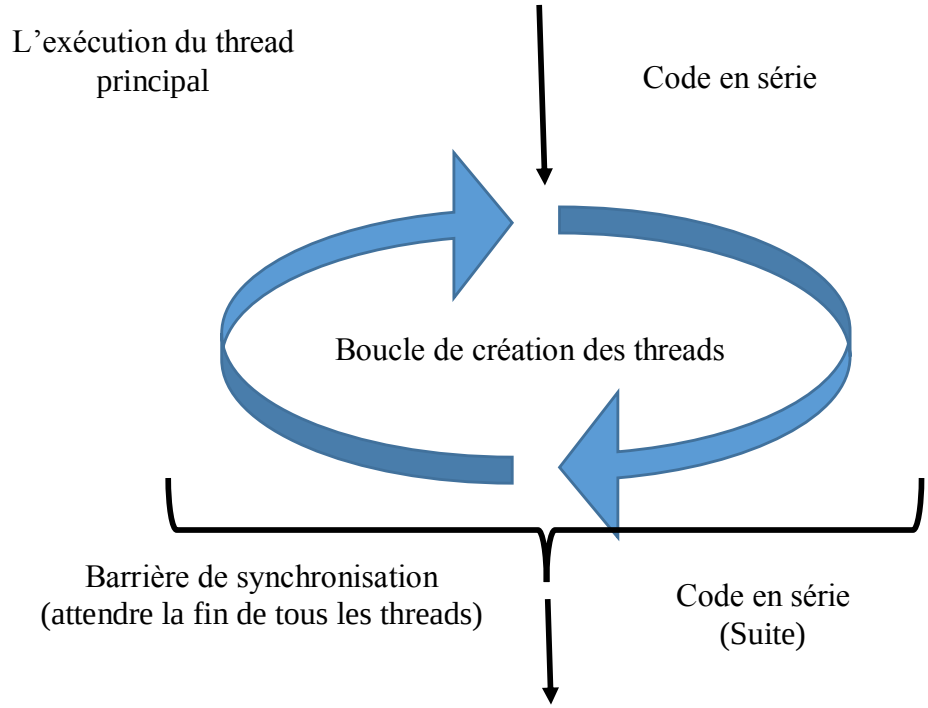


Figure II-3 Position de la barrière de synchronisation dans le thread principal

### 4.3.Résultat de l'algorithme

A la fin de cet algorithme, un fichier WAV est créé contenant les segments voisé seulement une écoute de celui-ci sera dès lors possible ainsi que le calcul des paramètres MFCC

## 5. Le Calcule MFCC et son Inversion

L'algorithme choisis permet non seulement le calcul des paramètres MFCC mais aussi la reconstruction à partir de ces derniers d'un fichier WAV.

### 5.1.L'Algorithme de calcule MFCC

#### 5.1.1. Calcul STFT

Le code commence par créer une fenêtre de taille L qui par défaut est de 256 élément afin de faire calculer la Transformé de Fourier à Court Terme (TFCT ou STFT en Anglais). Les résultats de cette opération sont des nombres complexes, le module de ces nombres est donc pris.

#### 5.1.2. Création du filtre de Mel

Et afin que les résultats soient plus parlants à l'oreille humaine, les fréquences résultantes de l'application du TFCT sont transformées sur l'échelle de Mel, grâce à la formule (1)

$$m = 1127 \times \ln\left(1 + \frac{f}{700}\right) \qquad \text{Équation II-1}$$

Ou f est la fréquence normal est m est la projection de cette fréquence sur l'échelle de Mel.

Diverse équations ont vu le jour pour effectuer cette projection mais l'auteur à choisis celle de Douglas O'Shaughnessy (1987) l'auteur original de cette recherche.

La fonction log est ensuit appliqué afin de créer le cepstre du signal qui passera par une transformé cosinusoïdal discrète, qui seront les fameux paramètres MFCC.

La Figure II-4 résume les étapes de ce calcul.

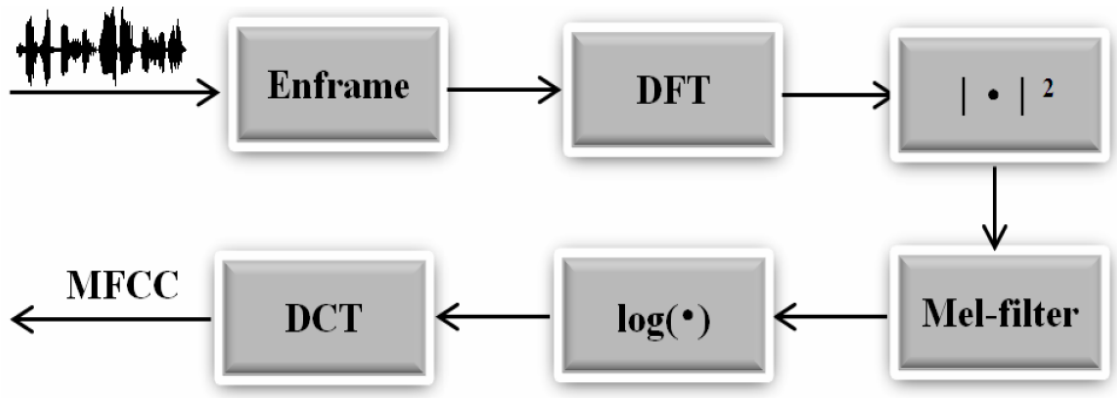


Figure II-4 Le processus d'extraction des MFCC (14)

### 5.2.Le calcule inverse des MFCC

Les MFCC sont normalement testé dans une application donnée (reconnaissance de la parole, reconnaissance du locuteur … etc.), mais vu que ces applications sont hors de portée de ce mémoire de simples tests d'écoute suffiront à évaluer ces paramètres. Il est donc nécessaire de recourir à la reconstruction du signal de parole à partir des MFCC.

#### 5.2.1. Algorithme du calcul inverse depuis les MFCC

L'algorithme commence à calculer les valeurs d'amplitude du signal grâce à la méthode des moindre carrées et en appliquant la méthode de minimisation L2 repondéré itérativement à 300 itération car une valeur supérieure ne donnerai pas de meilleur résultat selon les auteurs (14).

Ensuite, il calcule l'interpolation linéaire par consultation de table des valeurs de l'amplitude pour enfin reconstruire le signal de parole par l'algorithme LSE-ISTFTM (Estimation des moindres carrés, grandeur inverse des transformées de Fourier à courte durée).

Le signal de parole ainsi reconstruit est enregistré dans un fichier Wave pour être écouté.

# Chapitre III

# Résultats de la Simulation

## 1. Introduction

Nous avons maintenant toutes les briques de base de notre programme, nous allons tout d'abord présenter l'application crée pour ensuite présenter les résultats obtenus et l'explication de ceux-ci et enfin nos conclusions.

Premièrement, nous allons tester le programme traduit en Delphi avec celui du Matlab afin de vérifier son intégrité.

Puis nous allons présenter le corpus utilisé dans nos tests.

Par la suite, nous allons tester le programme de segmentation et calculer le ratio de la longueur du fichier original et vitesse de conversion que nous allons dès lors nommer LFO/VC, si celui-ci est supérieure ou égal à un (1), cela veut dire que le fichier est converti plus vite que le temps suffisant pour l'enregistrer donc que cet algorithme peut être applique en temps réel différé, nous insistons ici sur le terme différé car il faut quand même remplir un vecteur pour lui appliquer l'algorithme.

Nous allons aussi présenter les résultats du test de la segmentation en utilisant plusieurs thread, mais vu la longueur que cela peut imputer au document seulement un enregistrement dans chaque catégorie (c.-à-d. mots isolés, paroles continues et long discours) va être traité.

Enfin, nous allons présenter les résultats du calcul des paramètres MFCC et la reconstruction du signal de parole original issue de la segmentation.

## 2. Interface de l'application crée

L'interface de l'application est assez basique, un bouton charger permet de lire le fichier Wave donnée en entrée par son chemin, ensuite un boutton VAD(ou Voice Activity Detection) permettant d'executer la méthode de segmentation du signal de parole par energie continue. Un chechbox permet de choisir si le code doit être exécuté en parallel ou en série dans le main thread, le nombre de thread peut directement être introduit dans la zone de text se trouvant juste sous ce checkbox. Le temps d'execution de l'algorithme est en bas et affiché en seconde.

Enfin, un graphe, en bas, est généré apres l'execution de la ségmentation, ce graphe montre en bleu le signal de paroles normalisé, en orange l'energie moyenne calculé et en vert les ségments de paroles. Ce graph pourra être enregisté par le boutton enregistrer graph.

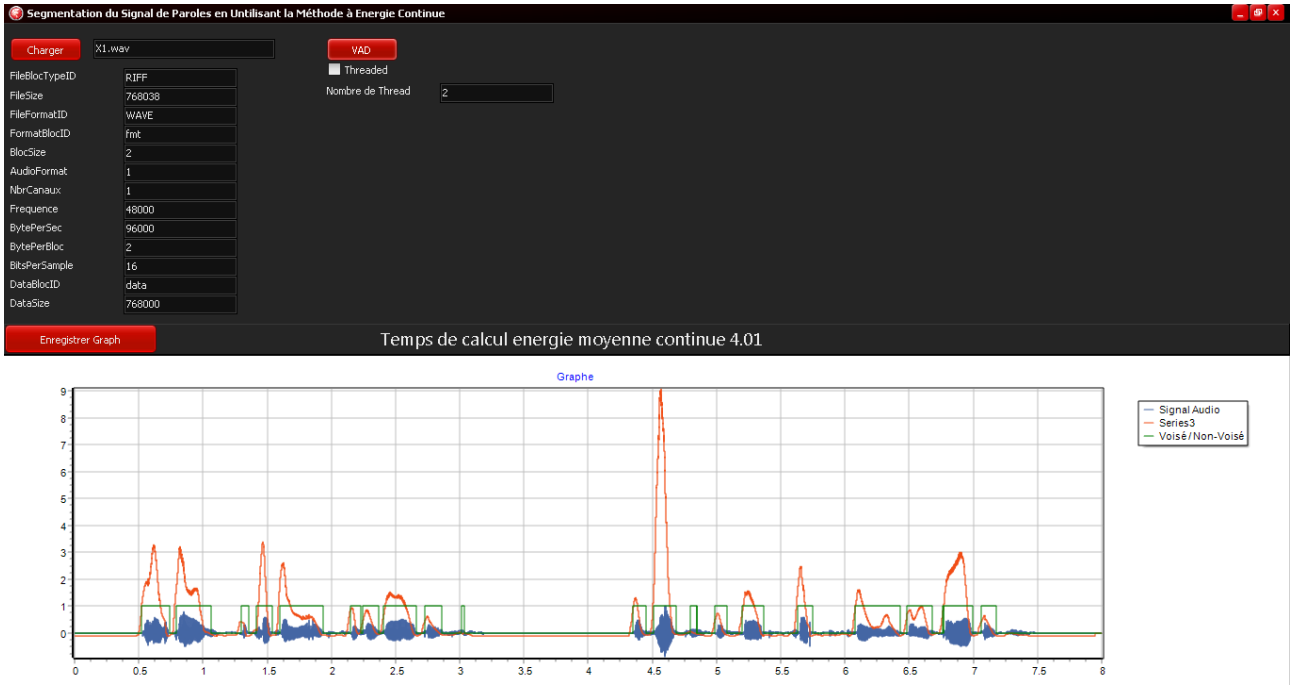


Figure III-1 L'interface du logiciel crée avec Delphi.

## 3. Test de l'Intégrité du Programme Traduit

Afin de tester l'intégrité du programme traduit nous allons soumettre le même fichier Wave aux deux versions du logiciel, à savoir la version MATLAB et la version que nous avons traduite en Delphi, ensuite, nous allons superposer les deux graphes obtenu pour voir leur concordance.

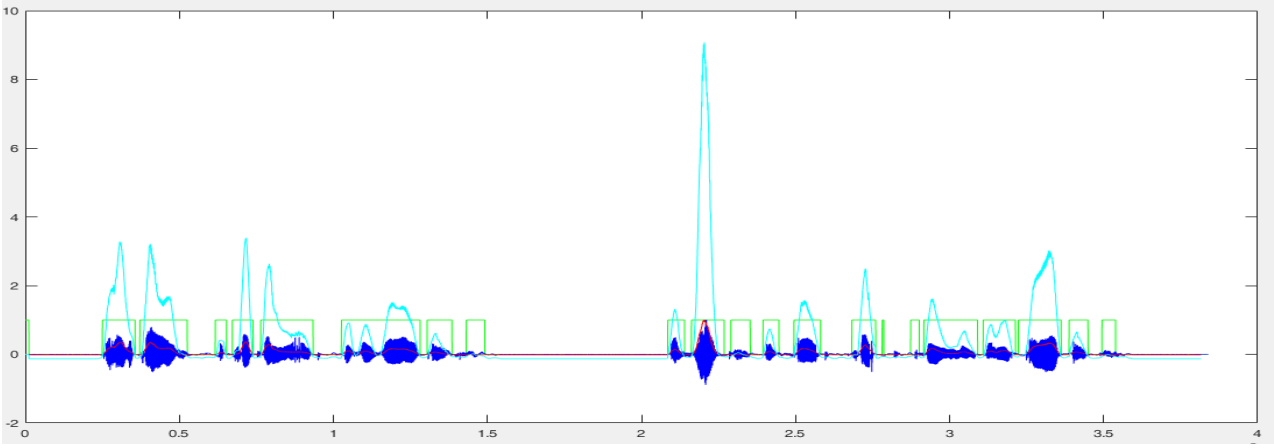

Figure III-2 Graphe de la segmentation obtenu par MATLAB

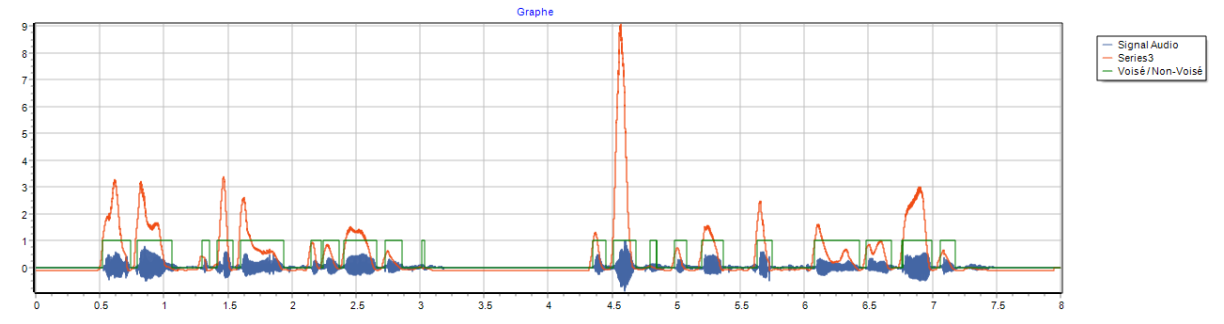


Figure III-3 Graphe de la segmentation obtenu par le code traduit en Delphi

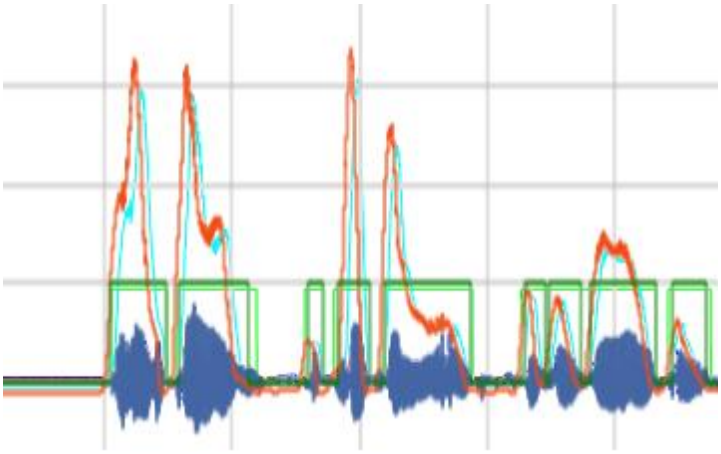

Figure III-4 les deux graphes (de Matlab et Delphi) superposé et zoomer pour pouvoir voir la différence.

A la Figure III-2 nous avons le graphe obtenu par le code MATLAB et à la Figure III-3 nous avons le graphe obtenu par le code traduit en Delphi. Nous avons superposé les deux graphes avec transparence dans un logiciel de traitement de photo et nous avons laissé un petit décalage afin de pouvoir voir les deux graphes en même temps, le résultat est présenté à la Figure III-4.

Nous remarquons que les de graphes se superpose parfaitement, ce qui démontre l'intégrité de la traduction du code MATLAB en Delphi.

## 4. Corpus de test :

Le corpus de test est résumé dans la Table III-1. Ce corpus contient sept (07) enregistrements Wave dans les trois langues : Français, Arabe et anglais. Trois (03) enregistrements sont de mots isolés, des nombres de un à dix, trois (03) autres enregistrement sont de paroles continue de courte durée avec deux phrases séparés par des pauses plus au moins longues. Certains enregistrements sont issues de chambres sourdes, d'autres ont été enregistré par nos soin grâce à un téléphone mobile équipé d'un double microphone afin de minimiser le bruit.

Le dernier enregistrement a été effectué dans une salle de conférence depuis la table de mixage durant la conférence du E-learning School qui s'est déroulé le 20/04/2017, cet enregistrement est intéressant vu que sa qualité n'est pas très bonne et qu'il est très long, ce qui permettra de mesurer la performance de la segmentation sur de long enregistrement.

Il est important de noter que tous les enregistrements essayer ici à la fréquence d'échantillonnage de 48000 Hz, qui est très élevé, car pour la parole une fréquence d'échantillonnage de 8000 Hz est suffisante en générale. Cette fréquence d'échantillonnage va rendre le calcul plus long, ce qui est à prendre en compte durant les tests.

Table III-1 Composition du corpus de test

| Nom du fichier | Type d'enregistrement | Langue | Longueur (seconds) | Source |
|---|---|---|---|---|
| **Nombrear.wav** | Mot isolé (Nombre 1 à 10) | Arabe(Daridja) | 10.536 | ALG-DARIDJAH |
| **Nombrefr.wav** | Mot isolé (Nombre 1 à 10) | Français | 19.52 | Enregistré par nos soins Par Téléphone |
| **Nombrean.wav** | Mot isolé (Nombre 1 à 10) | Anglais | 13.636 | audioblocks.com |
| **CSar.wav** | Paroles continue (2 Phrase) | Arabe(Daridja) | 7.4 | ALG-DARIDJAH |
| **CSfr.wav** | Paroles continue (2 Phrase) | Français | 10.044 | CNET Lannion, France Telecom |
| **CSan.Wav** | Paroles continue (2 Phrase) | Anglais | 8 | Internet |
| **Elearning.wav** | Long discours (Plusieurs phrases) | Anglais | 99.81 | Enregistrement à la Conférence Elearning School à Laghouat, Algérie |

## 5. Test et Résultat de la segmentation

Pour les tests, nous les avons effectués sur un PC portable HP Probook470 avec la spécification qui est présenté dans la Table III-2.

| | |
|---|---|
| Processeur | Intel Core I7-3632QM cadencé à 2.20GHz |
| Nombre de Thread | 8 |
| Mémoire | 8192MB |
| OS | Windows 7 64 bits |

Table III-2 Caractéristique de l'ordinateur de test.

### 5.1.Test sur Mots Isolé

#### 5.1.1. Test sur la langue Arabe (Daridja)

##### a. Test Mono Thread

En un seul thread le graph résultant est donné en Figure III-5, l'exécution de l'algorithme principal a pris 4.97 secondes, la longueur du fichier source est de 10.536 seconds et la longueur du fichier résultant après la segmentation est de 3.927 secondes et ce dernier est assez compréhensible.

Le LFO/VC est de 2.12 ce qui signifie que le traitement se fait 2.12 fois plus vite que le temps d'enregistrement.

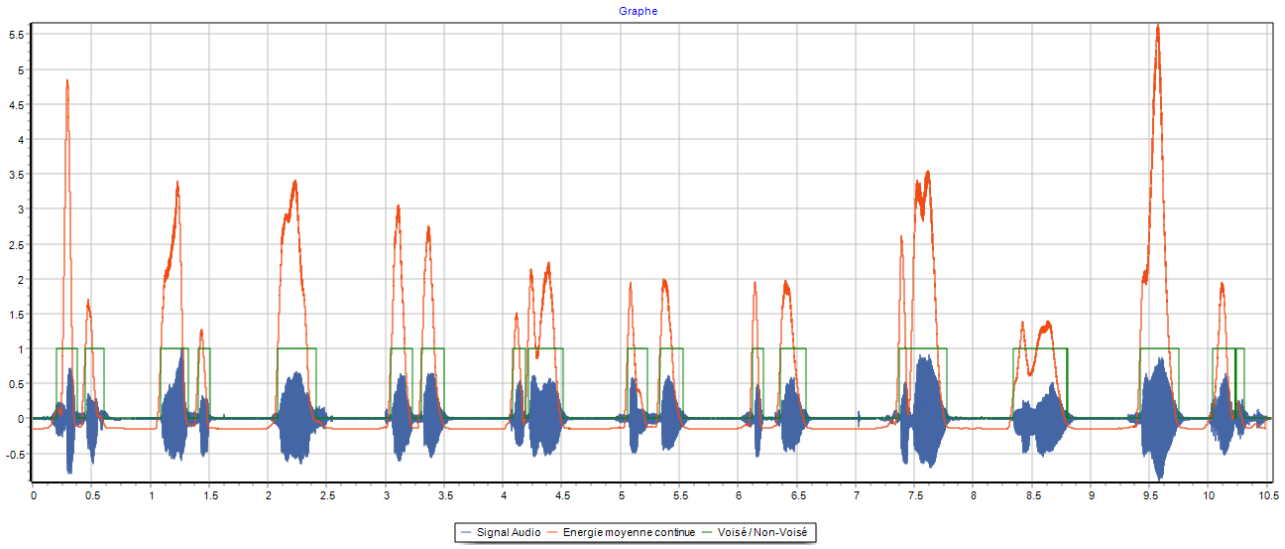


Figure III-5 Segmentation sur enregistrement de mots isolé en arabe (nombre de 1 à 10)

##### b. Test Multi Thread

| Nombre de Threads | Temps de Traitement (en secondes) |
|---|---|
| 2 | 2.51 |
| 4 | 1.75 |
| 8 | 1.43 |

Table III-3 Résumé des tests multithread pour les mots isolés

### 5.1.2. Test sur la langue Française

| | |
|---|---|
| Longueur de l'enregistrement | 19.52 secondes |
| Temps de traitement | 9.22 secondes |
| Longueur du fichier après traitement | 3.11 secondes |
| LFO/VC | 2.12 |
| Compréhensibilité | Bonne |

Table III-4 Performance du Programme sur mots isolés en Français

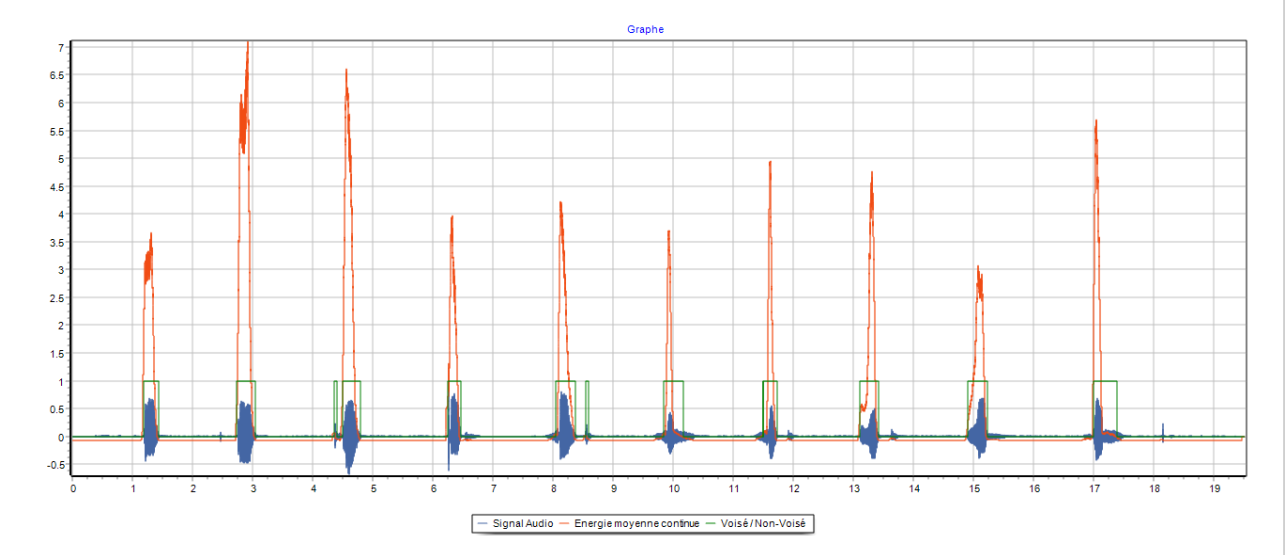


Figure III-6 Segmentation sur enregistrement de mots isolé en Français (nombre de 1à 10)

En un seul thread le graph résultant est donné en Figure III-6. Le temps de traitement est de 9.22 secondes, la longueur du fichier résultant après la segmentation est de 3.11 secondes et ce dernier est assez compréhensible vu la bonne qualité de l'enregistrement.

Le LFO/VC est de 2.12 ce qui signifie que le traitement se fait 2.12 fois plus vite que le temps d'enregistrement.

### 5.1.3. Test sur la langue Anglaise

| | |
|---|---|
| Longueur de l'enregistrement | 13.64 secondes |
| Temps de traitement | 6.24 secondes |
| Longueur du fichier après traitement | 3.07 secondes |
| LFO/VC | 2.18 |
| Compréhensibilité | Très bonne |

Table III-5 Performance du Programme sur mots isolés en Anglais

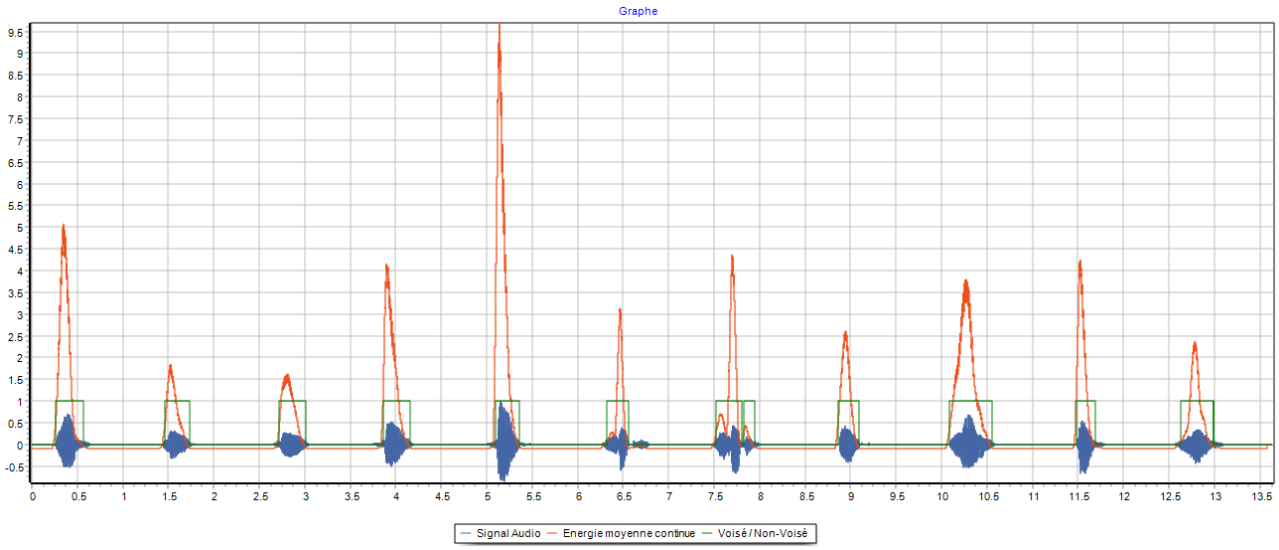


Figure III-7 Segmentation sur enregistrement de mots isolé en Anglais (nombre de 1à 10)

En un seul thread le graph résultant est donné en Figure III-7. Le temps de traitement est de 6..24 secondes, la longueur du fichier résultant après la segmentation est de 3.07 secondes et ce dernier est parfaitement compréhensible.

Le LFO/VC est de 2.18 ce qui signifie que le traitement se fait 2.18 fois plus vite que le temps d'enregistrement.

## 5.2.Test sur Parole continue

### 5.2.1. Test sur la langue Arabe (Daridja)

| | |
|---|---|
| Longueur de l'enregistrement | 7.40 secondes |
| Temps de traitement | 3.47 secondes |
| Longueur du fichier après traitement | 4.83 secondes |
| LFO/VC | 2.13 |
| Compréhensibilité | Moyenne |

Table III-6 Performance du Programme sur parole continue en Arabe (Daridja)

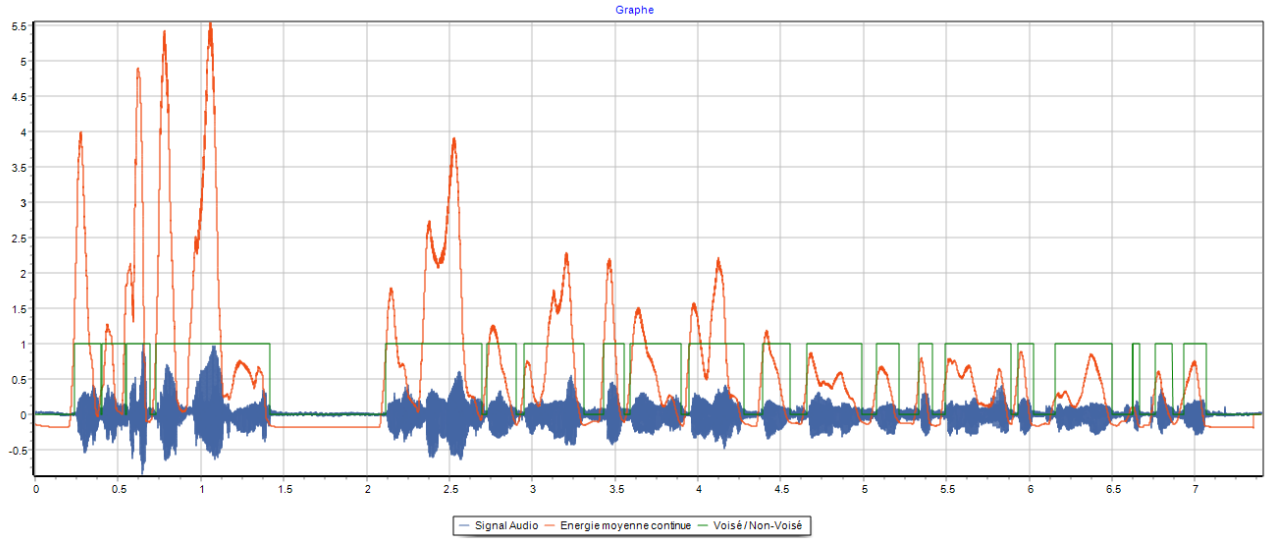


Figure III-8 Segmentation sur parole continue en Arabe (Daridja).

En un seul thread le graph résultant est donné en Figure III-8. Le temps de traitement est de 3.47 secondes, la longueur du fichier résultant après la segmentation est de 4.83 secondes et ce dernier est difficilement compréhensible vu la faible qualité de l'enregistrement.

Le LFO/VC est de 2.13 ce qui signifie que le traitement se fait 2.13 fois plus vite que le temps d'enregistrement.

### 5.2.2. Test sur la langue Française

#### a. Test Mono Thread

| | |
|---|---|
| Longueur de l'enregistrement | 10.04 secondes |
| Temps de traitement | 4.75 secondes |
| Longueur du fichier après traitement | 3.86 secondes |
| LFO/VC | 2.11 |
| Compréhensibilité | Très bonne |

Table III-7 Performance du Programme sur parole continue en Français en mono thread.

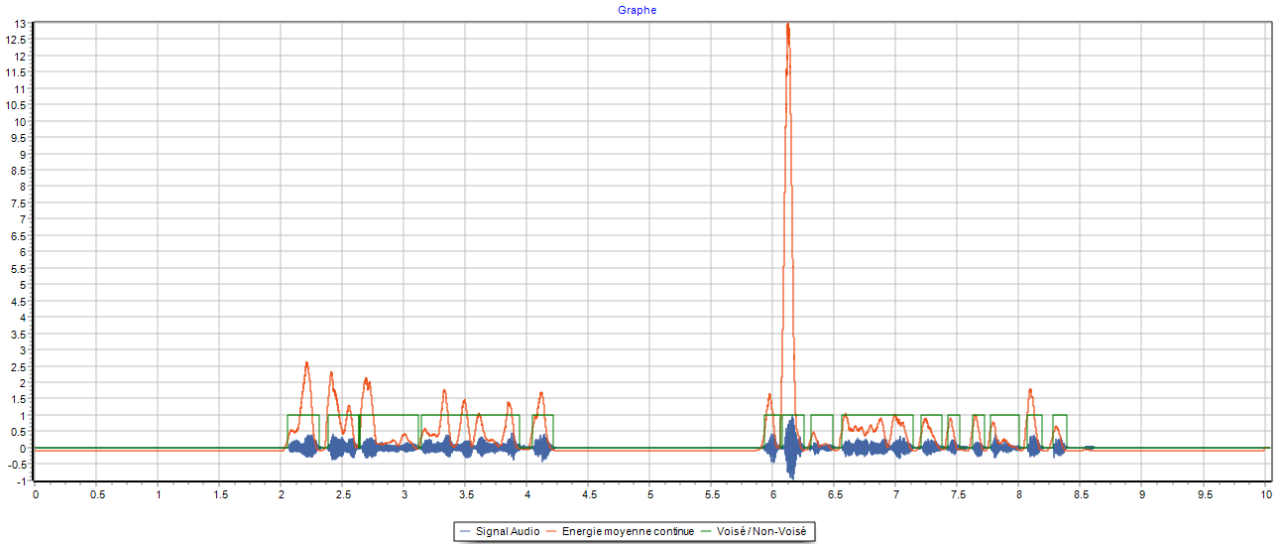


Figure III-9 Segmentation sur parole continue en Français.

#### b. Test Multi Thread

| Nombre de Threads | Temps de Traitement (en secondes) |
|---|---|
| 2 | 2.42 |
| 4 | 1.82 |
| 8 | 1.55 |

Table III-8 Résumé des tests multithread pour la parole continue.

### 5.2.3. Test sur la langue Anglaise

| | |
|---|---|
| Longueur de l'enregistrement | 8 secondes |
| Temps de traitement | 3.67 secondes |
| Longueur du fichier après traitement | 2.36 secondes |
| LFO/VC | 2.18 |
| Compréhensibilité | Bonne |

Table III-9 Performance du Programme sur parole continue en Anglais.

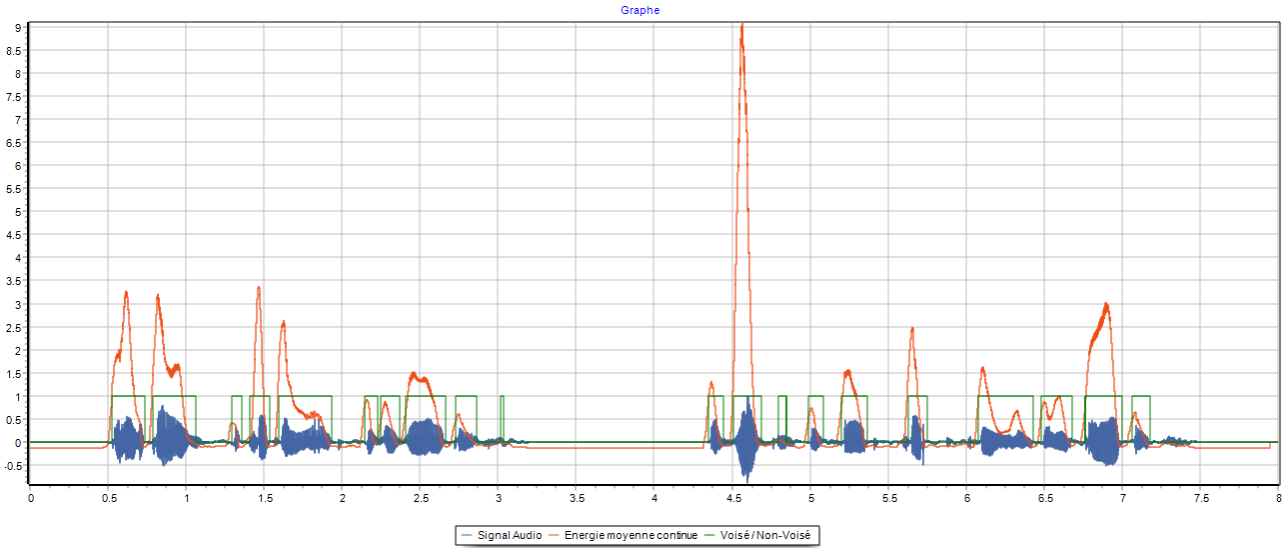


Figure III-10 Segmentation sur parole continue en Anglais.

## 5.3.Test sur Fichier Long

### a. Test Mono Thread

| | |
|---|---|
| Longueur de l'enregistrement | 99.81 secondes |
| Temps de traitement | 46.49 secondes |
| Longueur du fichier après traitement | 40.84 secondes |
| LFO/VC | 2.15 |
| Compréhensibilité | Bonne |

Table III-10 Performance de la segmentation sur fichier long.

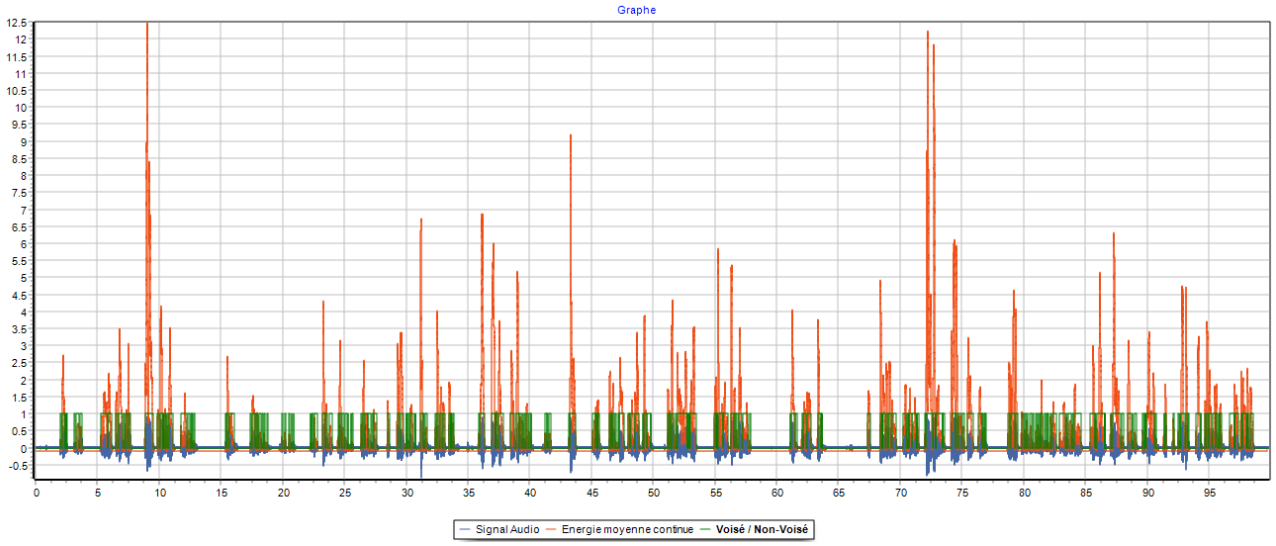


Table III-11 Graphe de la segmentation du fichier long.

### b. Test Multi Thread

| Nombre de Threads | Temps de Traitement (en secondes) |
|---|---|
| 2 | 15.85 |
| 4 | 13.5 |
| 8 | 11.05 |

Table III-12 Résumé des tests multithread pour le fichier long

### 5.4.Résumé des Performances

Nous remarquons tout d'abord que le LFO/VC reste stable autour 2.13, ce qui veux dire que cette algorithme est assez rapide pour faire du traitement en temps réel différé, il ne faut tout de même pas oublier que le taux d'échantillonnage choisi ici est très élevé de 48000 Hz.

Le fichier résultant reste assez compréhensible à l'écoute, même si cela diffère d'un enregistrement à un autre, cela est principalement due à la qualité de l'enregistrement la prononciation du locuteur et de la langue car certaine langues sont plus articulé que d'autre.

| Fichier | Nombrear.wav | Nombrefr.wav | Nombrean.wav | CSar.wav | CSfr.wav | CSan.wav | Elearning.wav |
|---|---|---|---|---|---|---|---|
| Type | Mots isolé | | | Parole continue | | | Fichier Long |
| Langue | Arabe | Française | Anglaise | Arabe | Française | Anglaise | Anglaise |
| Longueur de l'enregistrement (secondes) | 10.54 | 19.52 | 13.64 | 7.4 | 10.04 | 8 | 99.81 |
| Temps de traitement (secondes) | 4.97 | 9.22 | 6.24 | 3.47 | 4.75 | 3.67 | 46.49 |
| Longueur du fichier après traitement (secondes) | 3.93 | 3.11 | 3.07 | 4.83 | 3.86 | 2.36 | 40.84 |
| LFO/VC | 2.12 | 2.12 | 2.19 | 2.13 | 2.11 | 2.18 | 2.15 |
| Compréhensibilité | | Bonne | Très bonne | Moyenne | Très bonne | Bonne | Bonne |

Table III-13 Résumé des performances de la segmentation.

Les performances en parallèle augmentent à une échelle logarithmique à cause de travail nécessaire à la création des threads (en Anglais *over head work*).

Cependant, ces résultats ont une très nette amélioration de la vitesse de la segmentation comparée à la vitesse obtenue en mono thread.

## 6. Test et Résultat du calcul MFCC

Le programme utilisé pour calculer les MFCC et pour la reconstruction de ceux-ci en signal de parole contient plusieurs paramètres que nous allons présenter ici et définir leurs valeurs optimal afin d'obtenir les MFCC les plus compacte que possible pour une éventuel transmission mais aussi les plus efficaces lors de la reconstruction.

Sans oublier, que le calcul MFCC et la reconstruction doivent s'effectuer avec le minimum de ressource possible afin de permettre de les implémenter sur une plateforme mobile.

Le premier paramètre, et sans doute le plus important pour les deux critères décrit précédemment, est le nombre des coefficients MFCC, plus il augmente, plus la taille de la matrice de segmentation augmente qui sera un handicap lors de la transmission de cette matrice. En revanche si le nombre de ces coefficients est trop faible la reconstruction devient incompréhensible.

La taille de cette matrice en octet se calcule comme suite :

$$T = n \times \frac{t}{F} * 8$$

Ou,

- T est la taille de la matrice en octet.
- n est le nombre de paramètre MFCC
- t est la taille du vecteur du signal de paroles
- F est la taille de la fenêtre prise pour le calcul des MFCC
- Et 8 est la taille en octet d'un double.

Les autres paramètres seront pris comme il a été définit dans (15).

Pour éviter la redondance nous allons effectuer les tests sur un fichier déjà segmenté de chaque catégorie, donc faire le calcul des MFCC pour le nombre de coefficient de 10, 14, 20, 50 et 100 pour ensuite mesurer le temps pris lors de cette tâche et noter la taille de la matrice MFCC et la taille du fichier source et calculer le ratio de la taille du fichier original sur la taille de la matrice MFCC abrévié TFO/TMMFCC.

Nous pourrons ainsi commenter les résultats et les analyser.

Après avoir acquis la table MFCC nous allons faire la reconstruction en utilisant le nombre d'itération de 300 pour l'algorithme LSE-ISTFTM parce que cette valeur a été jugé maximale, donc, son augmentation n'augmente en rien la qualité de la reconstruction comme le montre la Figure III-11.

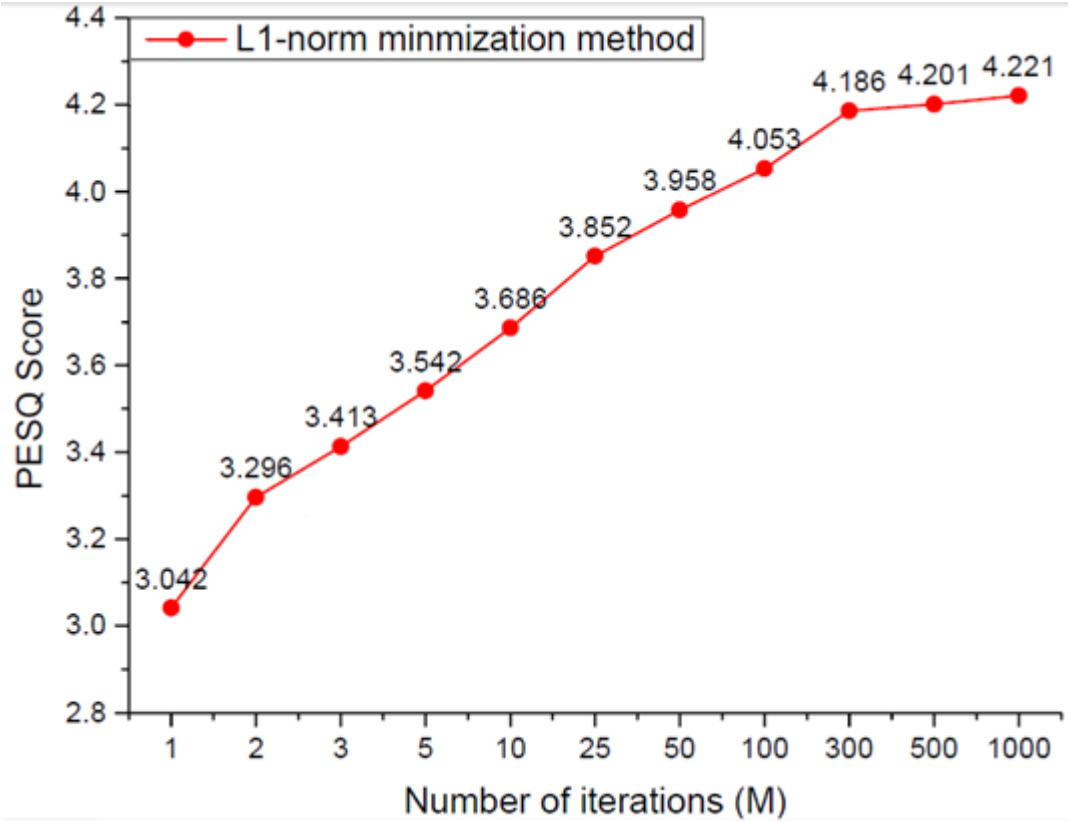


Figure III-11 Valeurs du score *Perceptual Evaluation of Speech Quality* PESQ par rapport au nombre d'itération de l'algorithme LSE-ISTFTM. (14)

### 6.1.Test sur Mots Isolé en langue Française segmenté

| Taille du fichier source (octet) | 298956 | | | | |
|---|---|---|---|---|---|
| Nombre de Coefficient MFCC | 10 | 14 | 20 | 50 | 100 |
| Taille MFCC (Octets) | 93280 | 130592 | 186560 | 466400 | 932800 |
| TFO/TMMFCC | 3.20 | 2.29 | 1.60 | 0.64 | 0.32 |
| Temps de calcule MFCC (secondes) | 0.11 | 0.12 | 0.12 | 0.11 | 0.13 |
| Temps de traitement MFCC Inverse | 17.46 | 18.08 | 18.88 | 33.22 | 40.73 |
| Compréhensibilité | Moyenne | Moyenne | Bonne | Très Bonne | Très Bonne |

Table III-14 Résumé des tests sur mots isolé en langue Française segmenté.

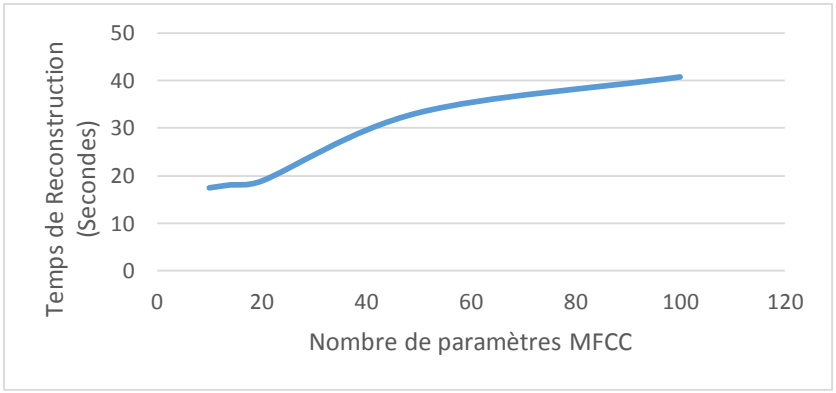


Figure III-12 Temps de reconstruction du signal de parole vs nombre de paramètres MFCC pour la langue Française pour mots isolé.

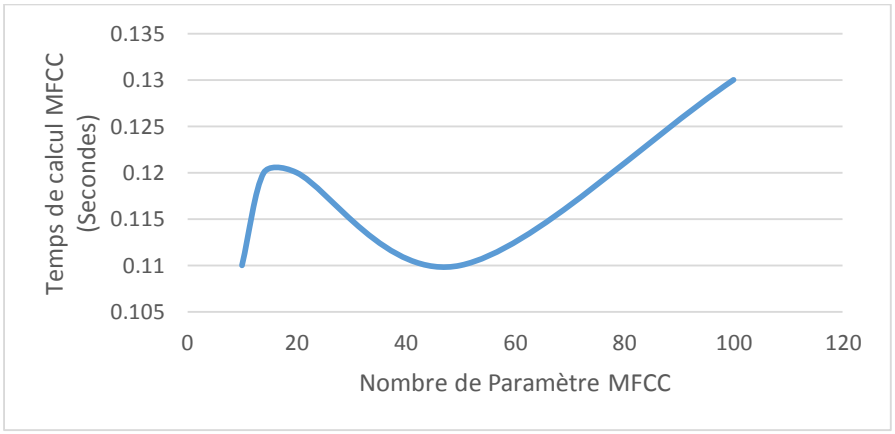


Figure III-13 Temps de calcul MFCC vs le nombre de paramètre pour la langue Française pour mots isolé.

### 6.2.Test sur Parole continue en Arabe (Daridja) segmenté

| Taille du fichier source (octet) | 464118 | | | | |
|---|---|---|---|---|---|
| Nombre de Coefficient MFCC | 10 | 14 | 20 | 50 | 100 |
| Taille MFCC (Octets) | 144880 | 202832 | 289760 | 724400 | 1448800 |
| TFO/TMMFCC | 3.20 | 2.29 | 1.60 | 0.64 | 0.32 |
| Temps de calcule MFCC (secondes) | 0.12 | 0.13 | 0.13 | 0.14 | 0.17 |
| Temps de traitement MFCC Inverse | 28.07 | 29.83 | 34.34 | 66.77 | 111.89 |
| Compréhensibilité | Moyenne | Moyenne | Bonne | Très Bonne | Très Bonne |

Figure III-14 Résumé des tests en langue Arabe (Daridja) segmenté pour parole continue à deux phrases.

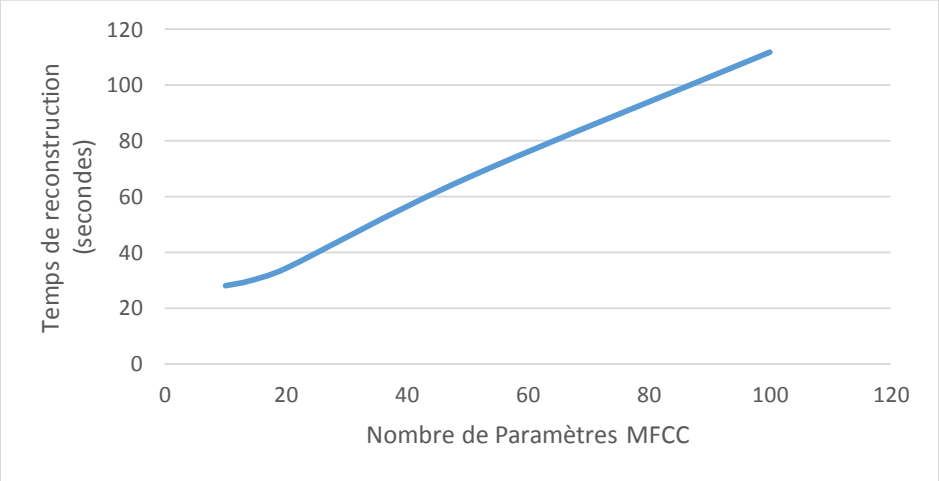


Figure III-15 Temps de reconstruction du signal de parole vs nombre de paramètres MFCC pour la langue Arabe (Daridja) discours continu à deux Phrases.

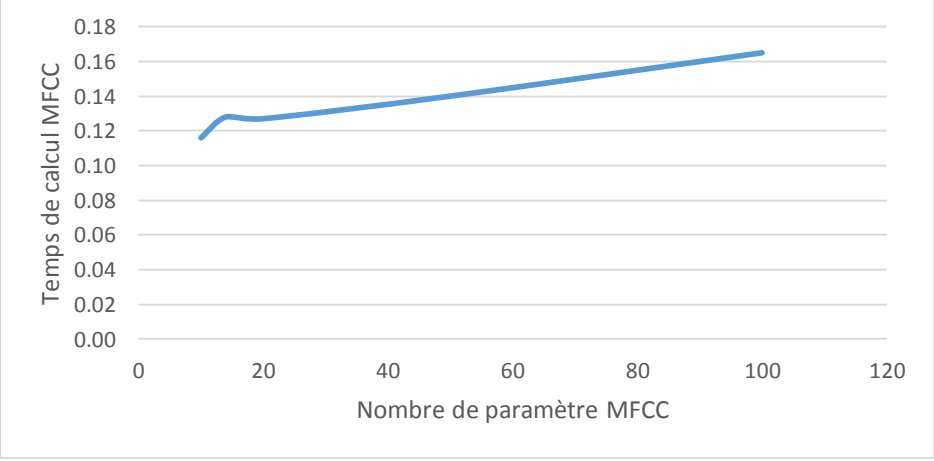


Figure III-16 Temps de calcul MFCC vs le nombre de paramètre pour la langue Arabe (Daridja) discours continu à deux Phrases.

### 6.3.Test sur Fichier Long en Anglais segmenté

| Taille du fichier source (octet) | 3920772 | | | | |
|---|---|---|---|---|---|
| Nombre de Coefficient MFCC | 10 | 14 | 20 | 50 | 100 |
| Taille MFCC (Octets) | 1225120 | 1715168 | 2450240 | 6125600 | 12251200 |
| TFO/TMMFCC | 3.20 | 2.29 | 1.60 | 0.64 | 0.32 |
| Temps de calcule MFCC (secondes) | 0.689 | 0.70 | 0.72 | 0.87 | 1.07 |
| Temps de traitement MFCC inverse | 231.78 | 242.99 | 259.37 | 416 | 554.23 |
| Compréhensibilité | Moyenne | Moyenne | Bonne | Très Bonne | Très Bonne |

Table III-15 Résumé des resultats des tests sur fichier long en langue Anglaise segmenté

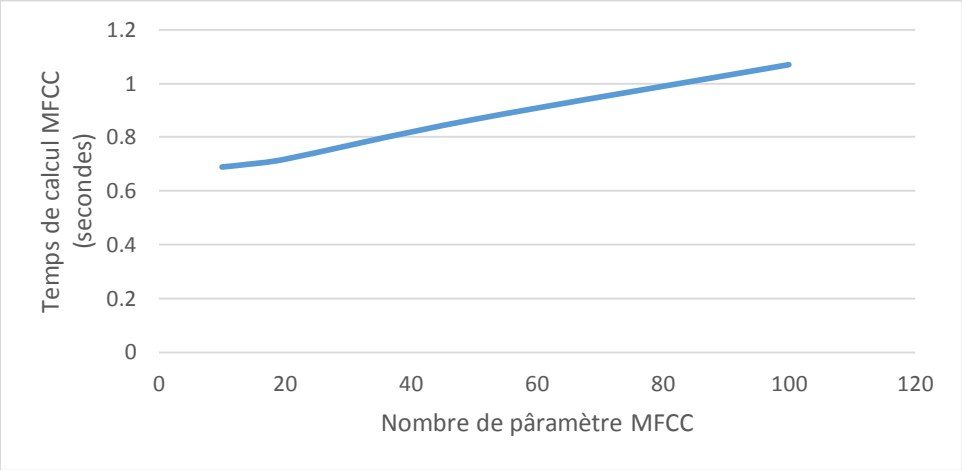


Figure III-17 Temps de calcul MFCC vs le nombre de paramètre pour la langue Anglaise fichier long.

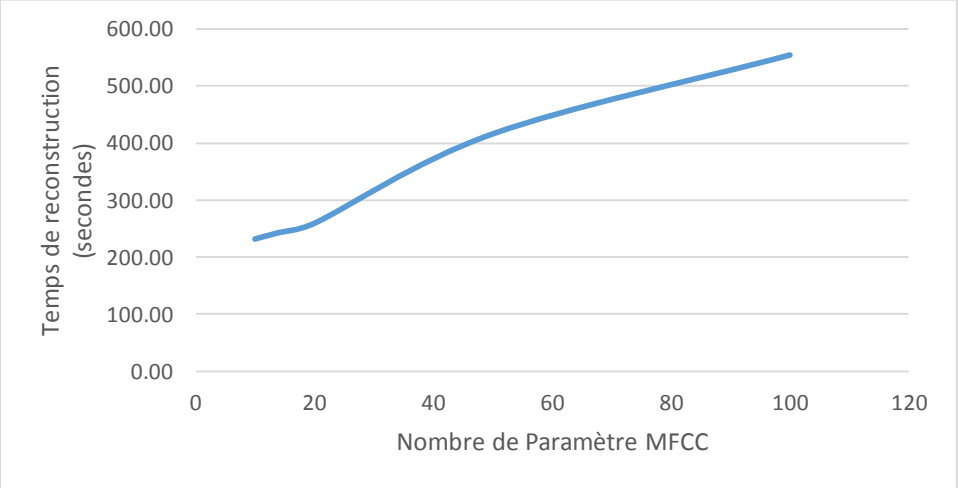


Figure III-18 Temps de reconstruction du signal de parole vs nombre de paramètres MFCC pour la langue Anglaise fichier long.

### 6.4.Résumé des Performances

Les temps de calcul des MFCC sont très courts par rapport au temps de reconstruction. On remarque sur les graphes aux Figure III-17 et Figure III-18 que le temps est linéaire pour les deux tâches à savoir le calcul MFCC et la reconstruction du signal de parole. Ceci n'est pas visible sur les autres graphes du fait que le temps été trop court pour avoir des résultats consistant due à la différence de charge de la machine lors des différents tests.

La taille de la matrice MFCC s'accroit linéairement avec l'accroissement du nombre de paramètres MFCC, et grâce aux valeurs TFO/TMMFCC on sait que au-delà de 20 paramètres MFCC la matrice de ceux-ci devient plus encombrante que le fichier original échantillonné à 48000 Hz, donc pour la transmission de celle-ci il vaut mieux utiliser des valeurs inférieur dépendamment de la qualité de la reconstruction désiré.

La qualité de la reconstruction reste robotique même avec cent (100) coefficients MFCC. La qualité est la même à l'écoute pour cinquante (50) coefficients et cent (100) coefficients.

La meilleure valeur intermédiaire à notre avis reste de vingt (20) coefficients même si cela reste en parti suggestif.

# Conclusion Générale

# Conclusion Général

Dans le premier chapitre, nous avons étalé une préface au traitement de signal et plus spécifiquement à la numérisation d'un signal analogique, nous avons aussi présenté le format RIFF ainsi que sa version WAVE, nous avons aussi survolé différentes méthode de segmentations automatique ainsi que de caractérisation du signal de paroles.

Et dans le deuxième chapitre, nous avons présenté l'algorithme de segmentation automatique à énergie moyenne continue ainsi que sa conversion en Delphi afin de pouvoir lui appliquer le parallélisme, nous avons aussi présenté un code de calcul MFCC ainsi que la reconstruction du signal de parole à partir de ces MFCC calculé.

Enfin dans le troisième et dernier chapitre nous avons résumé les résultats obtenus avec plusieurs fichiers audio de différentes catégories pour la segmentation et le calcul des MFCC que nous avons testé par la reconstruction du signal de parole et l'écoute de celui-ci pour en déduire sa fidélité au signal d'origine. Nous avons ainsi pu monter que le parallélisme permet un gain de temps énorme lors de la segmentation. Et nous avons établi que vingt (20) coefficients MFCC sont largement suffisants afin de reconstruire un signal sonore compréhensible et de petite taille permettant son envoi.

Les perspectives futures sont la réécriture du code de calcul MFCC en langage Delphi et l'inclure au sein du programme de segmentation, pour paralléliser tout le travail et ainsi pouvoir l'implémenter dans une librairie pour plateforme mobile.

# Bibliographie


1. *Automatic Segmentation of Wave File.* **Paraminder Singh, Nishi Sharma.** 2, 2010, International Journal of Computer Science and Communication, Vol. 1.

2. *A Novel Method for Arabic Consonant/Vowel Segmentation Using Wavelet Transform.* **Tolba, M F, et al.** 1, 2005, Faculty of Computer and Information Sciences, Military Technical College, Vol. 5.

3. *Segmentation of Continuous Speech into Syllables.* **Singh, Amanpreet Kaur, Tarandeep.** San Fransisco, USA : s.n., 2010. world congress of Engineering and Computer Science.

4. *Different methods of segmeneting a continuous speech signal into basic units.* **Kaur, Amanpreet.** 11 Nov 2013.

5. *Segmentation automatique de parole en phones. Correlation d'étiquetage par l'introduction de mesures.* **Nefti, Samir.** 16/12/2004.

6. *Automatic silence/ Unvoiced/ Voiced classification of speech.* **Mark Greenwood, Andrew Kinghorn.** Departement of computer science; Univeristy of Sheffield, UK.

7. *An automatic syllable segmentation method for Mandarin Speech.* **Cai, Runshen.** Computer Science and information Engineering College, Tianjin University of Science and Technology,Tianjin, China.

8. *Linear Predictive Coding.* **O'Shaughnessy, Douglas.** 1998, IEEE Potentials, pp. 29-32.

9. *Revising Perceptual Linear Prediction (PLP).* **Florian Hönig, Georg Stemmer, Christian Hacker, Fabio Brugnara.** Lisbon, Portugal : s.n., 2005. INTERSPEECH.

10. *Perceptual linear predective (PLP) Analysis of speech.* **Hermansky, Hynek.** 4, 1990, Acoustical Society of America, Vol. 87, pp. 1738-1752.

11. *Comparative Analysis of MFCC, LFCC, RASTA -PLP.* **P. Prithvi, Dr. T. Kishore Kumar.** 5, 2016, International Journal of Scientific Engineering and Research, Vol. 4, pp. 4-7.

12. *Speech Processing, Transmission and Quality Aspects (STQ); Distributed speech recognition; Front-end feature extraction algorithm; Compression algorithms.* **ETSI - European Telecommunications Standards Institute.** 2003, Technical standard ES 201 108, v1.1.3.

13. *Ségmentation Automatique du Signal de Parole Française.* **K. Heraoua, A, Hirache.** 2015, Mémoire de Master.

14. *Speech Reconstruction from Mel-frequency Cepstral Coefficients via ℓ1-norm Minimization.* **Gang Min, Xiongwei Zhang, Jibin Yang and Xia Zou.** Xiamen : s.n., 2015. IEEE 17th International Workshop on Multimedia Signal Processing (MMSP).

15. *Low Bit-Rate Speech Coding Through Quantization of Mel-Frequency Cepstral Coefficients.* **Boucheron, Laura E, De Leon, Philip L and Sandoval, Steven.** 2, Feb 2012, IEEE TRANSACTIONS ON AUDIO, SPEECH, AND LANGUAGE PROCESSING, Vol. 20, pp. 610-619.

16. *Comparison of MFCC, LPCC and PLP Features for the Determination of a Speaker's Gender.* **Ergün Yücesoy, Vasif V. Nabiyev.** Trabzon, Turkey : s.n., 2014. IEEE 22nd Signal Processing and Communications Applications Conference.

17. *Toward a rich Arabic Speech Parallel Corpus for Algerian Sub-Dialects.* **Bougrine, Soumia, et al.** Portoro, Slovenia : s.n., 2016. LREC'16, (OSACT2).

18. **audioblocks.** Numbers Spelling Male Deep Voice - English Sound effect. [Online] audioblocks. [Cited: 05 20, 2017.] https://www.audioblocks.com/stock-audio/numbers-spelling-male-deep-voice---english.html.

19. **sapp.org.** WAVE PCM soundfile format. *soundfile++: A Soundfile Reading/Writing Library in C++.* [Online] [Cited: Avril 17, 2017.] http://soundfile.sapp.org/doc/WaveFormat/.