Plan :

Introduction :

La biométrie est la reconnaissance de caractéristiques propres en vue d'une identification. Parmi les plus utilisés, on peut trouver la reconnaissance d'iris, l'analyse comportementale, les empreintes digitales, et la voix. La biométrie se divise en deux catégories d'analyses :
-la **vérification** : on veut savoir si celui qui parle est bien celui qu'il dit être.
-l'**identification** : on veut identifier une voix parmi une base de données préenregistrées → c'est notre objectif.

Comment, à partir d'un signal vocal humain, peut-on en déterminer les caractéristiques et les exploiter informatiquement en vue d'une identification ?

I) La voix en biométrie

La voix est un signal sonore. Il est produit par la contraction du diaphragme, envoyant de l'air dans les cordes vocales. Leurs vibrations forment un signal vocal. Celui ci est ensuite modifié par le pharynx, puis par les cavités nasales et buccales. [A]

Les informations extractibles du signal vocal sont de plusieurs types :
-les traits de haut niveau : le lexique utilisé, la prononciation, caractéristique de l'origine sociale, de l'éducation, de la langue maternelle, de la personnalité…
-les traits prosodiques et temporels : l'énergie, la durée, le rythme, le pitch (variation du fondamental)
-les traits spectraux à court terme : le spectre du signal au cours du temps.
→ (voir [C])

II) Principes théoriques de l'identification

On met en entrée le signal sonore sur lequel la voix est enregistrée.

On se base sur un modèle de production de la voix : on prend des hypothèses simplificatrices par rapport au modèle réel permettant de caractériser le locuteur, et ce modèle sert de base à la paramétrisation.

D'abord viens la **paramétrisation** : on transforme un signal initial lourd et non traité en un vecteur sonore léger et caractéristique du locuteur.
Ensuite, on a la **comparaison :** on cherche dans la base de donnée le vecteur sonore du locuteur le plus proche du vecteur sonore à tester, et on renvoie ce résultat.
→ Voir [B]

III) Mise en pratique

1e idée de modèle :

- **Signal d'entrée :** le locuteur dit un « a », durant la totalité du signal (2-3 secondes)

- **Comparaison** : Voir [D]
C'est la méthode la plus naturelle de comparaison.

- **Paramétrisation** : on analyse l'amplitude des harmoniques :

On repère quelle est la fréquence fondamentale f_0 du signal sur son spectre en amplitude (obtenu grâce au module de la FFT). Ensuite, on regarde le rapport entre l'amplitude du fondamental et celle des harmoniques. ([

Le vecteur sonore est donc : **[S(2f₀) / S(f₀), … S(kf₀) / S(f₀)]**

```
def paramHarm(temporel, fe, dt = 0):
    """Renvoie le rapport des 10 premières amplitudes des harmoniques avec celle de f0"""
    fourier = fft(temporel)
    amp_fond, fond = freq_amp_fond(fourier)
    return [amp_harm(fond, k, fourier) for k in range(2, 7)]
```

- **Modèle** : ici on suppose que le spectre ne change pas de forme selon la hauteur du son, et selon l'amplitude totale.

- **Résultats** : ce choix de modèle et de comparaison est plutôt mauvais :
Pour une base de 2 locuteurs, on a 100 % d'identification, mais pour une base de 5 voix, on tombe à 60 %, et ça tombe en dessous au-delà. De plus, en conditions réelles, la hauteur de la voix change d'un enregistrement à l'autre si ils sont espacés de plusieurs jours (comme en situation réelle), de même que la forme du spectre, qui dépend en réalité plutôt du son émis et des conditions d'enregistrement.

2e idée de modèle :

On garde la même **comparaison**, et le même **signal d'entrée**.

- **Modèle** :

On considère que les cordes vocales émettent un signal d'entrée e(t).
On modélise le passage par le pharynx et par les cavités nasales et buccales par un filtre.
Ainsi le signal qu'on enregistre est :

$$\mathbf{s(t) = h * e(t)}$$

On passe en complexe, et on obtient le spectre en amplitude en passant au module une fft.

$$\mathbf{|\underline{S}(f)| = |\underline{H}(f)| \times |\underline{E}(f)|}$$

On passe au logarithme, pour séparer l'influence du filtre et de l'entrée, puis on effectue une DCT pour repasser en temporel.

$$\mathbf{s'(}\textit{ce t}\mathbf{) = h'(}\textit{ce t}\mathbf{) + e'(}\textit{ce t}\mathbf{)}$$

ce t est homogène à un temps. En fait, on effectue un changement d'échelle temporel : on est dans l'espace *quéfrentiel*, et on fait de l'*alanyse cepstrale*.

- Paramétrisation :
On applique donc la formule, et on sélectionne les 50 premiers coefficients.

```
def paramCep(temporel, fe, dt):
        """ Calcule des coefficients cepstraux :
        calcule la dct du log de la fft en amplitude du signal"""
        return fftpack.dct(np.log(fft(temporel)))[0:50]
```

- Résultats :
On obtient de bons résultats : jusqu'à 9 (notre nombre de locuteurs), on arrive à 100 % de réussite.

Pour pousser plus loin l'analyse des résultats, on affiche une matrice de pixels, avec un pixel en (i,j) correspondant à la comparaison entre $c_{i\ source}$ et $cj_{\ test}$ (avec des pixels plus foncé quand on est plus proche.
On observe donc bien une diagonale plutôt foncée, ce qui nous indique les 100 % annoncés. Cependant, on remarque en observant les comparaisons entre les locuteurs que assez souvent, la différence entre la comparaison de $c_{i\ source}$ avec $c_{i\ test}$ et avec $c_{j\ test}$ peut être très faible, comme on le voit aux lignes 2, 6, 8.

La 2e idée présente donc 2 défauts :
-D'abord elle n'est pas adaptée à un grand nombre de voix, puisque plus on ajoute de voix dans la base de données, plus on ajoute d'écarts faibles, voire d'erreurs d'identification.
-Ensuite, on l'applique à un signal « a » qui n'est pas adaptée à la biométrie : cette technique d'identification est utilisée en réalité sur des textes, ou des mots très courts, et pas des « a » de plusieurs secondes.

Ainsi, on doit adapter notre idée à un signal d'entrée que l'on utilise en pratique, dans lequel le spectre, et à fortiori le cepstre, évolue au cours du temps.

3e idée de modèle :

- Signal d'entrée : le signal est un « a » court (de l'ordre du dixième de secondes) entouré de silence.

- Paramétrisation :
On doit prendre en compte le fait que tout le signal n'est pas à sélectionner, mais seulement la partie où la voix se fait entendre.

On commence donc d'abord par découper le signal en trames de temps fixe : la littérature nous dit que le signal ne varie pas trop en 20ms, on va donc prendre une longueur de trames de $n = 1024$ échantillons, (environ 23ms). Ensuite, on calcule l'énergie de chaque trame (la somme des carrés des amplitudes). On sélectionne quelles sont les trames d'énergie suffisante (supérieure à l'énergie totale sur 50).
On calcule les coefficients cepstraux comme en 2e idée sur chaque trame, et on les moyennes.

- Comparaison :
Après test, on se rend compte que la comparaison directe des premières idées est insuffisante. On va pousser plus loin la comparaison à l'aide d'un réseau de neurones.
On utilise un modèle de neurone biologique simplifié , le perceptron.

Il reçoit en entrée un vecteur $E = (e_i)$, et calcule la quantité $S = f(\sum e_i w_i)$, avec f la fonction de seuil et $W = (w_i)$ le vecteur poids.

On va créer un perceptron par locuteur. Chacun va s'entraîner à reconnaître un vecteur sonore parmi une liste de vecteurs (1 par locuteurs), ie à renvoyer 1 si on a le bon vecteur sonore et 0 si ce n'est pas le bon. On va donc mettre à jour chaque poids par la règle :

$$\mathbf{w_{i+1} = w_i + (s - S) \times x_i \times pas}$$

On commence donc par générer aléatoirement (w_i), puis on itère cette mise à jour jusqu'à que l'erreur s - S soit majorée par ε.

- **Résultats** : Parlons d'abord de la convergence. Elle est **plutôt rapide** : environ 10s pour 16 voix, une précision ε de 0.01 avec un ordinateur personnel. Cependant, l'identification est moins précise. Sur les 16 voix, **seules 12 sont reconnues**, et le réseau n'est sûr de lui à plus de 50 % **que pour 7 d'entre elles**. De plus, pour ε = 0.001, la convergence est beaucoup plus lente (de l'ordre de la minute, ce qui reste acceptable), et cela n'améliore pas le taux d'identification. (voir [K])
En fait, ici ce n'est pas vraiment une erreur de paramétrisation : en effet, en utilisant des MFCC (Mel Frequency Cepstral coefficients), une version améliorée des coefficients cepstraux utilisée couramment, on retrouve les même résultats.C'est donc le réseau qui pose problème.

Conclusion :

Ainsi, l'identification du locuteur n'est pas un problème simple. L'exemple de notre première idée nous le montre bien : il ne suffit pas d'une analyse spectrale simple pour savoir qui parle. Il faut modéliser plus en profondeur la voix, en tenant compte de comment elle est produite.
Ensuite, nous avons vu que sur un signal dont les propriétés varient dans le temps, nous sommes obligé d'utiliser une méthode de comparaison plus avancée. Cependant, les réseaux de neurones monocouches que nous avons implémentés sont insuffisants à identifier le locuteur dans une base de données.

La méthode utilisée couramment pour l'identification du locuteur par réseaux de neurones est constituée d'un réseau de neurones multicouches et de MFCC. Sans réseaux, on peut aussi dire que ce qui est caractéristique d'un locuteur n'est pas ces coefficients mais leur répartition statistique, souvent représenté par un modèle GMM (mixture de gaussiennes). Alors, on atteint sur des très grosses bases de données de très bons résultats.

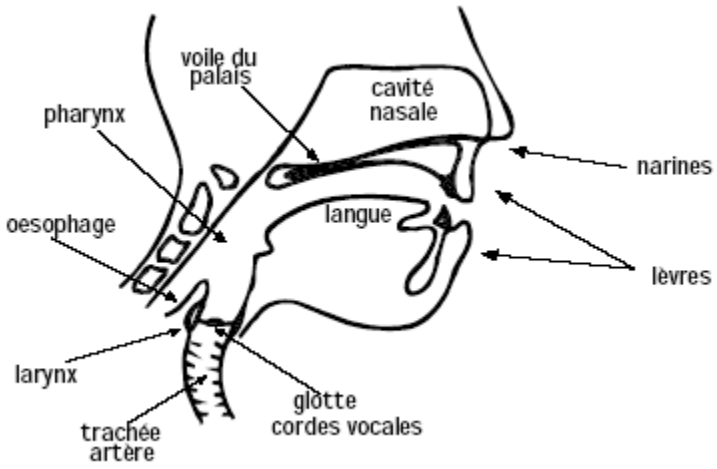

Figure A

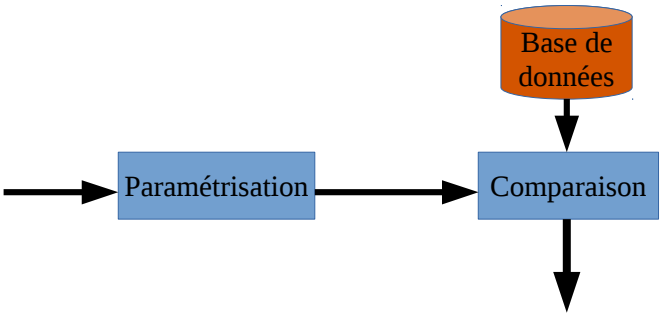

Figure B

Bas niveau (physiologique) → Haut niveau (comportemental)

De + en + difficile à extraire
De - en - affectés par les conditions d'enregistrement.

C'est ici que l'on se place.

Figure C

Vecteur sonore v

Base de données (u_i)

On choisis i tel que $\|v-u_i\|_2$ **minimal**

Figure D

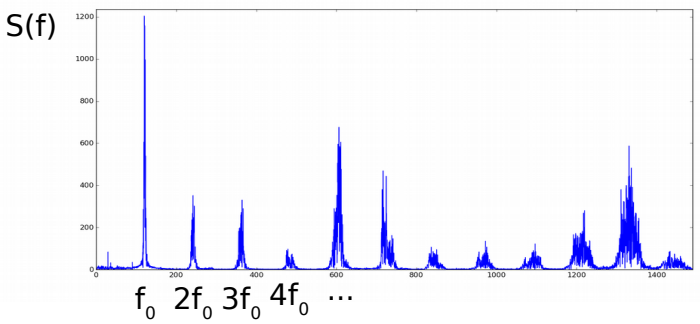

Figure E

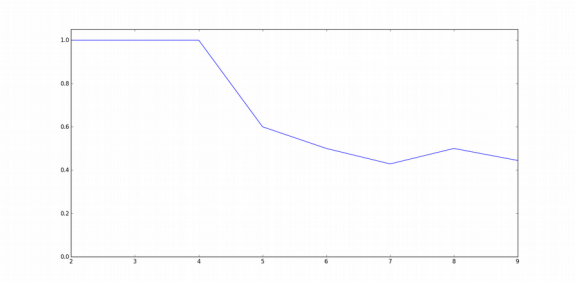

Figure F

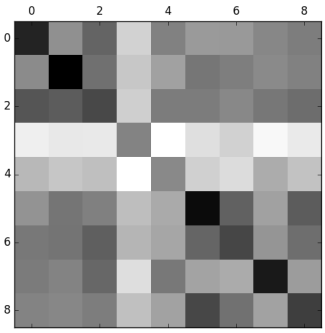

Figure G

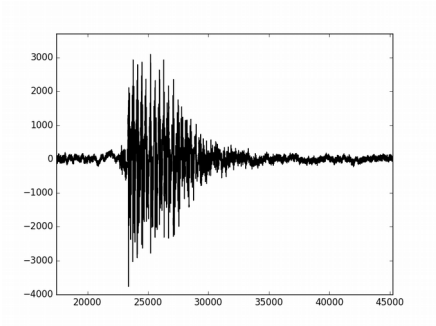

Figure H

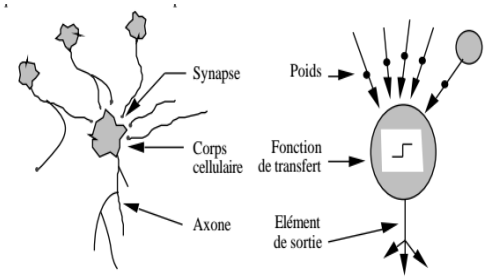

Figure I

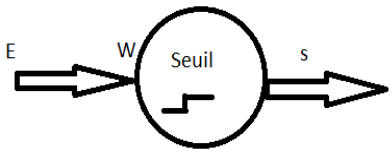

Figure J

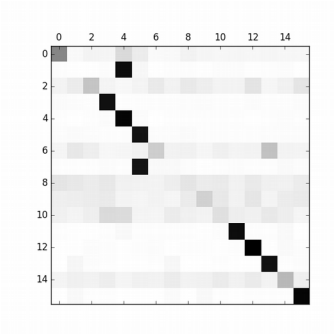

Figure K