ENIB S4 :
UE B - Espace euclidien

Projet DTMF : Reconnaissance d'un code DTMF au sein d'un fichier audio

DE MORAIS Theo
LE TALLEC NEGRIT Mael
PASCO Florian
WEBER SEGARRA Yann

Version 1.5 : 02 Novembre 2022

Table des matières

Table des figures

1 Le problème à résoudre

1.1 Le problème dans sa globalité

Partant d'enregistrements sonores de code DTMF (expliqué ci-dessous), il sera question de distinguer le numéro de la touche appuyé en fonction de l'enregistrement.

1.2 Qu'est-ce que le code DTMF ?

Un code DTMF est une combinaison de fréquences utilisée pour la téléphonie classique. Les fréquences utilisées sont spécifiées par le tableau ci-dessous. Ces codes sont émis lors de la pression sur une touche du clavier téléphonique et amènent à la production d'un son d'instinct pour chaque touche.

	1 209 Hz	1 336 Hz	1 477 Hz
697 Hz	1	2	3
770 Hz	4	5	6
852 Hz	7	8	9
941 Hz	*	0	#

FIGURE 1 – Le tableau des fréquences associés à chaque bouton

1.3 Qu'est-ce que l'enregistrement sonore ?

On nous fournira un enregistrement sonore qui correspondra à l'évolution de l'amplitude du signal sonore au fil du temps.

2 Modélisation mathématiques et justification de celle-ci

2.1 Espace vectoriel

Pour modéliser le problème mathématique, il sera question de se placer dans l'ensemble des fonctions continues de [0,1] . Ici on travaillera sur une famille de vecteur composée de fonctions de type sinus avec des fréquences de code DTMF (697, 770, 852, 941, 1209, 1336 et 1477).

Les fonctions, seront de cette forme :

$$u1 : [0,1] \to \mathbb{R}$$

avec f les fréquences de nos signaux DTMF.

$$t \mapsto \sin(f.2.\Pi.t)$$

2.2 Combinaison linéaire

Dans un premier temps nous devrons trouver chaque coefficient de la combinaison linéaire.
Afin de trouver cette combinaison linéaire nous pouvons faire le produit scalaire de l'enregistrement et des 6 fonctions de notre famille afin de trouver chaque coefficient de la combinaison linéaire.

La formule suivante nous permettra de réaliser le produit scalaire :

$$\int_0^1 u1(t)u2(t)\,\mathrm{d}t$$

on noteras la formule ci-dessus, pour produit scalaire entre u1 et u2 <u1,u2>

2.3 Démonstration du produit scalaire

On sait que dans R2 le produit scalaire de deux vecteurs est définit par :

$$\begin{pmatrix} a1 \\ a2 \end{pmatrix} \cdot \begin{pmatrix} b1 \\ b2 \end{pmatrix} = a1.a2 + b1.b2$$

Si nous généralisons pour les vecteurs de notre famille on obtient la somme de Riemann :

$$= \frac{b-a}{nbEchantillon} \times \sum_{k=1}^{nbEchantillon} f(a + \frac{k(b-a)}{nbEchantillon})$$

Avec f(x)=u1(x)*u2(x)

Si f est intégrable sur [a,b] alors les sommes de Riemann converge vers l'intégrale de f sur [a,b]

$$= \int_a^b f(x)dx$$

Dans le cas ou a = 0 et b = 1

$$= \int_0^1 f(x)dx$$

2.4 Norme

Nous remarquons que lorsque nous faisons le produit scalaire entre 2 mêmes vecteurs nous obtenons :
<u1,u1> = 1/2
Cela veut dire que la norme de nos vecteurs est de $1/\sqrt{2}$ et non pas de 1, ils ne sont donc pas unitaire. Notre famille n'est donc pas normée mais elle est bien orthogonale.
Pour palier à ce problème, il nous faudras donc multiplier nos fonctions/vecteurs par $\sqrt{2}$ pour avoir les bons coefficients correspondant au composantes dans les enregistrements DTMF. Cela revient à multiplier notre produit scalaire par 2.
Pour reprendre l'exemple au dessus, si nous voulons connaître le "nombre de u1 dans u1", il faut faire :
<$\sqrt{2}$u1,$\sqrt{2}$u1> = 1
2<u1,u1> = 1
Il y a bien 1 fois u1 dans u1. Ainsi pour un signal de forme f1, il nous suffiras de faire le produit scalaire entre celui-ci et

chaque vecteur de notre famille et de multiplier nos vecteurs par $\sqrt{2}$:

$<\sqrt{2}f1,\sqrt{2}u1> = a$
$<\sqrt{2}f1,\sqrt{2}u2> = b$
$<\sqrt{2}f1,\sqrt{2}u3> = c$
$<\sqrt{2}f1,\sqrt{2}u4> = d$
$<\sqrt{2}f1,\sqrt{2}u5> = e$
$<\sqrt{2}f1,\sqrt{2}u6> = f$
$<\sqrt{2}f1,\sqrt{2}u7> = g$

En fonction de la combinaison linéaire obtenue nous saurons de quels signaux DTMF un signal quelconque est composé et nous pourrons déduire le numéro de la touche appuyée.

2.5 Les vecteurs de notre famille sont orthogonaux entre eux

Ici nous pouvons appliquer notre définition du produit scalaire à chacun de nos vecteurs. En faisant cela nous constatons que tout les produits scalaires sont nuls. En effet, nos fonctions sinus ont des arguments de la forme "2kπ", lorsque l'on calcule notre produit scalaire cela va nous amener au final à avoir une somme de sin(2kπ) étant tous égal à 0.

2.6 La famille composé de fonction sinusoïdale (construites avec les fréquences DTMF) est une famille libre

Famille libre : Famille de vecteurs dont la seule combinaison linéaire de ses vecteurs qui soit égale au vecteur nul est celle dont tous les coefficients sont nuls.

Prenons nos fonctions de la famille DTMF u1, u2, u3, u4, u5, u6, u7 et faisons le produit scalaire tel que :
<u1, a0u0 + a1u1 + a2u2 + a3u3 + a4u4 + a5u5 + a6u6> = 0
(Nous supposons qu'il est nul d'après la propriété énoncée plus haut).

<u1,a0u0> + <u1,a1u1> + <u1,a2u2> + <u1,a3u3> + <u1,a4u4> + <u1,a5u5> + <u1,a6u6> = 0
a0<u1,u0> + a1<u1,u1> + a2<u1,u2> + a3<u1,u3> + a4<u1,u4> + a5<u1,u5> + a6<u1,u6> = 0
Si on fait les produits scalaires on obtient :
$a0 \times 0 + a1 \times 1 + a2 \times 0 + a3 \times 0 + a4 \times 0 + a5 \times 0 + a6 \times 0 = 0$
On a donc a1 = 0 (ce qui est ici vrai pour a1 les également pour les autres coefficients)
Ainsi la seule combinaison linéaire de ses vecteurs qui soit égale au vecteur nul est celle dont tous les coefficients sont nuls et notre famille est libre.

2.7 Signal avec déphasage

Un signal DTMF que que nous voulons décrypter comporte un déphasage. Pour palier à ce problème nous avons écrit un signal avec un déphasage ϕ de cette façon :

$$\sin(f.2.\Pi.t + \phi) = \sin(f.2.\Pi.t)\cos(\phi) + \cos(f.2.\Pi.t)\sin(\phi)$$

$$= \sin(f.2.\Pi.t)\cos(\phi) + \sin(f.2.\Pi.t + \frac{\pi}{2})\sin(\phi)$$

On peut donc considérer qu'un signal DTMF déphasé est composé de deux signaux DTMF identiques dont l'un est déphasé de $\frac{\pi}{2}$. Ils ont tout les deux un coefficient $\cos\phi$ et $\sin\phi$.

On a donc, pour une pulsation w_i :

$< w_i, \sin(w_i t) >= c_i \times cos\phi$
$< w_i, \sin(w_i t + \frac{\pi}{2}) >= c_i \times \sin\phi$

Nous voulons à présent retirer les coefficients $\cos\phi$ et $\sin\phi$, pour cela on peut ajouter les 2 produits scalaire et de les mettre au carré :
$< w_i, \sin(w_i t) >^2 + < w_i, \sin(w_i t + \frac{\pi}{2}) >^2 = c_i^2\cos(\phi)^2 + c_i^2\sin(\phi)^2 = c_i^2$

Le coefficient que nous obtenons est au carré, mais ce n'est pas gênant pour notre application, en effet, nous voulons uniquement trouver les coefficients les plus grands pour reconnaître la touche appuyée. Ainsi, pour décoder n'importe quel signal DTMF déphasé, il nous suffit d'agrandir notre famille de vecteurs en y ajoutant les mêmes vecteurs mais déphasés de $\frac{\pi}{2}$.

3 La stratégie pour résoudre le problème

3.1 Son classique

3.1.1 Crée la base orthonormée

La famille de vecteurs(ici nos fonctions sinus sont des vecteurs appartenant à l'ensemble des fonctions continues de [0,1]) permettant de créer des signaux DTMF est libre(comme démontrée précédemment). Nous sommes ici contraints de discrétiser chacune de ces fonctions pour les analyser puisque celle-ci sont de taille infinie.

```
def function_u(i,t):
    return np.sin(t*w[(i)%7]*2*np.pi+(i//7)*np.pi/2)

def createOrthonormalBasis(sample_rate,signal_freq):
    u = []
    t_ech=np.linspace(0.0,1.0,fs) # intervalle d'echantillonnage
    for i in range(14):
        u.append(np.array(function_u(i,t_ech)))
    return u
```

3.1.2 Récupérer le code DTMF que nous allons analyser

Ensuite, nous récupérons l'émission du code DTMF qui a été enregistrée afin de l'analyser

```
def getCodeFromFile(filename):
    file = open(filename, "rb")
    code_recover=np.load(file)
    file.close()
    return code_recover
```

3.1.3 Retrouver la combinaison linéaire l'ayant créer

Puis, nous allons retrouver la combinaison linéaire à l'origine du son émis. Pour cela, on va faire une projection orthogonale de notre son sur la base orthonormée crée précédemment.

```
def scalarProduct(u,v):
    return 2*integrale(np.multiply(u,v))

def coordonnees(u,v):
    c = []
    for i in range(len(u)):
        c.append(scalarProduct(u[i],v))
    return c

def projectionOrt(u,v):
    c = coordonnees(u,v)
    p = np.zeros_like(v)
    for i in range(len(u)):
        p += c[i]*u[i]
    return [c,p]
```

Afin de réaliser cette projection orthogonale, il va être question de réaliser le produit scalaire de notre son avec chaque vecteur de la base.

```
def scal(v,w):
    return 2 * integrale(v,w)
```

```
def integrale(w):
    dt = (1/(len(w)-1))
    sum_w = 0
    for i in range(len(w)-1):
        sum_w += w[i]

    return sum_w*dt
```

3.1.4 Retrouver les 2 fréquences principales

Pour former un code DTMF, on utilise 2 sinusoïdes de fréquences différentes, l'idée est ici de retrouver ces 2 fréquences.

```
def findingMainFrequencies(c):
    c_processing = c.copy()
    m1 = max(c_processing, key=abs)
    i1 = c.index(m1)

    #remove the largest number using its index
    c_processing.pop(i1)

```

```
    m2 = max(c_processing, key=abs)
    i2 = c.index(m2)
    return [i1,i2,m1,m2]
```

3.1.5 Retrouver le chiffre liée

À chaque combinaison de vecteur, est lié un chiffre du téléphone, la question est ici de savoir lequel.

```
def whatNumIsIt(i1, i2):
    smallest = min([i1,i2])
    biggest = max([i1,i2])
    matrice = [[1,2,3],
        [4,5,6],
        [7,8,9],
        ['*',0,'#']]
    if smallest <= 3 and biggest >= 4:
        return matrice[smallest][biggest-4]
    else:
        return "La combinaison linéaire ne permet pas de déterminer une touche"
```

4 Les outils techniques utilisés

- Python (Jupyter Notebook + diverses librairies)
- Espace vectoriel
- Orthogonalité et base orthonormée
- Combinaison linéaire
- Produit scalaire
- Norme
- Projection orthogonale
- Intégrale