Mode d’emploi
IHM EEG

EEG VISION

TÉLÉCOM SUDPARIS

8 JUIN

Projet Cassiopée 2020
Créé par : CALDAS Louis

Sommaire

Table des matières

I. Préambule et choix technologiques

a. Présentation générale

Le but de cette application est de servir **d'interface de visualisation** au niveau des signaux d'électroencéphalographie (EEG). Une partie permettant de visualiser des signaux bruts et une autre partie de l'application permettant de visualiser la synchronisation entre les électrodes.

Pour cela, nous avons décidé de s'orienter vers le langage de programmation cross-platform **Python**, en effet ce dernier nous permettait de réaliser un grand nombre de calculs et est réputé dans le cadre de la visualisation de signaux, ce qui correspondait parfaitement à notre projet. Reste le problème de la bibliothèque nous permettant de réaliser les fenêtres. Pour cela, le choix retenu a été la bibliothèque **PySide2**. Elle semblait offrir plus de possibilités que Tkinter. C'est une librairie basée sur le moteur de développement C++ Qt adapté en python.

En résumé :

- Langage de développement : Python.
- Librairie de création de fenêtre : PySide2.

b. Environnement de fonctionnement

L'environnement nécessaire pour faire fonctionner l'application est le suivant :

- Un environnement python (version 3 minimum).
- La librairie PySide2 pour les fenêtres.

```
pip install matplotlib
```

- Les librairies Matplotlib, Numpy, Scipy, Seaborn pour les calculs mathématiques et les visualisations de graphiques.

```
pip install matplotlib
pip install numpy
pip install scipy
pip install seaborn.
```

- Les librairies xlrd & xlsxwriter

```
pip install xlrd
pip install XlsxWriter
```

c. Lancement de l'application.

Pour lancer l'application, il suffit de lancer le script **main.py** via n'importe quelle console python via la commande :

python main.py OU python3 main.py

d. Format des données

Les différentes données utilisées par l'application se trouve dans le dossier /Datas/ (le dossier Assets comprenant uniquement les sources de l'application).
Les sous dossiers Active/Control/Death contiennent les matrices des signaux EEG en fonction du temps selon les patients et les différents domaines de fréquences d'ondes cérébrales. Un dossier par patient contenant 4 fichiers (1 par type d'onde).
IMPORTANT : Si l'on souhaite rajouter un patient, il faut rajouter la matrice matlab du signal brut (non traité) dans le dossier Raw/{Type du patient}/nomdupatient.mat. Puis il faut relancer l'application qui à son lancement se chargera de calculer les signaux des ondes cérébrales associées à cette matrice.
Cette matrice doit avoir un format particulier expliqué en annexe.

Le dossier PSC_Matrix contient les matrices de synchronisations de chaque patients, ces dernières sont à ajoutées manuellement par l'utilisateur après calcul sur matlab. Elles sont nécessaires à l'affichage des heatmaps de synchronisations ou de la fenêtre de visualisation de synchronisations.
Leur format est **simple :** mat['PSC'] doit contenir une matrice carré de dimension correspondante aux nombres des électrodes (20x20, 30x30…).
L'ordre des électrodes dans la matrice doit être le même que l'ordre des électrodes dans le fichier /Datas/electrodes_order.txt, **ATTENTION** la modification et l'adaptation de ce dernier aux données étudiées est essentielles pour le bon fonctionnement de l'application.
Il y a également un exemple du formalisme de ce fichier dans le cas de l'utilisation de 20 ou 30 électrodes.

II. Mode d'emploi de l'application

a. Fenêtre de lancement

Figure 1 : Fenêtre de lancement

La fenêtre de lancement est simple, elle propose d'entrer dans les deux modes de l'application : la visualisation des signaux en fonction du temps (bouton visualisation de signal à gauche n°1) ou la visualisation de synchronisation entre les électrodes (bouton visualisation de synchronisation à droite n°2).

b. Fenêtre de visualisation de signal

Liste déroulante permettant de sélectionner le patient dont on veut observer les signaux.

Liste déroulante permettant de sélectionner l'électrode que l'on veut observer. La liste provient du fichier electrodes_order.txt dans le dossier /Datas/

Liste pour la catégorie du patient, Active/Death/Control.

Liste pour le type d'onde à observer Alpha/Beta/Theta/Delta

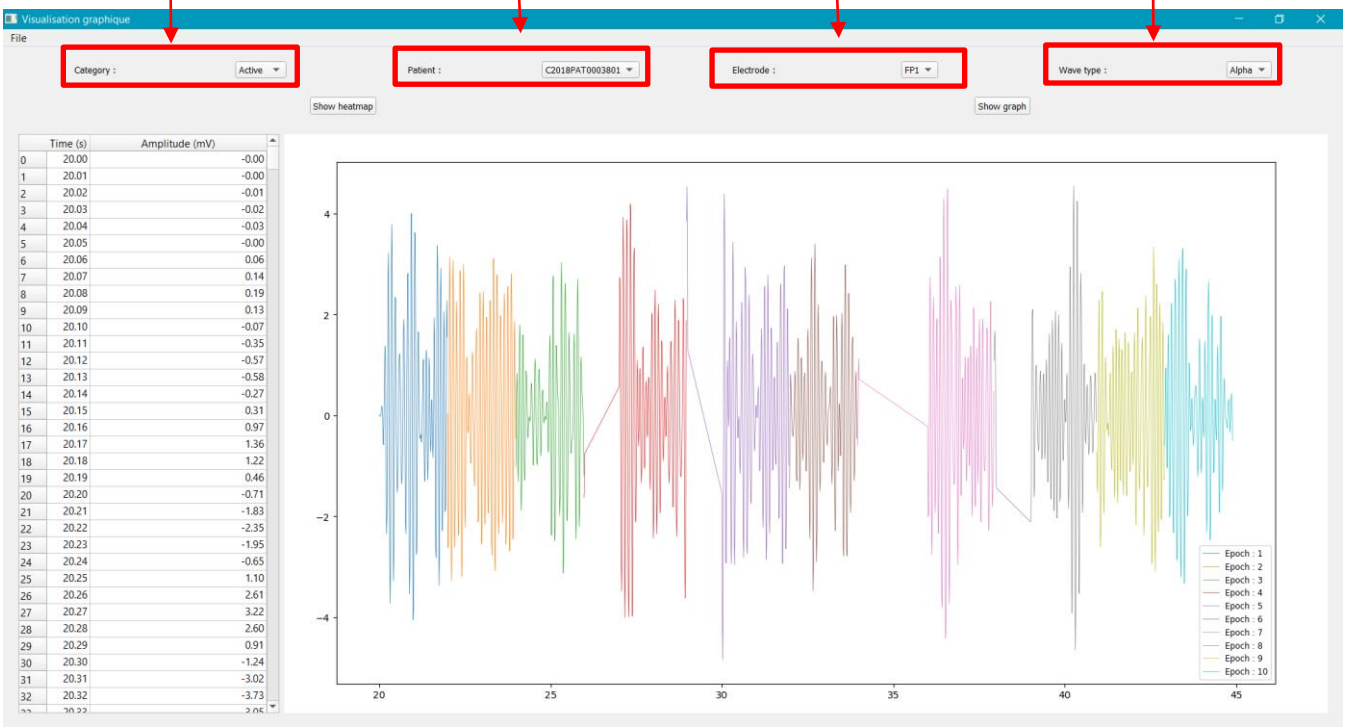

Figure 2 : Fenêtre de visualisation de graph

Comme le précise son intitulé le bouton « Show graph » permet de mettre à jour le graphique de la fenêtre. « Show heatmap » montre la matrice de synchronisation du patient SI ET SEULEMENT SI la matrice de synchronisation de ce patient et de ce type d'onde a été au préalable ajouté dans le dossier : Datas/PSC Matrix/{Type du patient}/nomdupatient.mat

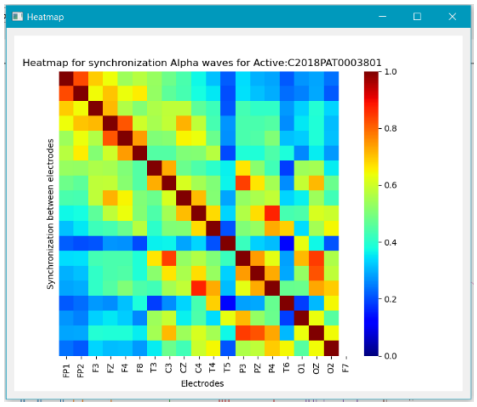

Figure 3 : Heatmap de matrices de synchronisation

Via le menu contextuel, File->Options, il est possible de mettre à jour les bornes des domaines de fréquences pour les ondes alpha, bêta, theta et delta.

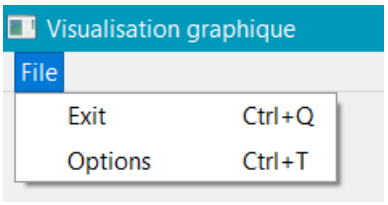

Figure 4 : Menu contextuel fenêtre de visualisation de graphiques

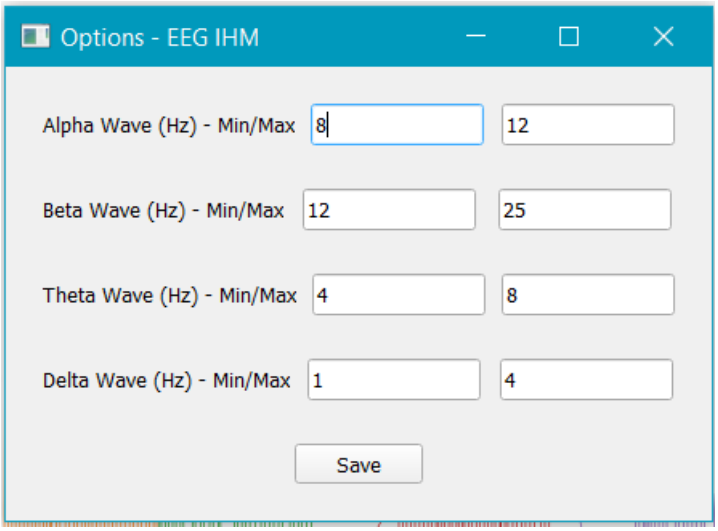

Figure 5 : Fenêtre permettant la personnalisation des domaines de fréquences des ondes cérébrales

c. *Fenêtre de visualisation de la synchronisation entre les électrodes*

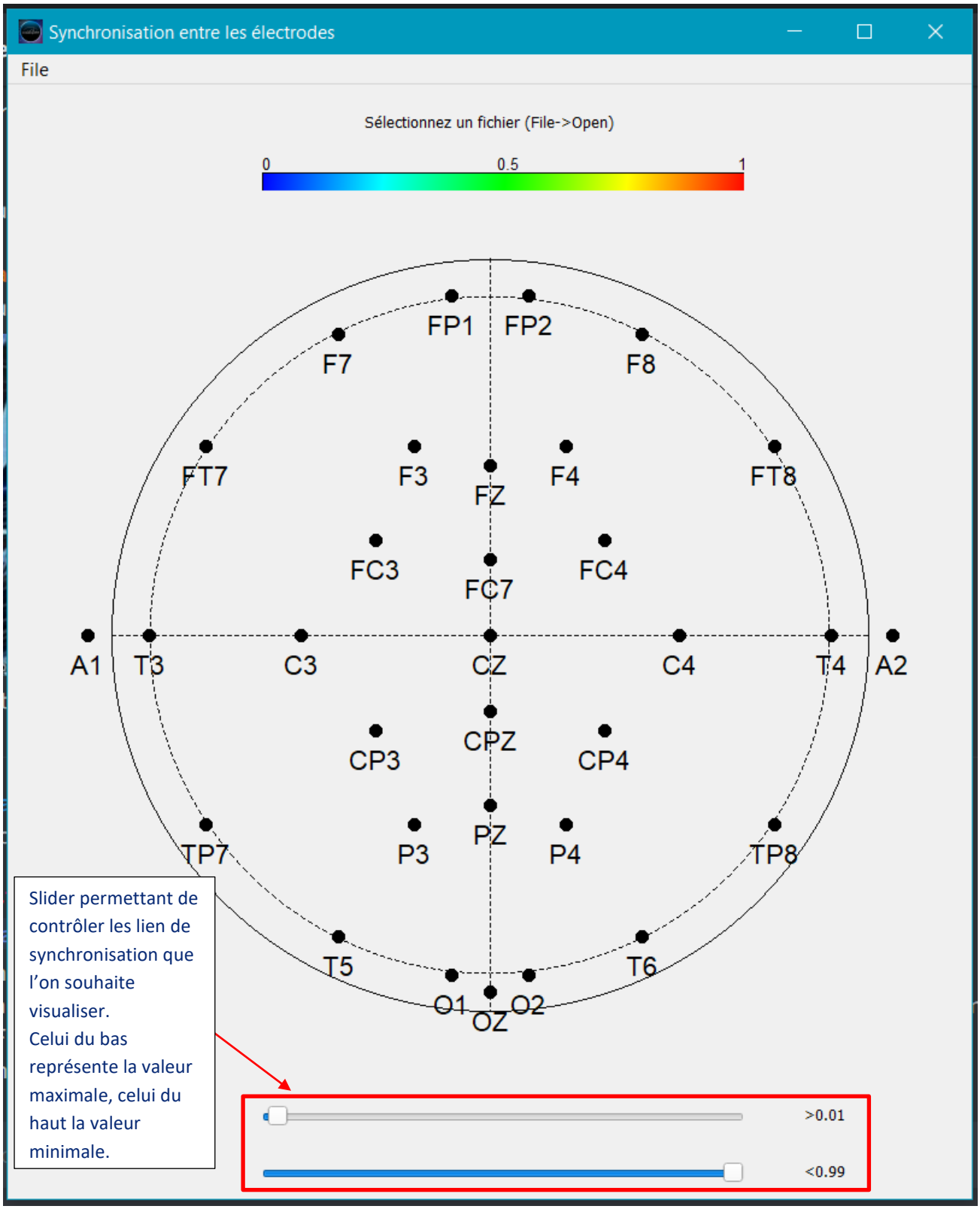

Figure 6 : Fenêtre de visualisation de la synchronisation entre les électrodes

Pour visualiser des synchronisations, il faut sélectionner une matrice de synchronisations via le menu contextuel File->Open.

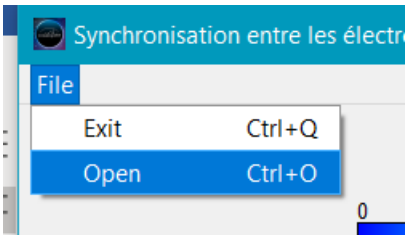

Figure 7 : Menu contextuel fenêtre de visualisation de la synchronisation entre les électrodes

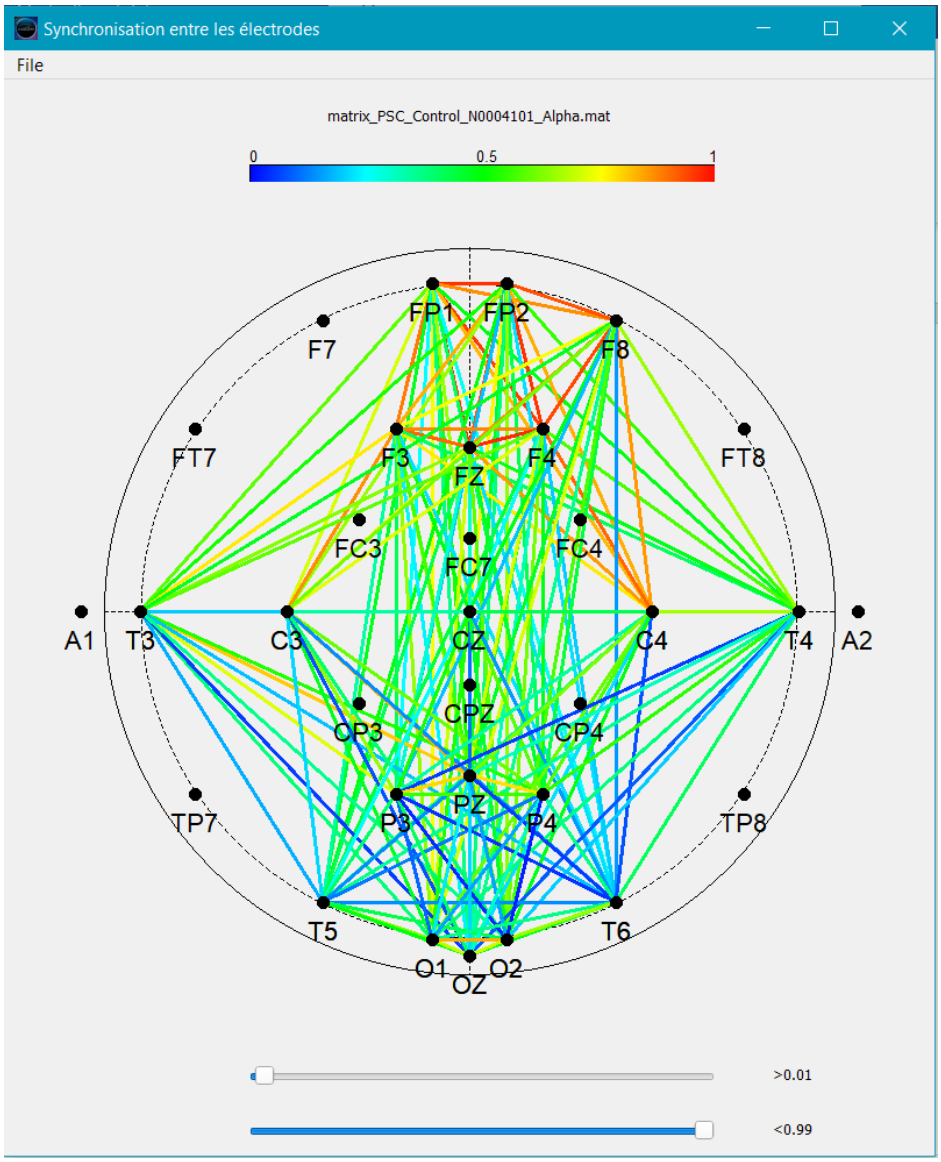

Figure 8 : Fenêtre de visualisation après sélection d'une matrice.

Les synchronisations sont différenciées par différentes couleurs (rouge = forte synchronisation, bleu = faible synchronisation) et des traits de différentes épaisseurs.

III. Annexes

a. Format des matrices matlab des signaux bruts

matrice['epochTime'] qui contient n (nombre d'époque) tableau de m points (nombre de points par époque).
matrice['epochRange'][n] qui contient pour un epoque time n : p tableau de m points (p correspondant au nombre d'électrodes=.
(Finalement on trace l'ensemble des epochRange en fonction des epochTime)

En bref : si l'on a un jeu de données continues, il faut un seule liste de valeurs du temps matrice['epochTime'][0] et toutes les valeurs correspondantes dans matrice['epochRange'][0][p] pour les valeurs de la p-ème électrode.

b. Souhait de modification des données d'entrée.

Si pour une raison quelconque, l'utilisateur souhaite modifier le format demandé par les données d'entrée, ce dernier devra uniquement modifier le fichier util.py présent dans le dossier Assets/mathematical_script/util.py. C'est en effet ce fichier qui gère toute l'intégration et le chargement des matrices. En ce qui concerne les matrices matlab des signaux bruts, c'est en particulier la fonction getRawDatas (https://github.com/Tic-Tac-Toc/EEG_HCI/blob/master/Assets/mathematical_scripts/util.py#L167), x représentant le temps et y les valeurs (en mV) du signal.

c. Code source de l'application

Le code source de l'application est disponible en ligne suivant ce lien : https://github.com/Tic-Tac-Toc/EEG_HCI

Pour tout renseignement complémentaire, vous pouvez contacter :
louis.caldas@telecom-sudparis.eu