Rapport de projet de 2ème année d'Ingénieur

Filière 1 : « Informatique des systèmes interactifs pour l'embarqué, la robotique et le virtuel »

Sujet : Acquisition et traitement de signaux sonores pour piloter différents équipements

Présenté par :

Jamila KAMAL et Thomas QUENTEL

Tuteur académique : M. Mamadou KANTÉ

Tuteur externe : M. Jean-Thierry BODIN

Auteure du projet : Mme. Lauriane CHARIÈRE--FIEDLER

Soutenance : **Mars 2023**

Durée du projet : **60h**

Campus des Cézeaux. 1 rue de la Chébarde. TSA 60125. 63178 Aubière CEDEX

Remerciements

Nous tenons à remercier Mme. Lauriane CHARIÈRE--FIEDLER pour l'idée de projet et M. Jean-Thierry BODIN pour son encadrement, aide et bienveillance tout au long de sa réalisation.

Nous remercions M. Frédéric MEIGNAN pour sa collaboration dans le cadre de l'accès à la salle pour utiliser le logiciel LabVIEW, important pour notre projet.

Nous remercions également M. Michel CHEMINAT et M. Romuald AUFRÈRE pour les connaissances en traitement numérique du signal.

Enfin, nous remercions tous ceux qui ont participé de près comme de loin à la préparation de ce rapport et de notre soutenance, nous pensons notamment à Mme. Murielle MOUZAT, M. Romuald AUFRÈRE et M. Mamadou KANTÉ.

Table des matières

Table des figures et illustrations

Résumé

Le but de ce projet est de réaliser un programme d'acquisition et de **traitement numérique de signaux** sonores acquis par des microphones. Ce programme doit informer l'utilisateur des amplitudes moyennes de la **Densité Spectrale de Puissance (DSP)** sur une plage de fréquences audibles (20 Hz à 20 kHz). Il est prévu d'être fonctionnel sous système d'exploitation Windows 10 et 11 et pour tout type de microphones.

À ce jour, nous avons réussi à acquérir et à faire traiter les sons perçus par un unique microphone. La programmation et les tests ont été effectué sous **LabVIEW 17** et sous système d'exploitation Windows 10 et 11**.**

Mots clés : traitement numérique du signal, Densité Spectrale de Puissance (DSP), LabVIEW 17

Abstract

Our project consists in developing a program that would acquire sounds through various microphones and realise **signal processing**. The final product should inform the user of the power averages of the **Power Spectral Density (PSD)** in given ranges of audible frequencies that is between 20 Hz and 20 kHz. It has to be functional on Windows 10 and 11 and for all types of microphone.

Currently, and with that marking the end of our project, our program functions for only one microphone connected. The development was carried out and tested under **LabVIEW 17** working on Windows 10/11 as Operating System.

Keywords: signal processing, Power Spectral Density (PSD), LabVIEW 17

Lexique

LabVIEW

Vi

Sous-vi

File (ou queue)

Défiler

Décibel (dB)

FFT

Filtre de Tchebytchev (et occasionnellement d'ordre 5)

Densité Spectrale de Puissance (DSP)

Amplitude

Échantillonnage

PowerShell

Installeur

Introduction de l'étude

Dans le cadre de la préparation de son diplôme pour les Beaux-Arts, Mme. Charière--Fiedler, étudiante à l'école ECACM, souhaite réaliser une œuvre d'art qui se basera sur les sons environnants. Son objectif est d'effectuer, par électrolyse sur du cuivre, des gravures dont les profondeurs varieront selon les amplitudes des sons. En effet, c'est en se servant de ces amplitudes que sera réglée l'intensité du courant utile pour réaliser l'électrolyse du métal. Le cuivre ne serait pas recouvert de vernis protecteur permettant ainsi au courant de l'attaquer de différents endroits et créer diverses profondeurs. Ainsi, pour pouvoir maîtriser les intensités requises par les différents équipements, et automatiser la récupération et le traitement des sons environnants, Mme. Charière--Fiedler fit appel à des anciens de l'ISIMA.
C'est donc M. Jean-Thierry Bodin, un ancien étudiant de notre filière (promotion 15 F1) qui nous confia ce projet. C'est aussi vers lui que nous nous tournions régulièrement pour faire le point d'avancement et discuter des choix techniques à prendre.

Pour notre part, notre travail consiste à acquérir les sons et à les traiter numériquement. Avec l'aide de M. Bodin, nous avons choisi de travailler sur le logiciel LabVIEW ; ce logiciel étant assez complet en composants liés au traitement de signaux.
Nous nous sommes convenu que nous fournirons à M. Bodin un programme exécutable qui sera indépendant du logiciel. Il aura entre autres une file comportant différentes amplitudes de sons moyennées sur une plage de fréquences de 20 Hz à 20 kHz.
Pour réaliser notre travail, nous avons, en premier lieu, dû acquérir le son d'un microphone et réaliser son traitement, ce qui inclut le filtrage du bruit, puis le calcul et l'affichage de la DSP. Ensuite, nous avons travaillé les données de la DSP pour déduire différentes amplitudes moyennes dans la plage de fréquences audibles. Nous avons ensuite réalisé l'interface utilisateur pour rendre notre programme facile à utiliser. Enfin, nous avons généré un installeur permettant au programme d'être fonctionnel sous tout ordinateur en Windows 10/11. Deux ordinateurs de l'ISIMA ont été mis à notre disposition pour travailler avec la licence du logiciel.

L'étudiante envisage deux options pour l'acquisition du son : recueillir le signal vibratoire ou directement le son audible. Ces sons divergent par le type de microphones à être employé. N'ayant pas encore tout le matériel à sa disposition, nous nous sommes servi des microphones des ordinateurs portables connectés à distance sur les machines de l'ISIMA ou encore des microphones prêtés par M. Bodin.
M. Bodin poursuivra nos travaux afin qu'ils puissent être le mieux adaptés aux besoins de Mme. Charière—Fiedler.
Dans ce rapport, nous discutons de nos choix d'implémentation et des détails techniques de notre programme dont nous sommes fiers d'appeler *LJTQK_LabVIEW*.

I. Matériel et méthode

I. 1. Organisation

Au début du projet, nous nous sommes convenu sur les attentes avec M. Bodin. Nous avons également fixé l'outil de programmation, ainsi que les réunions d'avancement et des moyens de contact.

Pour nous organiser dans notre travail de groupe, nous sommes partis du diagramme de Gantt de la figure 1. Mais, c'est finalement le planning réel de la figure 2 qui a été tenu.

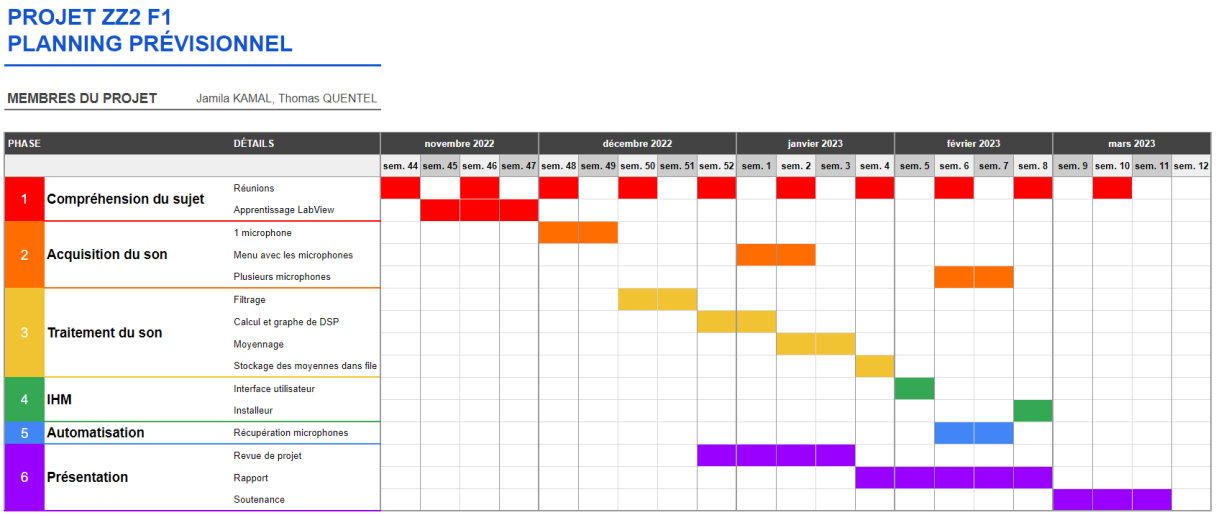

Figure 1 : Gantt prévisionnel

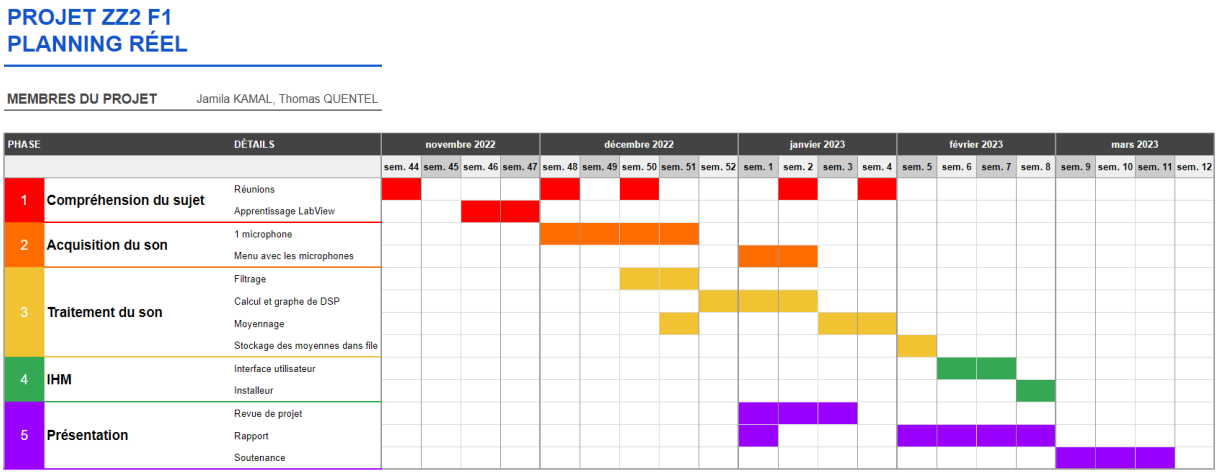

Figure 2 : Gantt réel

Les réunions régulières avec notre tuteur nous ont permis de pratiquer une méthode agile, à savoir : à chaque réunion, nous fixions une tâche à effectuer avant la prochaine réunion, ce qui permettait de vérifier avec notre tuteur si nous réalisions bien ses attendus sur le projet.

Pour effectuer un suivi de ce qui était à faire, ce qui était en cours de réalisation et ce que nous avons déjà fait, nous complétions régulièrement le Kanban de la figure 3 réalisé sur le site Trello. Cette technique nous permettait également de savoir qui s'occupait de quelle tâche.

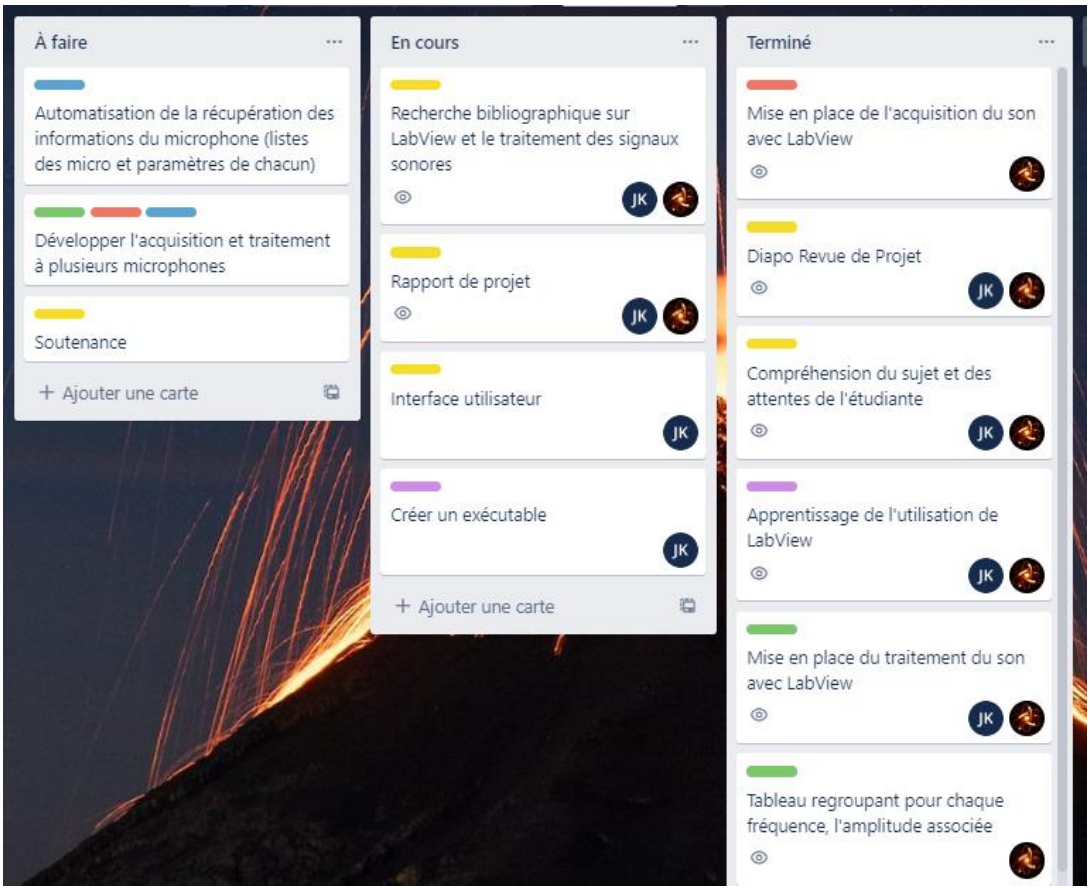

Figure 3 : Kaban d'organisation

I. 2. Implémentation

Le programme se découpe en plusieurs sous-programmes indépendants, appelés les « sous-vi ». La structure de notre programme est définie de façon suivante :

- *main.vi*
  - *recuperation_microphones.vi*
  - *acquisition_son.vi*
  - *traitement_son.vi*
    - *moyennage.vi*
  - *variables_globales.vi*

Dans la figure 4, nous avons le *main.vi*, notre programme d'entrée.

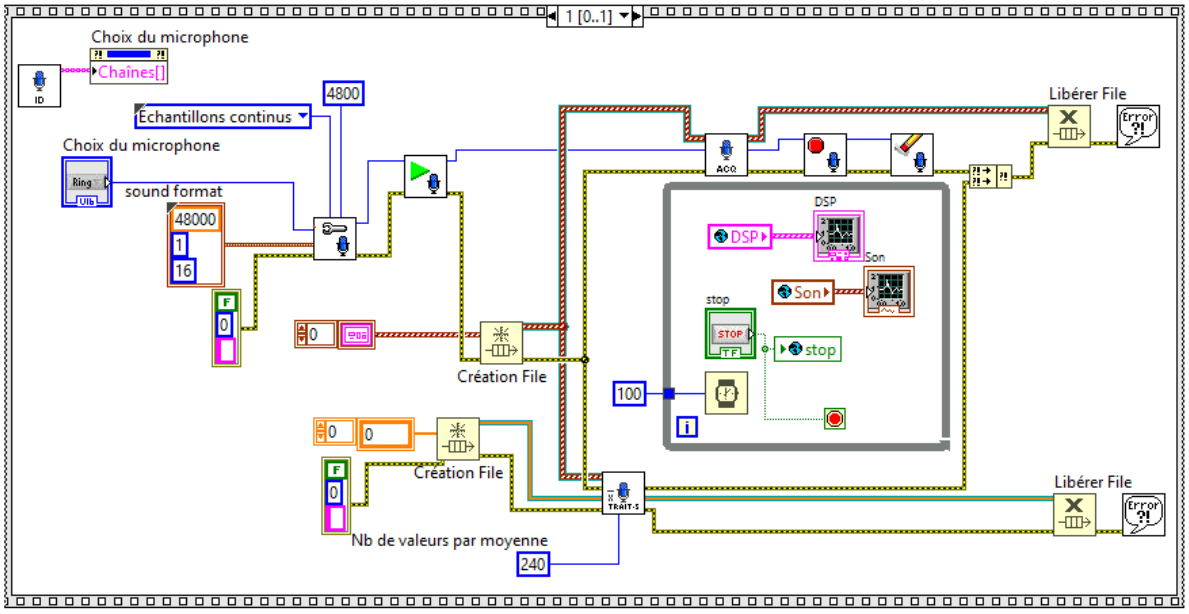

Figure 4 : Allure du programme principal

Sur la figure 4, à gauche, nous avons le paramétrage du microphone, en haut à droite, le bloc « ACQ » qui effectue l'acquisition du son et en bas, le bloc « TRAIT-X » pour le traitement du son. Pour passer de l'acquisition au traitement du son, nous utilisons une file qui récupère chaque acquisition avant de les traiter. Nous avons également une file qui récupère les valeurs attendues du projet après traitement.

I. 2. 1. Acquisition

Pour réaliser l'acquisition du son d'un microphone, nous avons, dans un premier temps, paramétré l'enregistrement du son comme le montre la figure 5, ce qui implique de respecter certains principes d'échantillonnage.

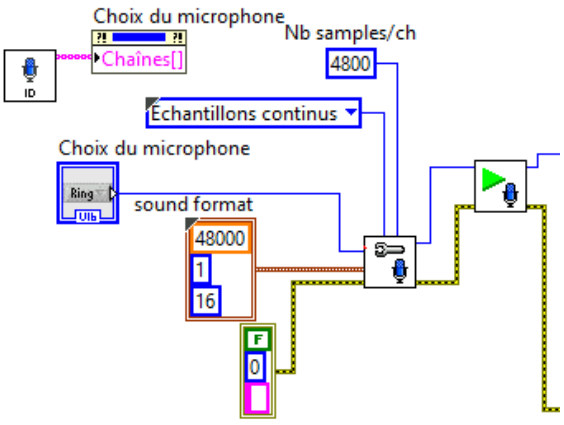

Figure 5 : Paramétrage de microphone

L' enregistrement doit respecter le théorème de Shannon : Un signal continu X(f) de spectre borné dans $[-Fm, Fm]$ est complétement déterminé, à l'aide de la formule d'interpolation de Shannon, par des valeurs qu'il prend à des instants régulièrement espacés de $T_e = \frac{1}{F_e}$ secondes au plus ; $F_e = 2 \cdot F_m$. T_e est appelée la période d'échantillonnage et $\mathrm{F_e}$ la fréquence d'échantillonnage. Le non-respect du théorème de Shannon, c'est-à-dire, $F_e < 2 \cdot F_m$, engendre un phénomène de recouvrement de spectre (*folding*, *aliasing*).[1]

À côté de l'échantillonnage, pour que les signaux soient utilisables par des calculateurs numériques, il faut réaliser une quantification. C'est une opération de discrétisation des valeurs du signal $y(t_n)$ pour leur faire correspondre des valeurs numériques. [1]

Ainsi, nous avons fait le choix de prendre un enregistrement de qualité DVD, proposé par défaut, et qui respecte ce théorème pour la plage de fréquences audibles allant de $20\ Hz$ à $20\ kHz$. Avec une fréquence d'échantillonnage de $F_e = 48\ kHz$, une résolution de $16\ bits$ et 1 canal. Comme le montre le bloc « sound format » sur la figure 5. La résolution étant le nombre de bits utilisé pour la quantification de l'enregistrement.[2]

Ensuite, nous avons réalisé un menu avec les différents périphériques d'entrées et de sorties audio pour laisser l'utilisateur choisir le microphone de son choix. Pour cela, nous avions besoin de récupérer les microphones.

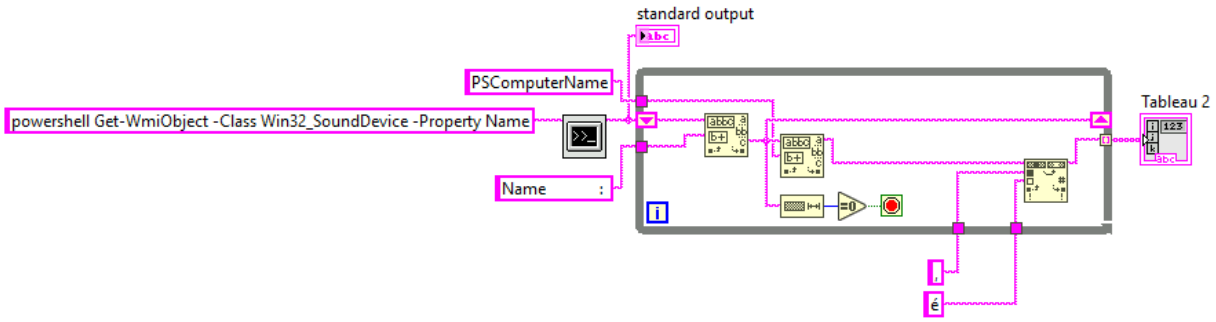

Figure 6 : « sous-vi » : recuperation_microphone.vi

Pour récupérer les microphones, nous utilisons l'invite de commande PowerShell de Windows. La commande « *Get-WmiObject –Class Win32_SoundDevice –Property Name* » permet d'obtenir le résultat de la figure 7.

```
__GENUS          : 2
__CLASS          : Win32_SoundDevice
__SUPERCLASS     :
__DYNASTY        :
__RELPATH        :
__PROPERTY_COUNT : 1
__DERIVATION     : {}
__SERVER         :
__NAMESPACE      :
__PATH           :
Name             : Realtek Audio
PSComputerName   :

__GENUS          : 2
__CLASS          : Win32_SoundDevice
__SUPERCLASS     :
__DYNASTY        :
__RELPATH        :
__PROPERTY_COUNT : 1
__DERIVATION     : {}
__SERVER         :
__NAMESPACE      :
__PATH           :
Name             : Son Intel(R) pour ,crans
PSComputerName   :
```

Figure 7 : Résultat commande sur PowerShell, récupération des périphériques audio

Parmi toutes ces lignes renvoyées, l'information importante est le nom correct des périphériques audio. Après analyse du résultat de la commande, nous avons remarqué que cette information se trouve entre « Name : » et « PSComputerName : ». Ainsi, pour automatiser la récupération des noms des périphériques audio, nous récupérons tout ce qui est après « Name : ». Ensuite, nous récupérons ce qui est avant « PSComputerName ». Puis, intervient la phase de formatage de l'élément récupéré. En effet, nous avons remarqué que les accents aigus étaient remplacés par des virgules. Nous remplaçons donc à la main les «, » par des « é », ce qui nous permet d'obtenir le nom correct du premier périphérique audio repéré dans la liste. Enfin, nous réitérons le processus précédent sur la chaine après « Name : » et l'algorithme s'arrête quand la chaine est vide. Les résultats récupérés sous le bon format sont stockés dans le tableau en figure 8.

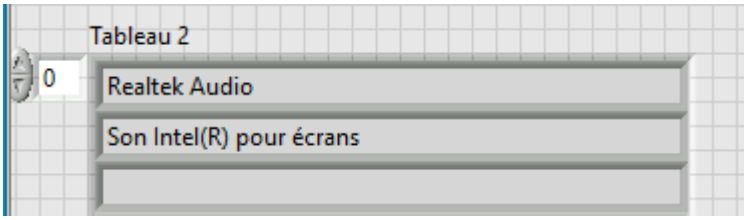

Figure 8 : tableau des noms de périphériques audio

Ce tableau est ensuite transformé en un menu où l'utilisateur peut choisir son microphone, comme le montre la figure 9.

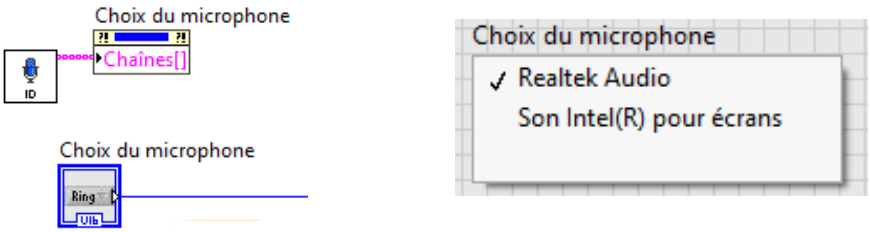

Figure 9 : menu choix microphone

Une fois tous ces paramétrages effectués, nous effectuons le passage à la boucle où se réalise l'acquisition du son, que nous avons mis dans le fichier *acquisition_son.vi* donné en figure 10.

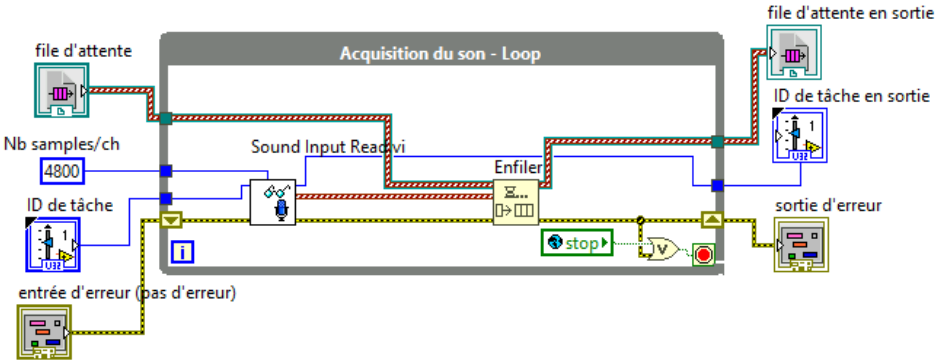

Figure 10 : « sous-vi » acquisition_son.vi

Le premier bloc de la boucle « Sound Input Read » permet de faire l'acquisition du son sous le format Waveform. En effet, LabVIEW traite des données du type Waveform qui sont structurées sous la forme d'un cluster intégrant les valeurs des mesures, l'instant initial et l'intervalle de temps entre deux échantillons. Ce type de données facilite le traitement des signaux en utilisant la transformée de Fourier.[3] Les acquisitions sont enfilées dans une file pour pouvoir être traitées par la suite.

I. 2. 2. Traitement

Une fois une acquisition effectuée, il faut la traiter pour récupérer les amplitudes de la DSP. Pour cela, le « sous-vi » *traitement_son.vi* prend en entrée deux files. Une file avec les acquisitions et une file qui permet de stocker les amplitudes moyennes de la DSP pour chaque acquisition, afin que M. Bodin puisse ensuite récupérer ces valeurs pour piloter les différents équipements participant à la gravure sur le cuivre.

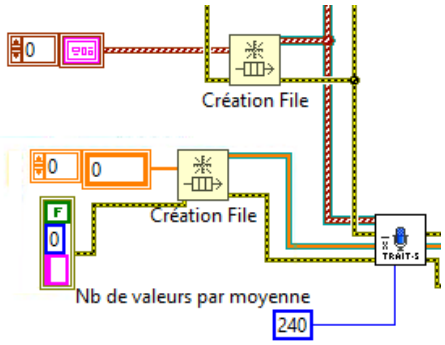

Figure 11 : Les deux files d'entrée au « sous-vi » traitement_son.vi

Nous allons à présent entrer en détails dans le fonctionnement du « sous-vi » *traitement_son.vi*, donné en figure 12, visible en figure 11 comme bloc « TRAIT-S ». La première partie à gauche opère sur le calcul de la DSP, elle est ensuite suivie d'une partie sur le calcul des valeurs moyennes d'amplitude du son.

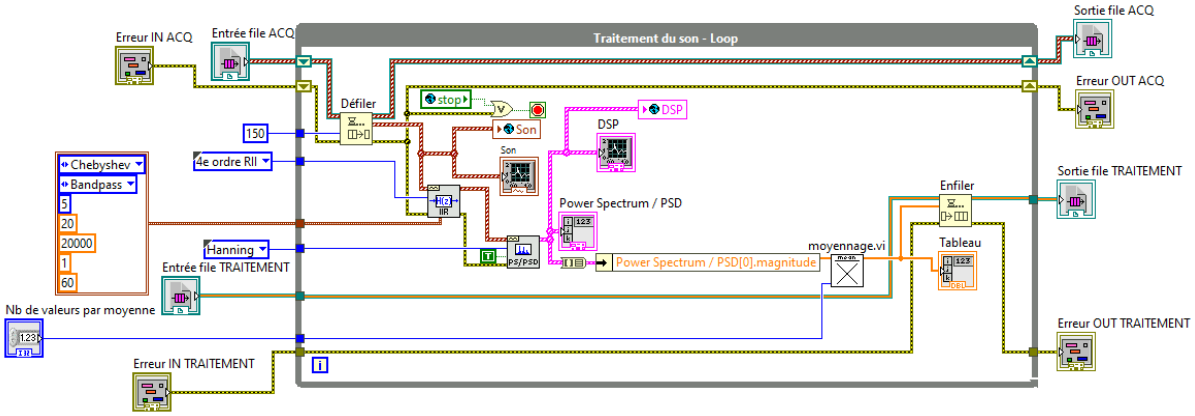

Figure 12 : « sous-vi » traitement_son.vi

Dans un premier temps, nous allons détailler le calcul de la DSP qui se trouve dans la partie gauche de la figure 11 et qui est mis en évidence dans la figure 13.

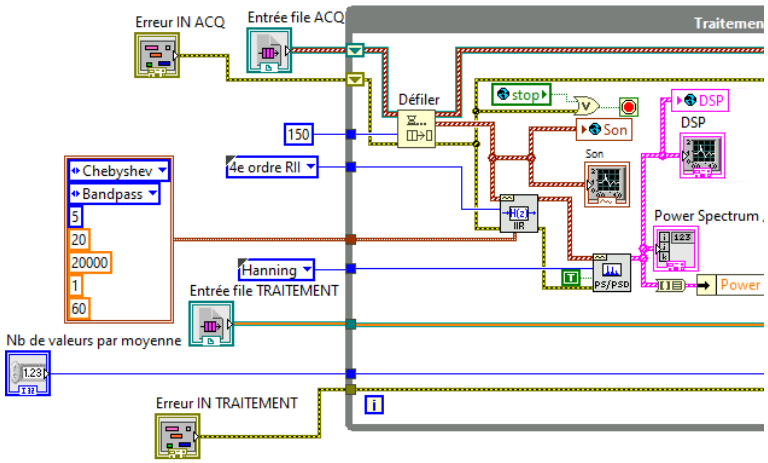

Figure 13 : Zoom sur la partie gauche de traitement_son.vi

Nous commençons par défiler une des acquisitions. Ensuite, pour des raisons pratiques, nous affichons le signal sonore reçu, comme présenté en figure 14.

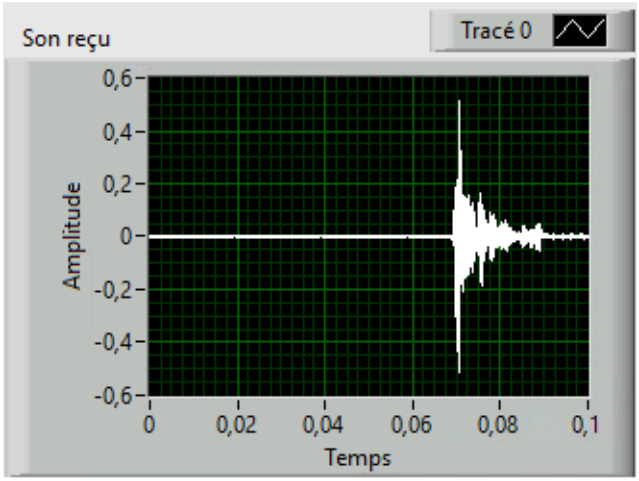

Figure 14 : Spectre temporel d'une acquisition de son

Ensuite, nous nous intéressons à la plage de fréquences audibles, donc de $20\ Hz$ à $20\ kHz$. Pour cela, nous avons choisi un filtre passe-bande de Tchebychev à l'ordre 5 avec une ondulation permise de $1\ dB$.

Puis, il faut obtenir une représentation fréquentielle $S(f)$ du signal sonore $s(t)$. Cette représentation est la transformée de Fourier de la représentation temporelle $s(t)$.[4] Pour calculer la transformée de Fourier d'un signal à l'aide d'un calculateur numérique, il faut utiliser la transformation de Fourier discrète. Comme ce calculateur est limité dans sa puissance de calcul, il ne peut fournir les résultats que pour un nombre limité de valeurs de la fréquence f. Ainsi, le signal $s(t)$ est remplacé par ses échantillons $s(nT_e)$ et les calculs portent sur une valeur finie N. Nous avons donc des nombres : $S(f) = \sum_{n=0}^{N-1} s(nT_e)e^{-j2\pi f_n T_e}$. [5]

Pour le calcul de la DSP, LabVIEW effectue directement une Fast Fourier Transform (FFT). Nous n'avions donc pas de choix à faire sur la transformée de Fourier. Cependant, nous avons fait le choix d'une fenêtre de Hanning pour diminuer les ondulations dues à la troncature de la fenêtre que nous avons avec une fenêtre rectangulaire, car notre signal sonore n'est pas périodique et que notre signal est à durée limitée. [6] Enfin, nous récupérons les valeurs des amplitudes en décibels (dB) comme demandé par notre tuteur. Un exemple de résultat est donné en figure 15.

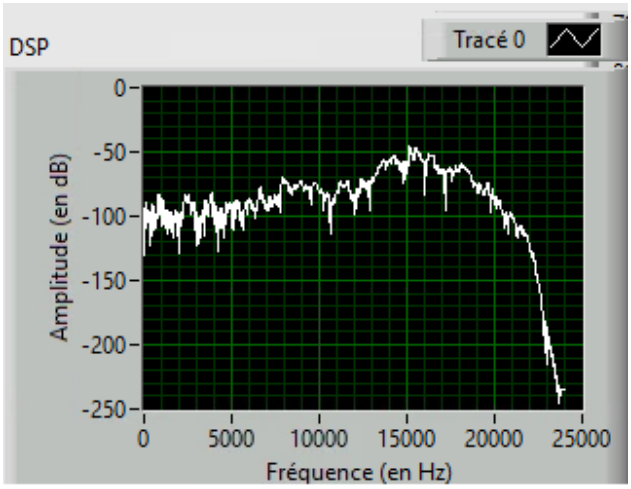

Figure 15 : Spectre fréquentiel d'une acquisition de son

Pour une acquisition, nous avons 2400 valeurs d'amplitude. En effet, nous avons une fréquence d'échantillonnage de $f_e = 48\,kHz$, ce qui signifie que la fréquence maximale pour respecter le théorème de Shannon est $f_{max} = \frac{f_e}{2} = 24\,kHz$. Or, le calcul de la DSP s'effectue avec un pas $df = 10\,Hz$. Nous obtenons donc bien 2400 valeurs. Pour le pilotage des équipements servant à la gravure sur le cuivre, c'est une énorme quantité de valeurs à traiter. M. Bodin nous a demandé d'avoir une dizaine de valeurs. Nous avons donc effectué des moyennes sur 240 valeurs afin d'obtenir à la fin 10 valeurs.

Pour faciliter le calcul des moyennes, nous souhaitions travailler sur des tableaux de dimension 1 ; or, en sortie du bloc de calcul de la DSP, le type de données est relativement complexe, comme le montre la figure 16.

Type de données du fil de liaison

Power Spectrum / PSD (tableau 1D de)
power spectrum (cluster de 3 éléments)
f0 (double précision [réel 64 bits (~15 chiffres significatifs)])
df (double précision [réel 64 bits (~15 chiffres significatifs)])
magnitude (tableau 1D de)
Numeric (double précision [réel 64 bits (~15 chiffres significatifs)])

Figure 16 : Type de données en sortie de DSP

Nous souhaitions donc récupérer uniquement le tableau 1D « magnitude » de la figure 16 qui contient les amplitudes en double précision. La figure 17 montre notre procédé.

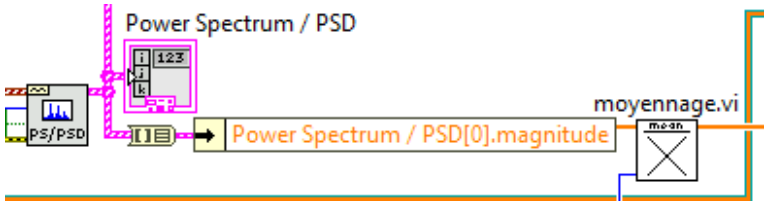

Figure 17 : Conversion de données en une dimension

Pour pouvoir le faire, nous avons d'abord converti le tableau 1D « Power Spectrum / PSD » de la figure 16 en un cluster de 9 éléments, comme le montre la figure 18.

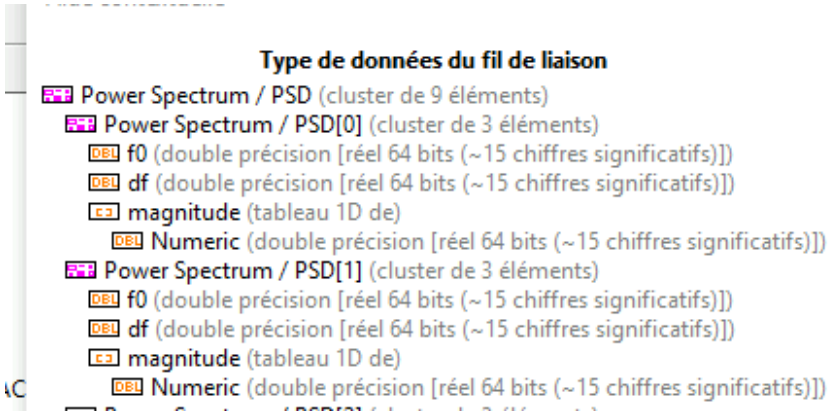

Figure 18 : Cluster des données de la DSP

Cependant, les huit sous clusters « Power Spectrum / PSD[n] » pour n > 0 sont vides. Les données nous intéressant sont dans le premier cluster. À présent que nous avons a un cluster, il est facile de récupérer le tableau 1D « magnitude » en récupérant « Power Spectrum / PSD[0].magnitude » à l'aide du bloc « Désassembler par nom » présenté en figure 19.

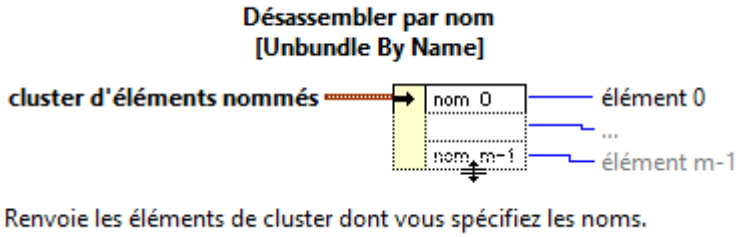

Figure 19 : Bloc « Désassembler par nom » un cluster

Ayant le bon type de données, nous entamons le calcul des moyennes des amplitudes de la DSP pour chaque acquisition. Pour cela, nous avons rédigé le sous-vi « moyennage.vi » de la figure 20.

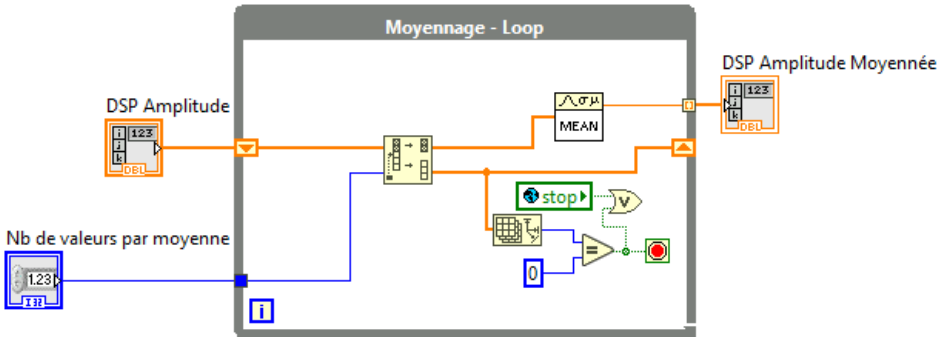

Figure 20 : « sous-vi » moyennage.vi

Le « sous-vi » de la figure 20 fonctionne de la manière suivante : nous récupérons en entrée le tableau 1D des amplitudes et le nombre de valeurs que nous souhaitions garder pour calculer chaque moyenne. Tout d'abord nous faisons la moyenne sur les 240 premières valeurs du tableau que nous stockons dans un nouveau tableau 1D. Ensuite, nous réitérons le procédé sur la suite du tableau et la boucle s'arrête lorsque le tableau est vide. La figure 21 montre un exemple de résultat.

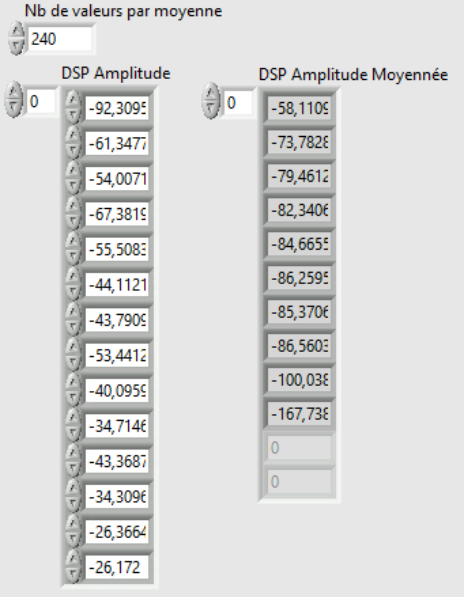

Figure 21 : Tableau des valeurs moyennes dans une acquisition

I. 2. 3. Interface utilisateur et rendu

Notre programme principal présenté en point I. 2. Implémentation, affiche une face avant qui nous sert d'interface utilisateur. Cette face avant est présentée en figure 22.

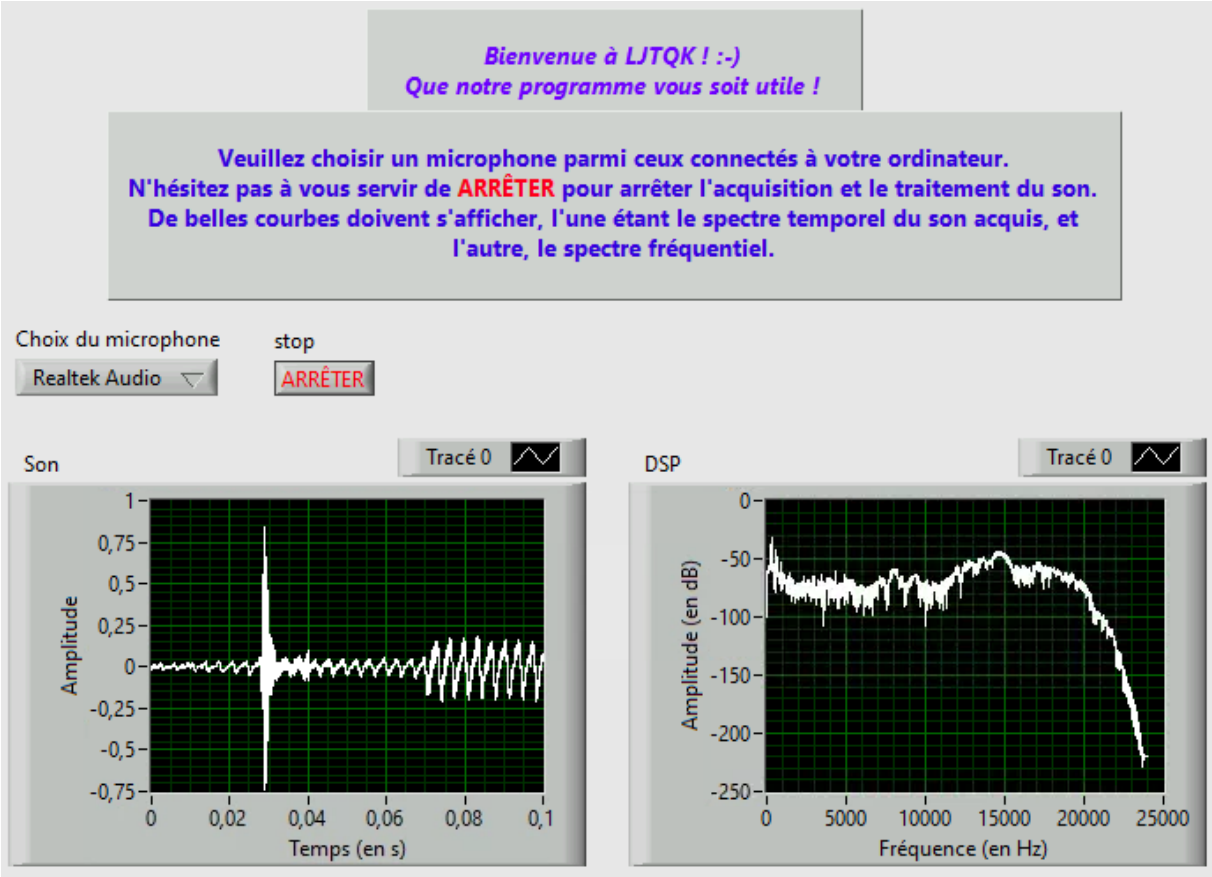

Figure 22 : Interface utilisateur du programme LJTQK

Enfin, nous fournissons à M. Bodin l'installeur et le code source de notre programme *LJTQK_LabVIEW*, l'un permettant à exécuter le programme sur un autre ordinateur (sans devoir à installer au préalable de logiciel LabVIEW) et l'autre afin qu'il puisse poursuivre le développement pour qu'il réponde au mieux aux besoins de Mme. Charière—Fiedler.

II. Résultats – Discussion

Le temps imparti pour ce projet arrivé à l'échéance, nous nous arrêtons avec un programme fonctionnant pour un seul microphone à la fois.

Pour utiliser notre programme, l'utilisateur doit télécharger l'installeur LJTQK_LabVIEW, Après installation, l'utilisateur doit sélectionner un microphone parmi les périphériques audio connectés à l'ordinateur. Grâce à la requête faite sur le PowerShell en point I. 2. 1. Acquisition, l'utilisateur ne peut sélectionner que des périphériques audio et pas d'autres périphériques.

Le choix de période d'acquisition du son n'est pas fourni à l'utilisateur, mais est déterminé en amont dans la programmation. Nous effectuons une acquisition toutes les

$100\ ms$. L'utilisateur peut voir l'allure du son acquis sur un spectre temporel du signal, mais également le spectre fréquentiel.

Dans le code source du programme, une file s'occupe de stocker les amplitudes moyennes. C'est en consultant le code source et en ouvrant le fichier : *moyennage.vi*, que nous avons la visibilité sur les moyennes calculées.

Parmi les tests de fonctionnement du produit réalisé, notre programme fonctionne sous Windows 10 et 11. Quand l'utilisateur sélectionne un périphérique de sortie d'audio, le programme affiche une erreur, poussant l'utilisateur à choisir un périphérique d'entrée. Le microphone sélectionné par défaut est celui de numéro d'identification zéro, qui peut être celui directement présent depuis la conception de l'ordinateur, mais s'il s'agit d'un haut-parleur, une erreur sera déclenchée.

Concernant la file de stockage des moyennes, elle est actuellement non vidée, mais régulièrement remplie de moyennes. Sachant que l'on a 10 moyennes par acquisition et qu'on effectue une acquisition toutes les $100\ ms$, il serait important de voir son comportement lorsque le programme est lancé pour une longue durée.

M. Bodin a régulièrement validé notre programmation et a testé notre installeur à la fin du projet.

Conclusion

En fin de ce projet, nous avons réalisé l'acquisition et le traitement de signaux sonores. Ce fut un projet enrichissant du point de vue pratique et apprentissage en profondeur du traitement du signal. Ce fut aussi un projet intéressant dans la mesure où il nous a permis de s'adapter au travail en groupe.

Perspectives d'amélioration

Ayant songé aux différentes pistes d'amélioration de notre travail, nous sommes arrivés à conclure qu'il serait intéressant d'étendre la récupération et le traitement de sons à partir de différents microphones.

Il serait également intéressant d'automatiser l'acquisition et le traitement des sons, après la détection et la récupération des données d'identifications de chaque microphone connecté. Ceci afin de ne plus devoir passer par l'utilisateur ; c'est-à-dire que le programme devra détecter automatiquement tous les microphones connectés à l'ordinateur et devra démarrer l'acquisition et le traitement pour chacun d'eux.

Références bibliographiques

[1] S. FEMMAM, Traitement numérique du signal, signaux et systèmes, iSTE editions, 2017 (traduit de l'anglais et publié sous la direction de M. CHARBIT)

[2] Assistance Scolaire Personnalisée, « ASP », *La numérisation des sons* [Online]. Disponible sur : https://www.assistancescolaire.com/eleve/1re/enseignement-scientifique/reviser-le-cours/1_sci_30. [Consulté le : 24-fév-2023]

[3] F. COTTET, L. DESRUELLE, M. PINARD, LabVIEW, Programmation et applications, Introduction à LabVIEW NXG, 4ème éd., Paris : Dunod, 2018

[4] P. ARQUÈS, N. THIRION-MOREAU, É. MOREAU, Les représentations temps-fréquence en traitement du signal, Technique de l'Ingénieur. 2000. Disponible sur https://www-techniquesingenieur-fr.ezproxy.uca.fr/res/pdf/encyclopedia/42416210-r308.pdf. [Consulté le : 10-nov-2022]

[5] M. BELLANGER, Traitement numérique du signal, Cours et exercices corrigés, 9ème éd., Dunod, 2012

[6] G. BAUDOIN et J.-F. BERCHER, « Transformée de Fourier Discrète : TFD », in *Transformée de Fourier Discrète*, 2001, p. 5-16 [Online].
Disponible sur : https://perso.esiee.fr/~bercherj/New/polys/poly_tfd.pdf. [Consulté le : 24-fév-2023]