



Technologie de l'Informatique

Avenue du Ciseau 15

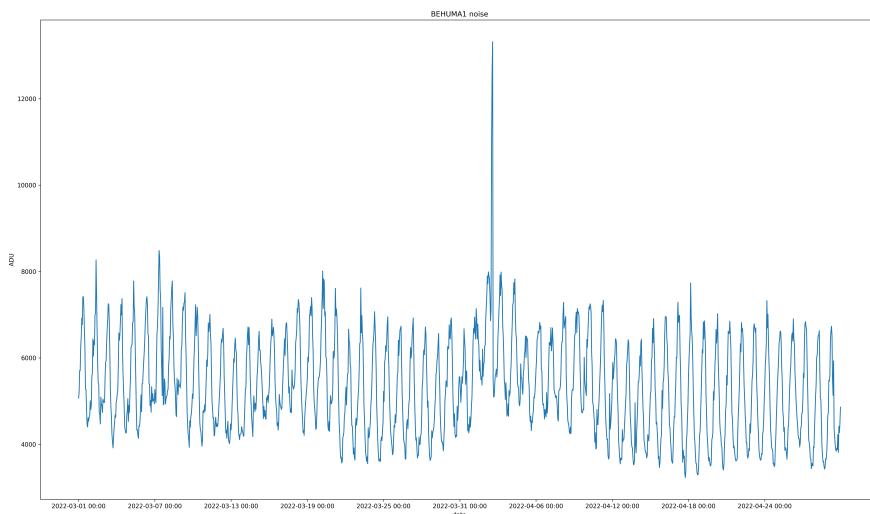
1348 Ottignies

Monitoring des données BRAMS et déttection automatique des échos de météore

Miguel Antoons

Rapporteur

Monsieur Arnaud Dewulf



2021-2022

Remerciements

Je tiens tout d'abord à remercier toutes les personnes qui m'ont aidé à réaliser ce projet de fin d'études.

En commençant par mon professeur rapporteur, à qui j'ai pu poser mes questions en cas de besoin et qui s'est assuré que tout se passe bien tout au long du projet.

Ensuite, je voudrais remercier Mr Hervé Lamy d'avoir proposé ce sujet, mais également d'avoir expliqué, de façon claire et précise, toutes les notions qui nécessitaient des explications.

Je tiens également à remercier Mrs Antoine Calegaro et Michel Anciaux, qui m'ont guidé quand c'était nécessaire et qui m'ont conseillé durant le projet de fin d'études.

Enfin, je voudrais exprimer ma reconnaissance envers toutes les personnes qui m'ont conseillé sur, et ont relu ce rapport de projet de fin d'études.

Table des matières

1	Introduction	4
2	Détection des météores par ondes radios	5
3	Le projet BRAMS	7
3.1	L'Émetteur	7
3.2	Les Stations de Réception	8
3.2.1	Les Stations V1	8
3.2.2	Les Stations V2	9
3.3	Les Données BRAMS	11
3.4	L'Archive BRAMS	15
4	Problématique liée à l'étude des données BRAMS	16
5	Méthodologie	17
6	Technologies utilisées	18
6.1	Python	18
6.2	MariaDB	18
7	Monitoring des Données BRAMS	19
7.1	Déetecter une anomalie sur une station	19
7.1.1	Le Bruit	20
7.1.2	Le Signal Calibreur	21
7.2	Fonctionnement du Logiciel de Surveillance	22
7.2.1	Analyse des Fichiers WAV	22
7.2.2	L'avertissement d'une anomalie	24
7.3	L'interface sur le site web BRAMS	26
8	Logiciel de Détection des Météores	29
8.1	Déterminer si un Écho Appartient à un Météore	29
8.2	Fonctionnement du Logiciel	29
9	Conclusion	34
10	Perspectives	35
A	Kernel de Convolution Permettant d'Amplifier les Signaux de Météores	39

B Résultat du filtre pour amplifier les échos de météores	40
C Résultat du filtre à percentile	40
D Résultat des éléments qui sont trop petits pour pouvoir être un écho	41

1 Introduction

Chaque jour, des milliers d'objets passent tout près de l'atmosphère terrestre. Parmi ces objets, on retrouve les météoroïdes : des corps pierreux ou métalliques d'une largeur pouvant varier de quelques millimètres à un mètre. Un météoroïde, une fois rentré dans l'atmosphère, devient un météore et peut créer un phénomène lumineux connu sous le nom d' "Étoile filante". Contrairement à ce que l'on peut penser, un météoroïde qui entre dans l'atmosphère terrestre est un événement qui se produit des milliers de fois par jour.

L'étude de ces météores permet de retrouver différentes informations telles que la masse ou encore la trajectoire de ceux-ci. Ce travail de fin d'étude a pour objectif de faciliter cette étude. Cet objectif sera accompli en permettant une visualisation facile de la qualité des données étudiées et en automatisant une partie de la détection des météores. Mais, afin de pouvoir les étudier, il est nécessaire de les détecter avant. Actuellement, différentes techniques existent pour détecter des météores dans l'atmosphère.

L'une d'entre elles est la détection à l'aide de caméras. Ceci a l'avantage de directement voir la trajectoire du météore et facilite donc l'étude. Cependant, elle a un grand défaut : lorsque le ciel est nuageux ou qu'il fait jour la technique est moins efficace. De plus, lorsqu'un petit météoroïde entre dans l'atmosphère, elle ne produit pas assez de lumière pour pouvoir être détecté par une caméra. C'est alors qu'une autre technique, celle par détection à l'aide d'ondes radio devient intéressante.

Ce travail de fin d'études consistera à faciliter l'étude des trajectoires de météores. Cet objectif sera accompli en automatisant, facilitant deux étapes du traitement des données produites par la détection de météores à l'aide d'ondes radios. Notamment la détection des météores et la détection de données erronées.

Ce travail, pour moi, permettra de prouver mes capacités à réaliser un projet d'une grande ampleur en autonomie. De plus, elle aidera également à mieux m'orienter dans les années à suivre.

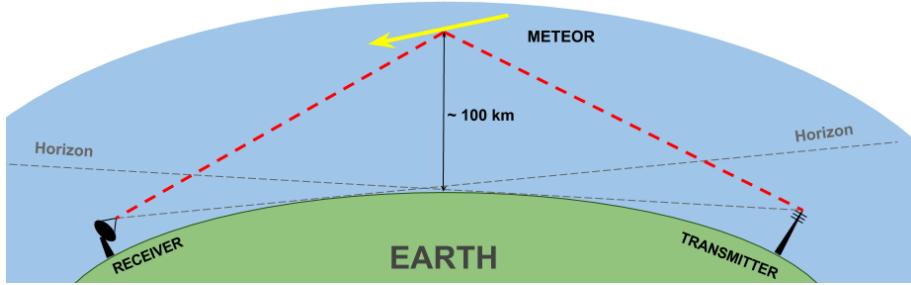


FIGURE 1 – Détection d'un météore à l'aide d'ondes radios.

2 Détection des météores par ondes radios

Quand un météoroïde entre dans la partie haute de l'atmosphère (approximativement à 80 - 120 km de la surface terrestre) il laisse derrière lui une trainée ionisée. Cette trainée à la propriété de réfléchir les ondes radio. On peut donc détecter un météore à l'aide de sa trainée ionisée.

Afin d'exploiter cette réflexion, il nous faut un émetteur dont le signal radio est réfléchi à l'aide de la trainée d'un météore et est ensuite enregistré par un récepteur. Cette procédure est illustrée à la figure 1, où la flèche jaune représente la trainée ionisée produite par le météore. Le signal reçu par l'émetteur est alors appelé un écho de météore. Un écho de météore peut durer entre 1 et 10 secondes, selon la durée d'existence de la trainée ionisée.

Une caractéristique importante de cette technique de détection est la réflexion spéculaire. Ceci veut dire que la trainée ionisée du météore agit comme un miroir sur lequel la réflexion se produit uniquement à un point précis, appelé le point de réflexion spéculaire. Le point de réflexion spéculaire dépend de la position de l'émetteur, la position du récepteur et la trajectoire du météore. La conséquence est que les données reçues par un récepteur particulier sont relatives qu'à une partie précise de la trainée.

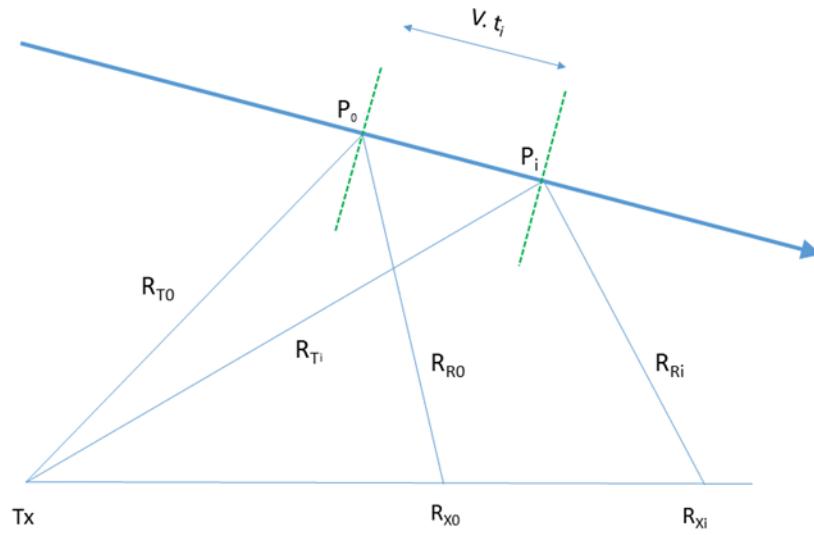


FIGURE 2 – Le point de réflexion spéculaire est différent pour chaque station réceptrice.

De plus, deux récepteurs, situés à des endroits différents, enregistreront un écho de météore à des instants différents puisque leurs points de réflexion spéculaire sont situés à des endroits différents sur la trajectoire. Ceci est illustré à la figure 2 où le point de réflexion P₀ entre le transmetteur Tx et le récepteur Rx₀ est situé à un endroit différent que le point de réflexion P₁ entre le transmetteur et le récepteur Rx₁.

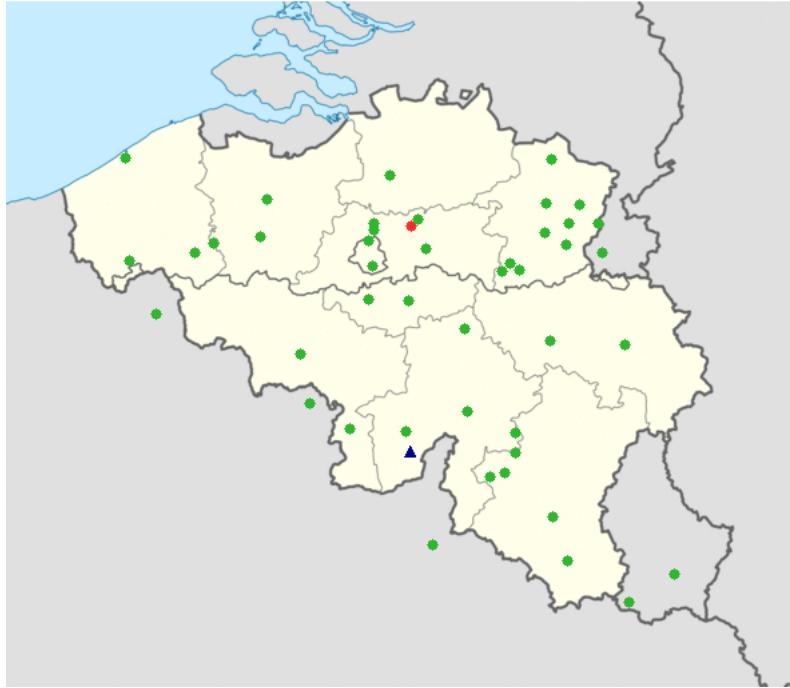


FIGURE 3 – Carte montrant l'emplacement de l'émetteur et des réceptrices.

3 Le projet BRAMS

Lancé en 2010 par Monsieur Hervé Lamy à l’Institut Royal d’Aéronomie Spatiale de Belgique, le projet BRAMS (Belgian RAdio Meteor Stations) a pour but de détecter et d’étudier les météores. Il dispose pour ceci d’un réseau d’un émetteur et de quarante-deux récepteurs situés dans la Belgique et dans les pays avoisinants qui utilise la détection des météores par ondes radios. Les emplacements de ces stations peuvent être visualisé sur la figure 3, où le triangle bleu représente l’émetteur et les boules vertes sont des réceptrices. Dans cette section, ce réseau et son fonctionnement seront expliqués.

3.1 L’Émetteur

Le réseau BRAMS dispose donc d’un émetteur unique situé à Dourbes, dans le sud de la Belgique. Cet émetteur transmet de façon continue un signal à la fréquence 49.970 MHz et d’une puissance d’approximativement 120 W.

Ce signal sera réfléchi sur d'éventuelles trainées de météores et pourra être détecté par des récepteurs.

3.2 Les Stations de Réception

Les stations réceptrices permettent donc de récupérer le signal de l'émetteur, réfléchi par les échos de météores. Durant la période d'existence du projet BRAMS, plusieurs stations réceptrices ont été développées. Actuellement, le réseau BRAMS est composé de deux types de stations différentes. Ces deux stations réceptrices et leurs différences seront expliquées ci-dessous.

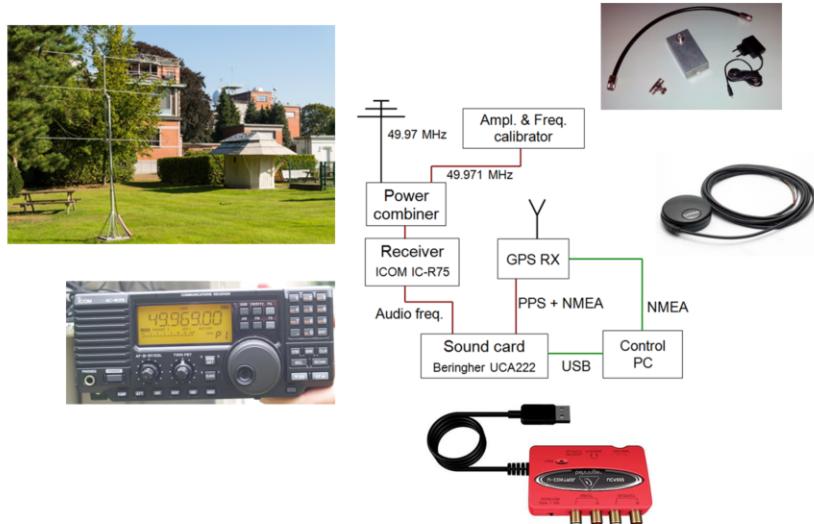


FIGURE 4 – Station v1 du réseau BRAMS

3.2.1 Les Stations V1

Les premières stations utilisaient des récepteurs analogiques ICOM-R75. Dans ce récepteur étaient injectés le signal venant de l'antenne à 49,97 MHz et le signal calibreur. Le signal calibreur est constant et dispose d'une fréquence de 49,975 MHz (500 Hz au-dessus du signal de l'antenne). Elle est additionnée au signal de l'antenne à l'aide d'un simple T. Ce signal a un but bien précis : permettre la conversion des valeurs reçues par le récepteur en unités Watts. Comme on connaît sa puissance, on peut alors calculer la

puissance des autres signaux captés par l'antenne.

Le récepteur décale le signal reçu par l'antenne et le calibreur à 1 kHz à l'aide de l'oscillateur local qui est réglé à 49,969 MHz. Le signal passe ensuite par une carte son externe, où il est échantillonnée à un taux de 22050 Hz. Cette carte son est également connectée à une horloge GPS¹ Garmin. L'horloge ajoute un signal de 1 PPS² suivi de données NMEA³ au signal reçu par l'antenne. Les données NMEA contiennent, entre autres, les informations à propos de la seconde associée au dernier PPS (horodatage). Ceci permet d'avoir un temps très précis pour chaque échantillon du signal.

La carte son est ensuite connectée à un ordinateur faisant tourner le logiciel 'Spectrum Lab'. C'est ce logiciel qui pilote la station et la procédure permettant de détecter des échos de météores. Afin de synchroniser l'horloge de l'ordinateur, celle-ci est également connectée à l'horloge GPS. Les données récupérées sont enregistrées toutes les cinq minutes dans des fichiers de format WAV⁴. La figure 4 montre un schéma de la station v1.

Ces stations ont fonctionné très bien pendant longtemps. Cependant, une fois arrivé en 2018 un problème avec le récepteur ICOM était de plus en plus récurrent sur les stations v1. Remplacer ces récepteurs s'est avéré compliqué comme le récepteur ICOM n'était plus produit. D'autres récepteurs analogiques pouvaient servir comme remplaçant, mais ceux-ci étaient bien plus chers. De plus, la station v1 disposait de plusieurs autres problèmes comme une faible portabilité ou encore une installation complexe. Ceci a poussé les membres du projet BRAMS au développement d'une nouvelle solution permettant de remplacer les stations v1.

3.2.2 Les Stations V2

En 2017 un étudiant venant de l'Ephec, nommé Antoine Calegaro, a créé un prototype d'une nouvelle station. À partir de ce prototype Michel Anciaux, membre du projet BRAMS, a développé la station v2, destiné à remplacer les stations v1

La station v2 est piloté par un Raspberry PI dont l'horloge est synchro-

-
1. Global Positioning System
 2. Pulse Per Second
 3. National Marine Electronics Association
 4. Waveform Audio File

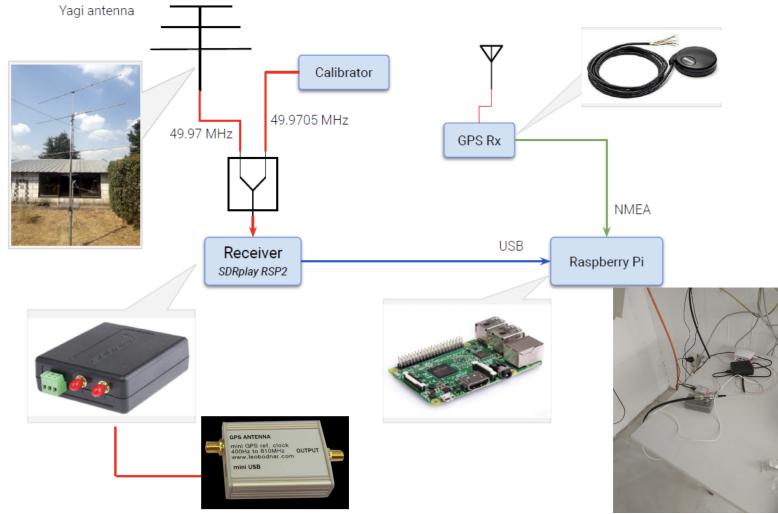


FIGURE 5 – Station v2 du réseau BRAMS.

nisé avec le signal PPS du même GPS Garmin utilisé sur la station v1. Son récepteur est un RSP2 qui à l'avantage d'avoir une portée dynamique plus grande et une sensibilité supérieure par rapport au récepteur ICOM. Il est également moins cher que son prédecesseur. Cependant, il ne permet pas l'échantillonnage du signal de l'antenne ensemble avec le signal GPS de 1 PPS. La fréquence du récepteur est stabilisé à l'aide d'une fréquence de référence externe à 24 MHz donné par un GPSDO⁵.

L'utilisation de deux horloges GPS (un pour le timing sur le Raspberry et un pour la fréquence sur le récepteur) permet d'avoir la fréquence de l'émetteur exactement à 1 kHz et le signal calibreur exactement 500 Hz au-dessus. Ceci était un problème sur les anciens récepteurs qui ne permettaient pas de stabiliser la fréquence à l'aide d'une source extérieure, ce qui causait des dérives de fréquence selon la température de celui-ci. La figure 5 montre un schéma des composants de la nouvelle station réceptrice. En 2020, la station serait placée dans une boîte métallique et la fréquence de référence du GPSDO serait également utilisé pour stabiliser l'oscillateur local du calibreur.

Cette station est donc beaucoup plus simple à installer puisqu'il ne nécessite plus d'ordinateur externe pour son fonctionnement. En plus d'être

5. GPS Disciplined Oscillator

plus compact et facile à installer que la station v1, la station v2 améliore la qualité des données et est moins cher.

3.3 Les Données BRAMS

Un fichier WAV venant d'une station réceptrice contient donc le signal capté par l'antenne ensemble avec un signal calibreur. Il est composé d'une seule piste audio, échantillonnée à une fréquence de 5512.5 Hz pour les stations v1 ou 6048 Hz pour les stations v2. Cette fréquence permet d'enregistrer des données dans une bande de fréquences allant jusqu'à 2756.25 Hz ou 3024 Hz (ou la moitié de la fréquence d'échantillonnage), selon le théorème de Nyquist. Sachant que les échos de météores apparaissent typiquement dans une bande de fréquence de 100 Hz autour du signal de l'émetteur qui est décalé à 1000 Hz, cette bande de fréquence couvre l'ensemble des signaux utiles à l'étude des météores.

À chaque fichier WAV est rajouté un bloc de données (data chunk) conçu pour faciliter l'étude des données. Dans ce bloc, on retrouve quelques informations relatives à la station de réception, la station émettrice, le signal GPS ou encore, la fréquence d'échantillonnage.

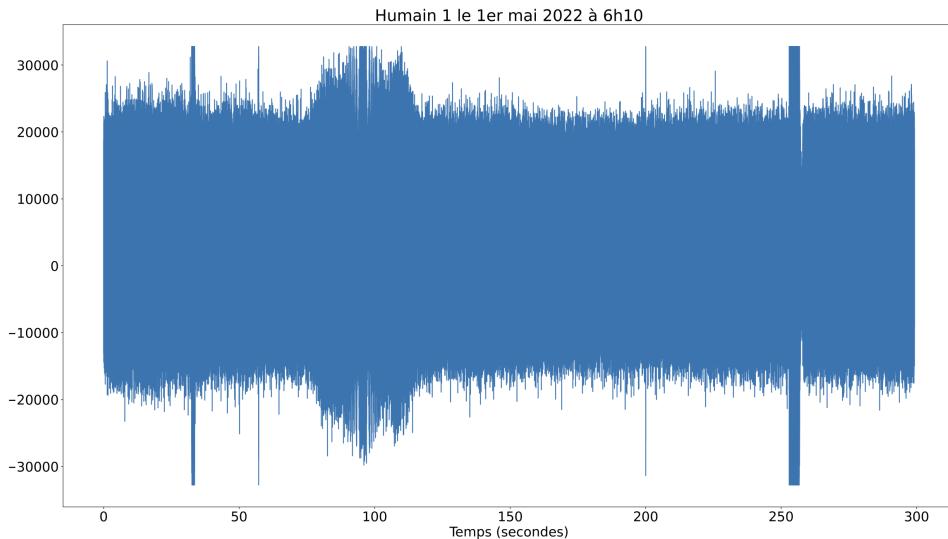


FIGURE 6 – Données brutes venant d'un fichier audio d'une station réceptrice.

Ces fichiers audio, comme montré à la figure 6, sont très difficiles à interpréter. Ils sont composés de bruits et de parasites sur l'entièreté de leur bande de fréquence. Par contre, lorsque l'on calcule le spectrogramme du fichier WAV, les données deviennent beaucoup plus simples à lire.

Un spectrogramme est une représentation différente des données contenues dans un fichier audio. Au lieu de représenter la puissance sur l'ordonnée et le temps sur l'abscisse, un spectrogramme représente la répartition des fréquences au cours du temps. Il contient donc trois dimensions :

- Le temps sur l'abscisse.
- La fréquence sur l'ordonnée.
- La puissance représentée par un code couleur.

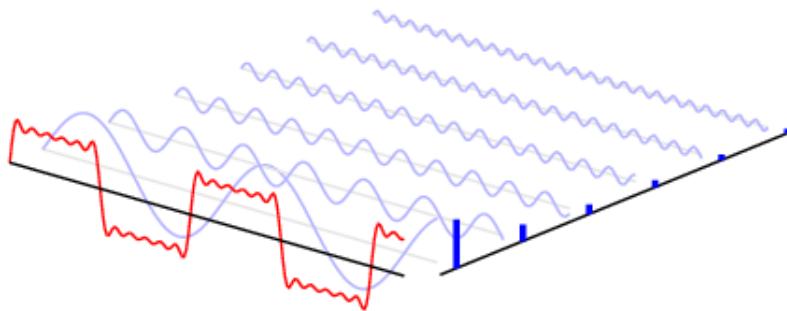


FIGURE 7 – Représentation visuelle de la Transformée de Fourier.

Pour générer un spectrogramme, il faut calculer un ensemble de Transformées de Fourier. La Transformée de Fourier permet de représenter la répartition de puissance entre les fréquences contenues dans un signal ou une partie d'un signal temporel. Elle produit donc ce qu'on appelle le spectre du signal. Elle se calcule sur un nombre quelconque d'échantillons qui se suivent dans un signal temporel. Ce spectre est souvent calculé à l'aide de la FFT⁶ qui est plus rapide, mais qui exige un nombre d'échantillons 2^n .

Sur la figure 7, on peut voir en rouge un signal temporel et le spectre de

6. Faste Fourier transform

ce même signal en bleu. Les signaux en mauve claire sont les signaux à une fréquence, dont le signal rouge est composé.

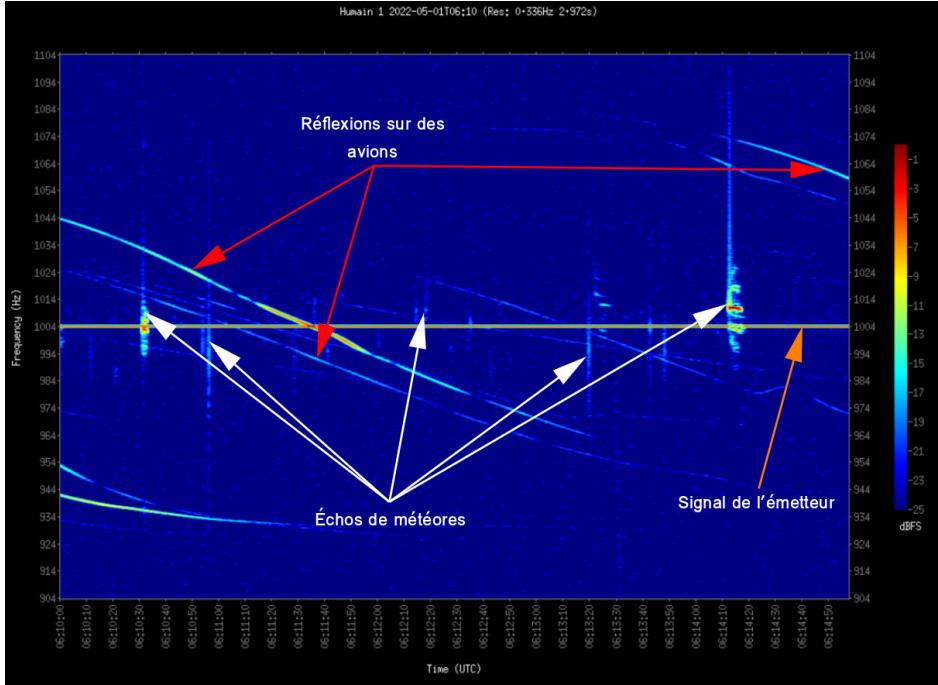


FIGURE 8 – Exemple d'un spectrogramme du projet BRAMS de 900 Hz à 1100 Hz. Les données représentées sont les mêmes qu'à la figure 6

Dans le cas des spectrogrammes pour le projet BRAMS, les FFT contenus dans un spectrogramme sont calculés sur 16384 échantillons. Si on suppose qu'un fichier WAV venant d'une station réceptrice dure en moyenne trois-cents secondes (ou cinq minutes), un spectrogramme est composé d'environ 101 spectres pour les stations v1 et 111 pour les stations v2. Ce nombre est obtenu avec la formule ci-dessous, où F_s est la fréquence d'échantillonage, T la durée en secondes du signal et $nfft$ le nombre d'échantillons utilisés pour générer un spectre.

$$\frac{F_s * T}{nfft}$$

Le spectrogramme généré aura une résolution fréquentielle, donnée par la formule $F_s/nfft$, de 0.34 Hz pour les stations v1 et 0.37 Hz pour les stations v2. Sa résolution temporelle, donnée par la formule $nfft/F_s$, est de 2.97

secondes pour les stations v1 et de 2.7 secondes pour les stations v2. Un exemple de spectrogramme venant d'une station de réception est affiché aux figures 8 et 9. Sur la figure 8, on voit uniquement la partie du spectrogramme où l'on retrouve les échos de météores. On y retrouve également le signal direct de l'émetteur à environ 1000 Hz et des parasites qui sont typiquement des réflexions sur les avions. Sur la figure 9 on voit l'entièreté du même spectrogramme affiché à la figure 8. Le contenu de cette dernière se trouve alors entre les deux lignes rouges. On retrouve, sur la figure 9 le signal du calibreur à environ 1500 Hz. En-dehors des lignes orange, on peut observer que le signal est filtré. Ce filtrage est automatique et est appliquée sur tous les fichiers WAV du projet BRAMS.

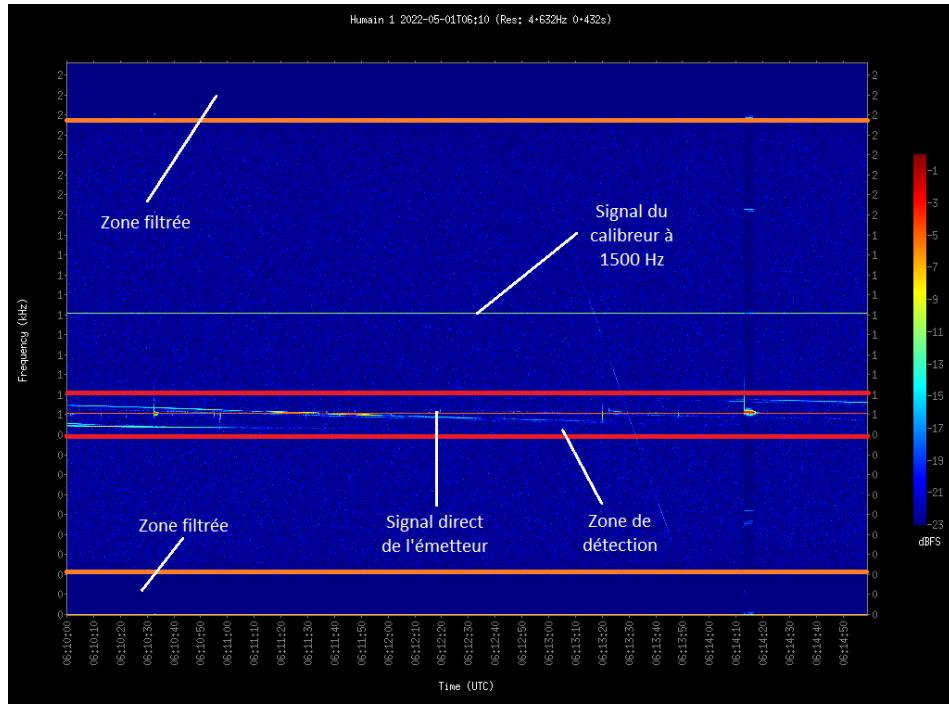


FIGURE 9 – Exemple d'un spectrogramme du projet BRAMS de 0 Hz à 3000 Hz. Les données représentées sont les mêmes qu'à la figure 6

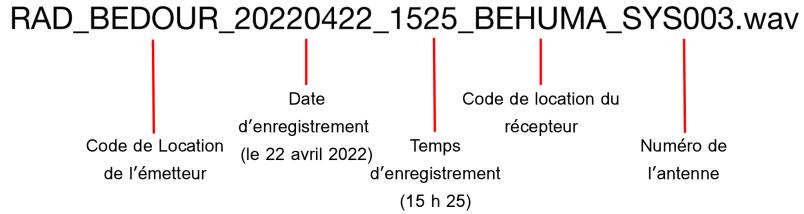


FIGURE 10 – Exemple de nom de fichier BRAMS.

3.4 L'Archive BRAMS

Les stations réceptrices produisent donc, en théorie, un fichier WAV toutes les cinq minutes. Ces fichiers ne sont bien évidemment pas stocké sur les stations mêmes, mais sont envoyés à intervalle régulier aux serveurs dédiés du projet BRAMS. Les fichiers qui arrivent aux serveurs sont archivés une fois par jour. Cette procédure d'archivage est activé manuellement en lançant un script bash.

Quand les fichiers venant des stations de réception sont archivés, ils sont placés dans une structure de répertoires spécifique. Le chemin pour accéder à un fichier WAV spécifique et son nom dépendent de la station d'où vient le fichier ainsi que la date de début d'enregistrement du fichier. C'est grâce à ces noms de fichiers, dont vous trouverez un exemple à la figure 10, et cette structure fixe qu'on peut retrouver facilement chaque fichier au sein de l'archive.

4 Problématique liée à l'étude des données BRAMS

Le but du projet BRAMS est donc d'étudier les fichiers WAV venant des stations réceptrices afin de retrouver, entre autres, la trajectoire des météores. Pour retrouver la trajectoire d'un météore, il est d'abord nécessaire de retrouver tous les fichiers contenant un écho de ce météore. C'est une action qui demande beaucoup de temps puisque l'utilisateur doit pour ça ouvrir chaque fichier de toutes les stations et vérifier s'il y a bien écho venant du météore.

De plus, actuellement il n'existe aucune façon de détecter une station qui produit des données erronées. Ceci veut dire qu'une station pourrait produire des fichiers inutilisables pendant plusieurs mois sans que quelqu'un s'en aperçoit. Ces fichiers prendraient non-seulement de l'espace de stockage inutilement, mais dans le cas où une étude nécessite les données de cette station, ça pose un grand problème.

C'est pour ces deux raisons que les membres du projet BRAMS souhaite faciliter et automatiser certaines étapes dans l'étude des données BRAMS. Ils ont donc demandé à réaliser un ou plusieurs solutions permettant les actions suivantes :

- La détection la plus précise des échos d'un météore dans les fichiers de stations réceptrices différentes.
Le résultat du programme doit être un fichier de format CSV⁷ qui indique chaque écho dont le programme pense qu'il vient du météore recherché. Pour chaque écho indiqué dans le résultat il faut bien entendu également rajouter des informations telles que la station où il a été détecté ou encore le temps de détection.
- Le monitoring constant des données des différentes stations à l'aide d'une interface sur le site web du projet BRAMS. Ce monitoring doit permettre aux scientifiques du projet de détecter facilement une anomalie dans les données.
- L'avertissement via mail si une anomalie est détectée dans les données BRAMS. Cette action implique également la détection automatique d'anomalies dans les données venant des stations réceptrices.

7. Comma Separated Values

5 Méthodologie

Afin d'arriver à un résultat final de qualité et qui est conforme aux requis des membres du projet BRAMS, il est important d'utiliser une bonne méthodologie.

La première chose que j'ai fait est : comprendre quoi exactement le programme devrait faire. Pour ce faire, plusieurs réunions ont eu place avant la réalisation du travail. Durant ces réunions, j'ai pu poser mes questions et demander des explications sur les concepts à connaître pour pouvoir réaliser le travail. Ayant reçu des documents qui expliquent de façon claire et précise le fonctionnement du réseau BRAMS, j'ai pu me préparer avant de m'attaquer à l'analyse et la réalisation du travail.

Durant la période d'analyse et de réalisation du TFE, j'ai pu régulièrement demander validation quant à la direction que je prenais pour mon travail. Je présentais régulièrement mes réalisations aux scientifiques du projet BRAMS afin d'avoir un feedback régulier et de pouvoir perfectionner mon programme un maximum.

Dans le but d'être plus efficace lors de l'écriture des fonctionnalités pour le programme, je décrivais avant ce que cette fonctionnalité devait faire. Ensuite, je la divisais en tâches techniques afin de pouvoir m'organiser plus facilement. Chaque tâche technique contenait et expliquait les étapes que le programme devrait exécuter pour accomplir cette tâche technique. Les étapes étaient décrites textuellement, par pseudo-code, par schéma ou encore, par le mélange de deux ou trois des moyens cités.

Durant l'entièreté du projet de fin d'études, j'ai tenté de garder un rythme de travail régulier. Je me suis organisé de telle façon de pouvoir travailler une moyenne de deux jours par semaine. Une grande partie de ces heures se sont déroulés lorsque je rentrais de mon stage ou pendant le week-end.

À la fin de ce projet de fin d'études tous les codes et programmes écrits ont été donnés aux membres du projet BRAMS. Comme demandé, le code et les programmes sont commentés et sont également accompagnées d'un mode d'emploi.

6 Technologies utilisées

6.1 Python

Le programme réalisé est entièrement écrit en Python. Ce choix a été pris premièrement pour les librairies performantes et open-source qu'offre Python. En effet, des librairies comme numpy, scipy ou encore matplotlib ont été très utiles pour arriver à un résultat performant et fonctionnelle. Ces librairies offrent beaucoup de flexibilité et offrent une haute performance grâce à leur écriture en C, C++ et même en Fortran.

Un autre avantage du Python est sa portabilité. Puisque c'est un langage interprété, largement répandu, quasiment tous les systèmes d'exploitation le supportent. Ceci a facilité aussi bien la phase de développement que la phase de testing du projet de fin d'études.

Un des désavantages souvent évoqués pour Python, est que le langage est lent pour des gros traitement de données. Cependant, dans le cas de ce travail, ceci ne fut pas un problème et les librairies utilisées étaient largement assez rapide. Ceci est dû, entre autres, grâce à la nature open-source de ces librairies ainsi que leur âge. Suite à ces deux facteurs, des milliers de personnes travaillent sur ces librairies depuis plusieurs années dans le but d'optimiser le plus possible ses fonctions et méthodes.

6.2 MariaDB

Afin de sauvegarder les données produites par le programme, une base de données est requise. Les données seront enregistrées dans la base de données existant du projet BRAMS. Le type de base de données du projet BRAMS est MariaDB.

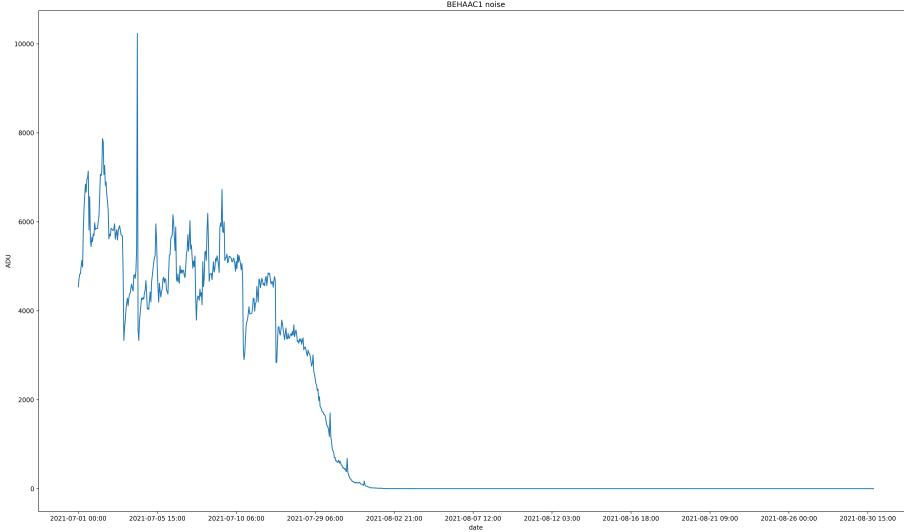


FIGURE 11 – Intensité du bruit de la station BEHAAC, antenne 1 entre le 1^{er} juillet 2021 et le 31 août 2021.

7 Monitoring des Données BRAMS

Comme cité plus haut dans ce rapport, le monitoring des fichiers peut faciliter l'étude des données BRAMS. Le développement d'un bon logiciel de surveillance relève plusieurs défis. Il doit être cohérent, et ceci pour tous les types stations. De plus, sachant que par jour une grande quantité de fichiers est produit par chaque station, l'optimisation du logiciel est également important afin de réduire l'utilisation des ressources système et le temps nécessaire pour obtenir les résultats du monitoring. Dans cette section, le développement du logiciel de surveillance ainsi que les démarches prises afin de résoudre ces défis seront expliquées.

7.1 Déetecter une anomalie sur une station

Afin de déterminer si un fichier contient une anomalie ou non, il est essentiel de bien choisir les éléments du fichier à surveiller. Dans les sections qui suivent, les différents éléments surveillés dans chaque fichier, ainsi que leur pertinence seront expliquées.

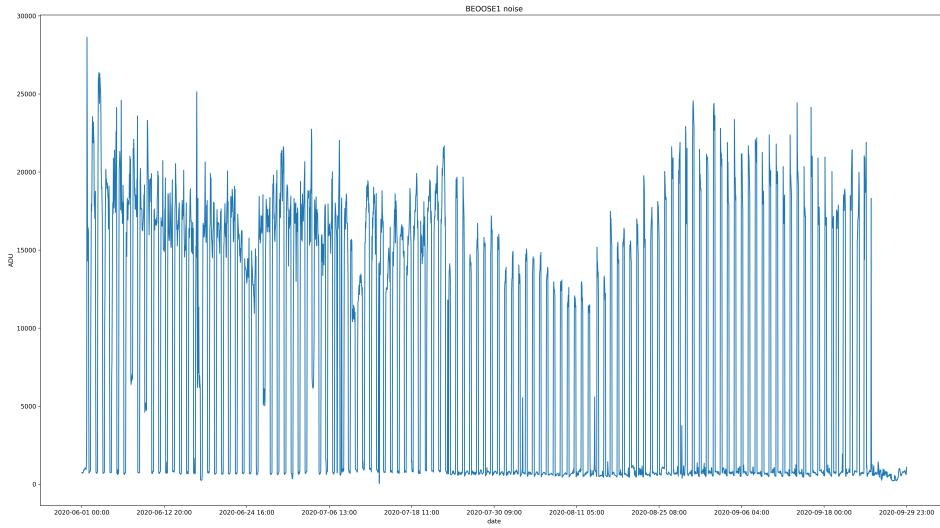


FIGURE 12 – Bruit de la station BEOOSE, antenne 1 entre 1er juin 2020 jusqu’au 30 septembre 2020.

7.1.1 Le Bruit

L’intensité du bruit est le premier élément qui sera surveillé. Bien que celui-ci puisse varier légèrement avec le temps, sur une longue période, elle devrait rester constante et les grosses variations du bruit sont souvent signe que la station est défectueuse.

Ceci était le cas par exemple pour la station BEHAAC lors du mois de juillet 2021. Comme on peut le voir sur la figure 11, on constate une diminution graduelle puis très rapide du bruit. Ce phénomène, pour les stations v1, est signe que le récepteur ICOM doit être remplacé. Il faut savoir qu’à partir du moment où le récepteur ne capte presque plus rien, jusqu’à la solution du problème, les données produites par la station sont inutilisables.

Un autre exemple de grosses variations de bruit est la station de BEOOSE entre le 1er juin 2020 jusqu’au 30 septembre 2020. Quand on visualise la figure 12, on remarque une augmentation du bruit par période. La cause de ce problème peut varier au cas par cas, il peut être un mauvais branchement au niveau de la station, des câbles endommagés ou encore un parasite extérieur à la station réceptrice. Il est important rapidement détecter et résoudre ce

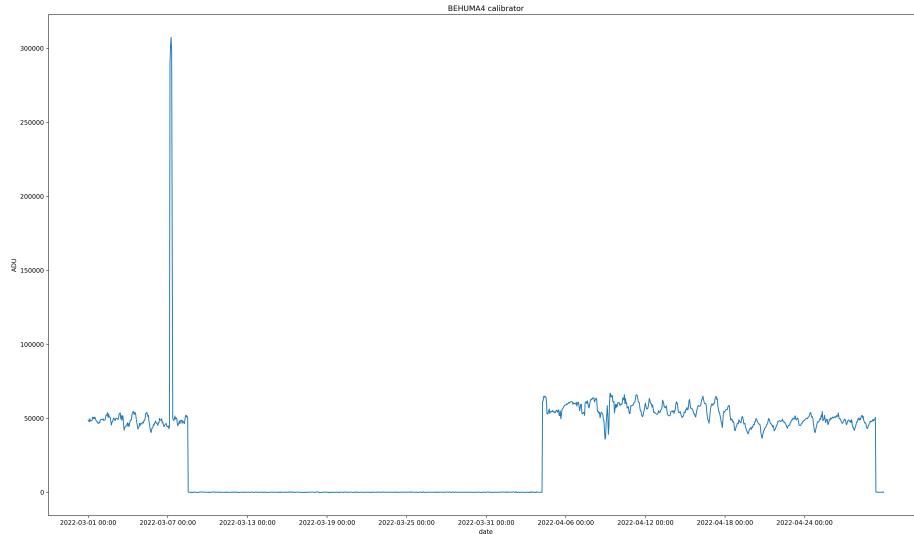


FIGURE 13 – Signal calibreur de la station BEHUMA, antenne 4 entre le 1^{er} mars 2022 et le 30 avril 2022.

problème puisque le bruit est tellement haut qu'il est impossible de détecter autre chose dans le fichier.

Une autre raison de surveiller le bruit est qu'il doit être présent dans chaque fichier. Un fichier avec un bruit nul est impossible et veut dire qu'au moment où la station a généré le fichier, il avait un problème. L'intensité du bruit est donc non-seulement une valeur facile à surveiller, mais il permet également de détecter toute une série de problèmes qui sont de nature différente.

7.1.2 Le Signal Calibreur

Le second élément que le programme surveille est l'intensité du signal calibreur ainsi que la fréquence à laquelle se trouve celle-ci. Mesurer l'intensité de ce signal permet, comme la mesure du bruit, de détecter si un récepteur de la station v1 est cassé. De plus, comme le bruit, le signal calibreur est un élément qui est censé être présent dans chaque fichier produit par une station réceptrice, s'il est absent ça veut dire que la station a un problème et ça doit donc être détecté.

Vient ensuite la surveillance de la fréquence à laquelle se trouve le signal

calibreur. En théorie, il devrait toujours se trouver à 1500 Hz, en pratique cette valeur peut varier en particulier avec les anciennes stations où il peut y avoir une translation de fréquence en fonction de la chaleur. Une variation de cette fréquence entre les valeurs de 1350 Hz et 1750 Hz ne posent pas un grand problème et ne doivent donc pas être détectées. Par contre, une trop grosse variation indique que la station a un défaut, et selon l'anomalie peut rendre les données inutiles. C'était notamment le cas pour la station BEHUMA4, lors d'une période qui a commencé le 8 mars 2022 et qui s'est étendu jusqu'au 3 avril 2022. Pendant ce temps, toutes les fréquences étaient décalées d'environ 1000 Hz vers le bas dans chaque fichier généré, ceci peut être visualisé à la figure 13. Bien que ça ne pose pas vraiment de soucis pour le signal calibreur même, qui se trouve alors à environ 500 Hz, toutes les données utiles qu'on retrouve typiquement autour de 1000 Hz se retrouvent maintenant autour de 0 Hz. Sachant qu'en plus les fréquences basses sont filtrés dans les fichiers BRAMS, l'information utile est donc absent.

Le signal calibreur est donc également un élément, se trouvant dans chaque fichier, qui peut indiquer rapidement s'il y a un problème avec une station. Ensemble avec le bruit, ils forment une bonne base pour déterminer si les données générées sont utilisables et si une station a besoin d'une intervention où non.

7.2 Fonctionnement du Logiciel de Surveillance

Ensemble avec les scientifiques du projet BRAMS, nous avons décidé que le programme doit se lancer automatiquement après chaque archivage de données. Le programme se lance en ligne de commande et ne nécessite aucun paramètre pour fonctionner. Une fois que le programme est lancé, il va chercher un par un les fichiers WAV nécessaires dans l'archive BRAMS. Dans le but d'avoir une surveillance suffisante sans utiliser trop de ressources machines, nous avons décidés d'analyser un fichier par heure par station. Ceci revient à l'analyse de 24 fichiers par jour par station.

7.2.1 Analyse des Fichiers WAV

Pour chercher et lire un fichier WAV provenant d'une station réceptrice, le logiciel utilise une classe nommée BramsWavFile. La base de cette classe a été écrit par Michel Anciaux, scientifique qui travaille sur le projet BRAMS. Cependant, ce code a subi deux grandes modifications afin de l'adapter pour le programme de monitoring et pour le rendre plus modulable. Le premier

```

1 def get_psd(f, flow=800, fhigh=900):
2     # get fourier tranform from BramsWavFile class
3     freq, S, fbin = f.FFT(f.Isamples)
4     idx = (freq >= flow) * (freq < fhigh)
5
6     # calculate the total power of the wanted frequencies
7     p = (S[idx] * S[idx].conj()).real / 2
8
9     # get a mean normalized to 1Hz
10    psd = p.mean() / fbin
11
12    return psd
13

```

FIGURE 14 – Code de la fonction permettant de calculer la dsp.

grand changement implique la recherche des fichiers dans l’archive : tandis que le code non modifié nécessitait un chemin pour trouver le fichier, maintenant il est capable de chercher tout seul un fichier dans l’archive BRAMS ou dans un autre dossier sur base de quelques informations (date et heure du fichier, station, numéro d’antenne). Le second changement concerne l’optimisation et donc le temps nécessaire pour lire un fichier WAV. Toutes les étapes non nécessaires ont été effacées et plusieurs librairies ont été testées afin de trouver celui qui était le plus rapide. Pour la méthode FFT de la classe BramsWavFile, qui permet de calculer la transformée de Fourier d’un fichier WAV, la librairie Scipy offrait par exemple des performances supérieures à la librairie Numpy tout en offrant les mêmes fonctionnalités et la même portabilité.

Une fois qu’on a les valeurs du fichier WAV, on calcule d’abord la moyenne de la densité spectrale de puissance (dsp) du bruit. La dsp représente la répartition des puissances en fonction des différentes fréquences contenues dans un signal. L’unité de la dsp est exprimé en puissance par Hertz.

Afin d’avoir une estimation précise du bruit sans devoir calculer la dsp de tout le bruit dans un fichier, on prend uniquement la dsp entre 800 Hz et 900 Hz. Entre ces deux fréquences on ne trouve normalement pas d’autres éléments que du bruit, dans un cas contraire, c’est souvent une indication qu’il y a un problème avec une station. Le code permettant de calculer la dsp du bruit est affiché à la figure 14. Dans un premier temps, on calcule la FFT du fichier WAV, ceci nous donne trois variables : une liste avec les valeurs

de l'axe x (freq), un vecteur avec les valeurs de l'axe y (S) en fonction de l'axe x et la résolution fréquentielle de l'axe x (fbin). Ensuite, on calcule le spectre des puissances en multipliant le vecteur S par son conjugué, ce qui nous donne son carré et donc la puissance pour chaque fréquence de l'axe x. Enfin, on fait la moyenne de ce vecteur et on divise cette moyenne par la résolution fréquentielle ce qui nous donne une moyenne par Hz du bruit et donc la dsp.

Après, le programme passe à la recherche de la fréquence du signal calibreur. Il faut ceci en recherchant l'intensité maximale entre les fréquences de 1350 Hz et 1750 Hz dans un fichier. Comme le signal calibreur s'étend sur l'entièreté de la durée du fichier, il doit toujours avoir la valeur maximale entre ces deux fréquences. La fréquence retrouvée nous permet alors de calculer la dsp du signal. Il y a ici deux différences par rapport au calcul de la dsp du bruit. La première est la bande de fréquence utilisée pour le calcul : pour le signal calibreur, elle est calculée sur une bande de 10 Hz autour de la fréquence trouvée pour cette dernière. La deuxième est qu'une fois la dsp du signal calibreur est calculée, on y soustrait la dsp du bruit afin de disposer d'une estimation plus précise.

Les étapes listées ci-dessus permettent d'avoir les informations nécessaires afin de pouvoir déterminer s'il y a un problème avec un fichier, et donc par conséquence, avec une station. Les deux valeurs de dsp sont enregistrées dans la base de données du projet BRAMS et sont donc réutilisables.

7.2.2 L'avertissement d'une anomalie

Avant l'enregistrement, une autre étape est faite : la détection automatique d'une anomalie. Pour cette étape, le logiciel applique les mêmes actions sur le bruit et le signal calibreur avec une exception. Ces actions sont faites à chaque fois qu'une nouvelle valeur de dsp est générée.

Afin de détecter une grande variation dans les données, le programme se base sur l'ensemble des valeurs de dsp des deux semaines précédent la nouvelle valeur de dsp. À partir de cet ensemble, il faut définir une frontière haute et une frontière basse. Si la nouvelle dsp se trouve en dehors d'une frontière, il sera considéré comme anormal et les scientifiques du projet BRAMS seront avertis.

Différentes méthodes existent pour définir ces frontières. La méthode uti-

```

1 def detect_variations(y_data, current_value):
2     q1 = np.percentile(y_data, 25, interpolation='lower')
3     q3 = np.percentile(y_data, 75, interpolation='higher')
4
5     interquartile = q3 - q1
6
7     upper_limit = q3 + (2.4 * interquartile)
8     lower_limit = q1 - (2.4 * interquartile)
9
10    if current_value >= upper_limit:
11        return 1
12    elif current_value <= lower_limit:
13        return -1
14    else:
15        return 0
16

```

FIGURE 15 – Code de la fonction permettant de détecter des variations excessives dans les données.

lisée dans ce logiciel, utilisant la gamme interquartile, peut être visualisé à la figure 15 et est expliquée ci-dessous.

Il faut savoir que les deux paramètres de la fonction affichée à la figure 15, **y_data** et **current_value**, sont l'ensemble des valeurs de dsp des deux semaines précédent la nouvelle valeur de dsp et la nouvelle valeur de dsp même. Le programme commence par calculer le 25e et le 75e percentile de l'ensemble des données qui se trouvent dans **y_data**. Un percentile est une mesure indiquant la valeur en dessous de laquelle se trouvent un pourcentage donné d'observations, dans un ensemble d'observations. Par exemple, le 20e percentile est la valeur en dessous de laquelle se trouvent 20% des observations dans un ensemble d'observations. Ensuite, on calcule la gamme interquartile de **y_data**. Cette valeur est calculée en faisant la différence entre la 25e et la 75e percentile, que le programme a déjà calculé.

Le logiciel définira à partir de ces trois valeurs calculées, la frontière haute et basse. Les formules utilisées sont $Q3 + (x * IQR)$ pour la frontière haute et $Q1 - (x * IQR)$ pour la frontière basse où $Q1$ et $Q3$ sont le 25e et le 75e percentile et IQR la gamme interquartile. Le facteur x permet de définir la largeur entre les deux frontières, au plus sa valeur est élevée, au plus les frontières sont écartées.

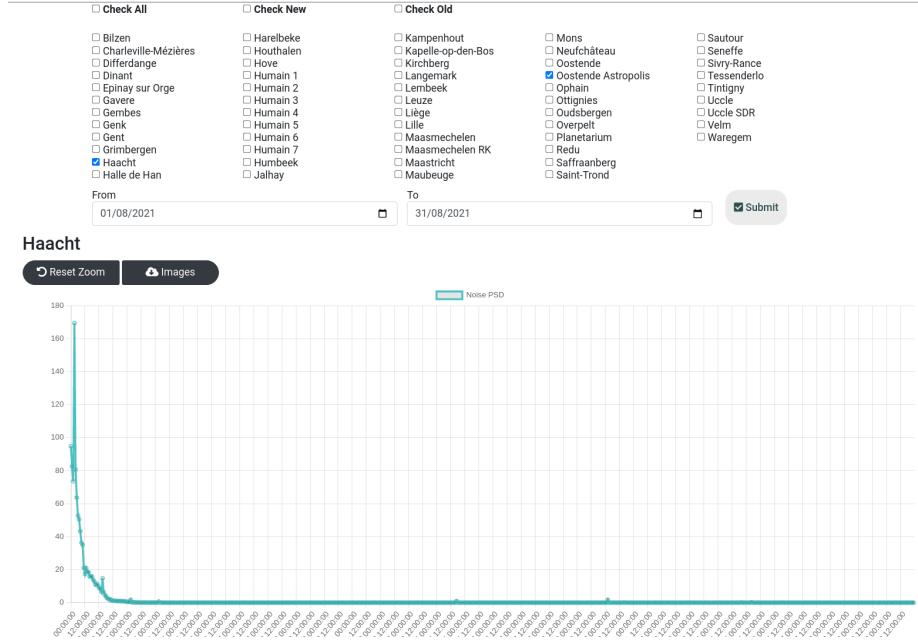


FIGURE 16 – Interface graphique du logiciel de surveillance.

C'est ce facteur qui est différent pour le bruit et le signal calibreur. Tandis que pour le bruit, il vaut 2.4, il est défini à 2.5 pour le signal du calibreur. Ceci est parce que le bruit varie moins et il n'est donc pas nécessaire d'avoir les deux frontières aussi écartées que pour le signal du calibreur.

Finalement, on vérifie si la nouvelle valeur de dsp (**current_value**) dépasse ou non une des frontières. Si c'est le cas, on informe l'utilisateur d'un problème dans les données.

Bien que cette méthode permet bien de détecter des anomalies, elle n'est pas parfaite. En effet, parfois il y a des faux positifs et le programme alerte l'utilisateur alors qu'aucun problème est présent.

7.3 L'interface sur le site web BRAMS

En plus de la détection automatique d'une anomalie, une interface graphique permettant de visualiser l'évolution de la dsp du signal calibreur et du bruit a été développée. Il faut noter que cette interface est indépendant

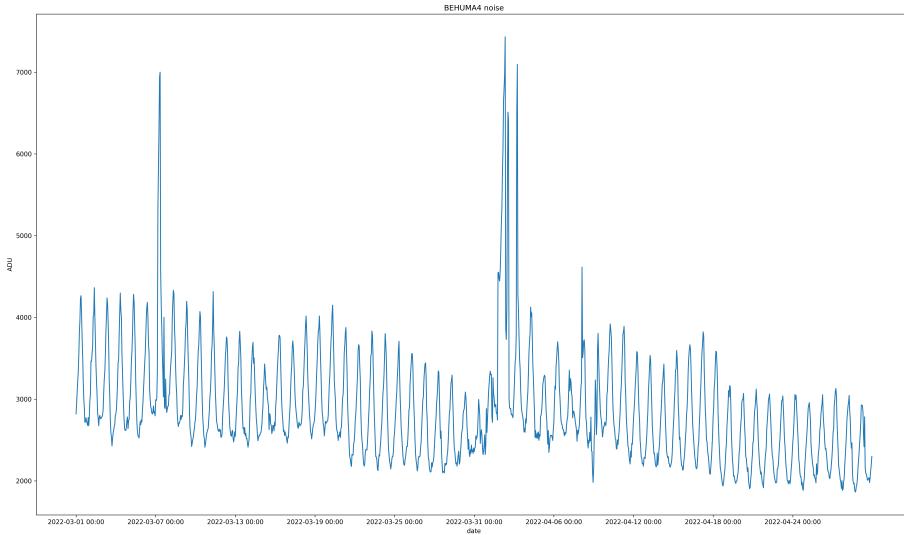


FIGURE 17 – Signal du bruit de la station BEHUMA, antenne 4 entre le 1^{er} mars 2022 et le 30 avril 2022.

du programme de monitoring même, il utilise, cependant, les données générées par ce-dernier. Comme on peut le voir sur la figure 16, l’interface affiche deux graphiques pour chaque station que l’utilisateur a sélectionné. Le premier représente les valeurs de dsp pour le bruit, pour une période que l’utilisateur a indiqué et le deuxième affiche la même chose, mais cette fois ci pour le signal calibreur.

Cette interface permet aux scientifiques du projet BRAMS de surveiller ces valeurs et éventuellement de détecter un problème qui ne serait pas détecté par la surveillance automatique. De plus, elle donne la possibilité de mieux comprendre les variations en observant des motifs qui reviennent. Un des motifs retrouvés lors des tests est une diminution de la dsp du bruit lorsque la nuit approche. Ceci est particulièrement visible sur la figure 17 qui représente la dsp du bruit pour la station BEHUMA4 du 1er mars 2022 au 30 avril 2022.

Cette interface graphique, comme l’entièreté du nouveau site web BRAMS utilise Joomla comme back-end. Joomla est un framework gratuit et open-source qui utilise le langage PHP. Pour afficher les différents graphiques, le

front-end de l'interface utilise la librairie JavaScript open-source ChartJS⁸. Cette librairie permet d'afficher des beaux graphiques et de les agrandir si nécessaire. Cette interface sera disponible publiquement sur le nouveau site web BRAMS dès le déploiement de cette dernière.

8. <https://www.chartjs.org/>

8 Logiciel de Détection des Météores

Une fois que les données sont archivées, les membres du projet BRAMS commencent à les analyser et les interpréter. Un objectif assez important de ce projet est le calcul de la trajectoire d'un météore dans l'atmosphère. Pour retrouver la trajectoire d'un météore à l'aide du réseau BRAMS, il faut qu'un météore soit détecté par au moins huit stations réceptrices. Vérifier manuellement, pour chaque station, si celle-ci a détecté un météore est une tâche qui prend un temps non négligeable. Dans cette section, le développement d'un programme permettant de réduire la perte de temps causée par cette procédure est expliquée. Ce programme est entièrement indépendant du programme de monitoring, expliqué dans la section 7.

8.1 Déterminer si un Écho Appartient à un Météore

Avant de développer un logiciel, il est nécessaire de savoir comment trouver, sur différentes stations réceptrices, les échos venant d'un seul météore. Différents facteurs peuvent indiquer qu'un écho vient d'un météore spécifique ou non. Parmi les facteurs connus on trouve l'emplacement de la station qui a détecté l'écho de météore et le moment où l'écho est détecté. Dans le cas de ce logiciel, ce sera le facteur temps qui sera utilisé.

Quand on compare plusieurs échos de météores, tous détectés sur des stations différentes, le facteur temps permet de donner une idée s'ils proviennent du même météore où non. En effet, par expérience on sait que lorsqu'un météore est détecté par plusieurs stations, toutes ces détections se font dans un intervalle d'environ trois secondes. Bien que ce facteur ne permet pas d'assurer complètement qu'un écho vient d'un météore spécifique, il permet d'éliminer toutes les stations où aucun écho a été détecté pour un météore, réduisant grandement le temps nécessaire pour les scientifiques du projet BRAMS. De plus, si dans l'intervalle recherché on ne trouve qu'un écho, ceci augmente fortement les chances que celui-ci appartient au météore recherché.

8.2 Fonctionnement du Logiciel

Le programme développé se lance en ligne de commande et nécessite un paramètre obligatoire : un temps précis à la seconde, autour duquel on va rechercher d'autres échos. Un deuxième paramètre optionnel est le code de location d'une station où on est sûr d'avoir détecté le météore recherché.

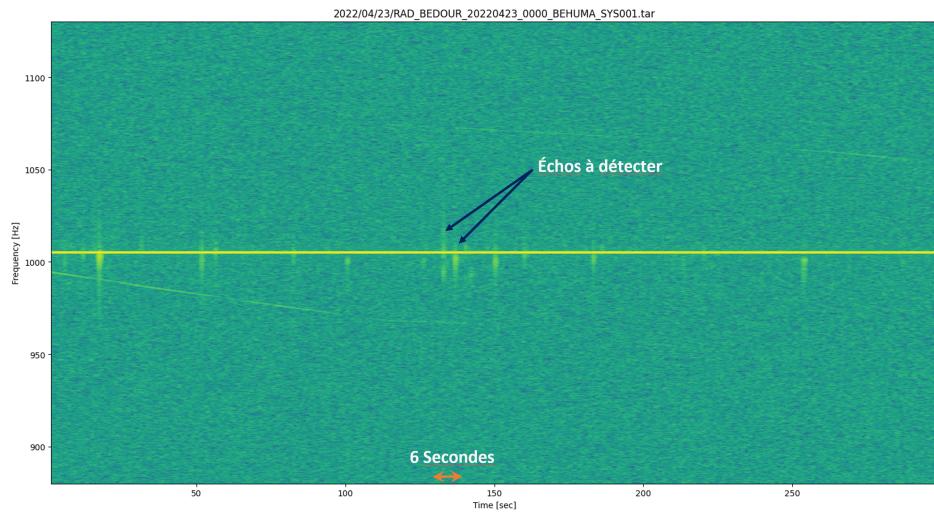


FIGURE 18 – Spectrogramme original de la station BEHUMA 1 le 23 avril 2022 à 00h00

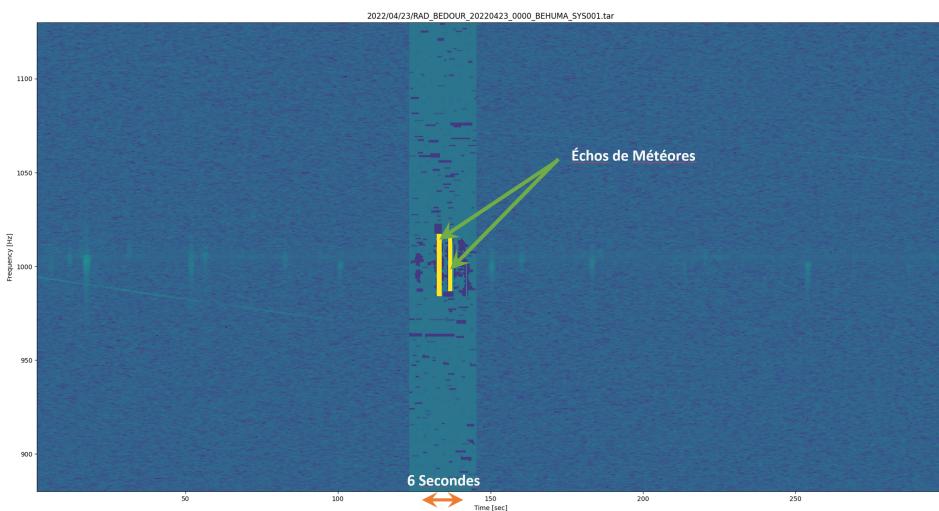


FIGURE 19 – Résultat du système de détection de météores.

Une fois que le programme est lancé, elle calcule l'intervalle. Si le temps donné en entrée est nommé t_0 et que Δt vaut trois secondes, l'intervalle calculé vaut $[t_0 - \Delta t; t_0 + \Delta t]$. Le programme va ensuite chercher tous les fichiers contenant cet intervalle. Afin de compléter cette action, elle utilise la classe BramsWavFile qui est également utilisé pour le programme de monitoring.

Pour chaque fichier récupéré, il calcule le spectrogramme et détecte les météores présents dans l'intervalle calculé. La détection des météores, en ce moment, est fait par un code développé avec le logiciel. Cependant, ce code est temporaire et est vouée à disparaître pour une solution plus efficace qui est en cours de développement par les membres du projet BRAMS et qui utilise le machine learning. Actuellement pour détecter les météores, différentes étapes sont exécutées par le code sur le spectrogramme. Ces étapes sont listées ci-dessous et leur application sera montrée sur base du spectrogram non changé affiché à la figure 18.

1. L'amplification des objets ressemblants à des météores par un filtre. Ce filtre a été développé spécialement pour ce but et peut être visualisé à l'annexe A. Le filtre est appliqué à l'aide d'une convolution. Une convolution est une opération mathématique qui prend deux fonctions en entrée et qui produit un nouveau signal. Dans ce cas si, les signaux en entrée sont le spectrogramme, ou une partie du spectrogramme, et le filtre. Un résultat de cette convolution sur un spectrogramme est représenté à l'annexe B.
2. La suppression du bruit en appliquant un filtre par percentile. Ceci implique que pour chaque colonne du spectrogramme dans l'intervalle calculée, le programme va mettre à zéro toutes les cellules dont la valeur est inférieure que la 95e percentile de cette même colonne. On peut observer l'effet de ce filtre à l'annexe C, où l'on voit bien que l'intervalle dans laquelle on recherche des échos de météores est bien plus propre qu'à l'annexe B.
3. L'élimination de tous les objets qui sont trop petits pour pouvoir être un écho d'un météore. Pour cette étape, une labellisation du spectrogramme est faite. Une labellisation, dans ce cas, est une action où l'on retrouve tous les objets séparés, non-nuls sur le spectrogramme. On vérifie alors leur taille et on la compare à la taille minimale qu'un écho de météore peut avoir. Si la taille de l'objet est inférieure à la taille minimale, elle est mise à zéro. Ceci peut être visualisé à l'annexe ?? où l'on peut voir, dans l'intervalle où on recherche des échos de météores, des tâches bleues. Ces tâches bleues sont des éléments se sont

fait éliminer par cette étape.

4. La suppression des échos d'avions. En effet, les météores ne sont pas les seuls objets qui traversent l'atmosphère et qui sont détectés par les stations réceptrices. Les échos d'avions sont des parasites dans les fichiers BRAMS et le programme ne peut pas les confondre avec des échos de météores. C'est pour cette raison que le logiciel vérifie, pour chaque objet restant sur le spectrogramme à cette étape, si cet objet n'est pas trop large pour être un météore.

Après toutes ces actions, les objets restants sur le spectrogramme sont considérées comme étant des échos de météores et le programme récupère leurs coordonnées. Un exemple de résultat final de la détection des météores est affiché à la figure 19. Notez que cette image n'est pas la sortie du programme et qu'elle ne s'affiche pas lors du déroulement du programme.

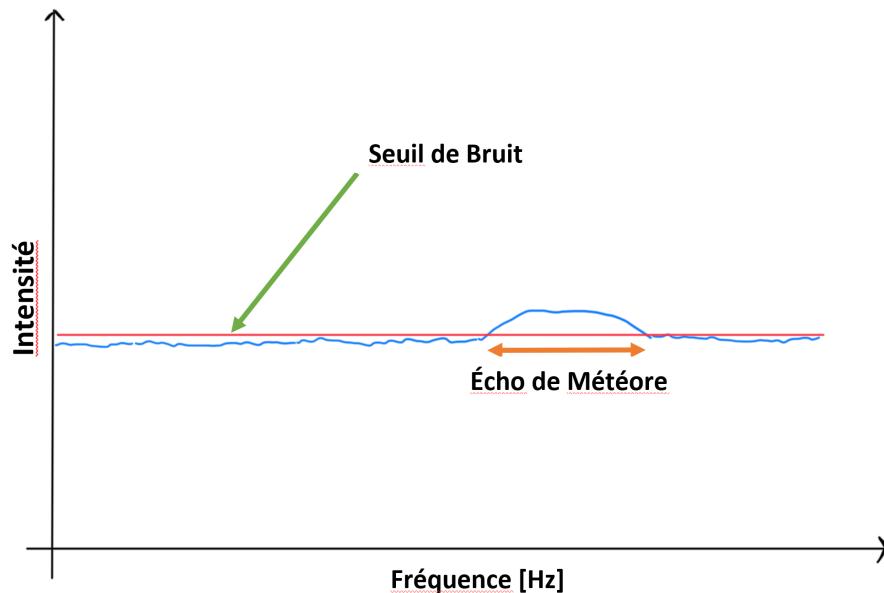


FIGURE 20 – FFT contenant un écho de météore.

Il ne reste plus que la recherche de la fréquence minimale et la fréquence maximale sur lequel s'étend l'écho. Ceci est fait en calculant la transformée de Fourier de l'intervalle pendant laquelle l'écho de météore dure. Une fois qu'on a la FFT, le logiciel détermine un seuil de bruit à la 85e percentile. Comme on peut le voir à la figure 20, ceci met en évidence l'écho de météore

et permet de retrouver facilement sa fréquence minimale et maximale.

```
1 location_code,antenna_id,file_start,meteor_count,meteor_time,fmin,fmax,distance_km
2 BEHUMA,1,202204230000,2,2022-04-23T00:02:13:801995,1000.2845764160156,1015.7615661621094,0.0 km
3 BEHUMA,1,202204230000,2,2022-04-23T00:02:17:585397,987.835693359375,1010.7147216796875,0.0 km
4 BEHUMA,3,202204230000,2,2022-04-23T00:02:17:585397,987.4992370605469,1009.0324401855469,0.0 km
5 BEHUMA,3,202204230000,2,2022-04-23T00:02:13:801995,987.1627807617188,1012.7334594726562,0.0 km
6 BEHUMA,4,202204230000,2,2022-04-23T00:02:17:585397,988.5086059570312,1010.0418090820312,0.0 km
7 BEHUMA,4,202204230000,2,2022-04-23T00:02:13:801995,989.8544311523438,1012.7334594726562,0.0 km
8 BEHUMA,5,202204230000,2,2022-04-23T00:02:13:801995,987.835693359375,1012.7334594726562,0.0 km
9 BEHUMA,5,202204230000,2,2022-04-23T00:02:17:241451,988.1721496582031,1011.7240905761719,0.0 km
10 BEHUMA,6,202204230000,2,2022-04-23T00:02:13:801995,990.52734375,1002.6397705078125,0.0 km
11 BEHUMA,6,202204230000,2,2022-04-23T00:02:17:585397,990.52734375,1009.368896484375,0.0 km
```

FIGURE 21 – Exemple de fichier CSV généré par le programme de détection de météores.

Enfin, le programme crée un fichier CSV qui contiendra une ligne par écho de météore détecté. Chaque ligne contiendra :

1. Le code de location de la station où est détecté l'écho.
2. Le numéro de l'antenne qui a détecté l'écho.
3. La date et le temps de début d'enregistrement du fichier d'où vient l'écho.
4. Le nombres d'écho comptés dans le même fichier que celui d'où vient l'écho.
5. Le temps, précis à la microseconde, de détection de l'écho.
6. La fréquence minimum de l'écho.
7. La fréquence maximum de l'écho.
8. La distance entre la station où est détecté l'écho et la station reçue en entrée au lancement du programme.

Un exemple de fichier CSV généré par le programme peut être visualisé à la figure 21.

9 Conclusion

Le but principal de ce projet de fin d'études était de faciliter et automatiser une partie du travail des scientifiques du projet BRAMS. Premièrement elle devait permettre aux scientifiques de détecter rapidement une anomalie dans les données produites par les stations réceptrices. Si un défaut n'est pas détecté rapidement, la station défectueuse continue à produire des données inutilisables. Ce problème a été résolu grâce au logiciel de monitoring qui permet non-seulement de détecter une anomalie dans les données, mais également de visualiser la qualité des données grâce à son interface intégré au site web BRAMS.

Deuxièmement, le travail devait faciliter la recherche de tous les échos de météores pouvant correspondre à un météore spécifique. Le programme de détection de météores permet aux scientifiques de grandement réduire le temps nécessaire pour chercher ces échos en cherchant tous les échos dans l'intervalle de temps nécessaire.

Ce travail m'a permis d'étendre mes connaissances dans le domaine du traitement du signal en mettant en pratique plusieurs notions théoriques vues durant les cours. Le mélange de ces connaissances avec des nouvelles notions, que je n'avais pas encore appris, afin d'arriver à un résultat final a été un vrai défi pour moi.

Enfin, ce projet m'a permis de contribuer à un projet scientifique de grande ampleur et m'a appris comment un tel projet fonctionne.

10 Perspectives

Suite à une réflexion sur les programmes réalisés, quelques pistes d'améliorations, intéressantes pour le futur, me sont venus en tête :

- Bien que cela ait déjà été dit, le remplacement du système de détection de météores, dans le programme qui retrouve les échos d'un météore, par le système plus avancé développé par le projet BRAMS représenterait une réelle amélioration.
- Pour le programme qui retrouve les échos d'un météore, on pourrait, à terme, rajouter le facteur d'emplacement de la station pour retrouver les échos appartenant à un météore. Cependant, cela demande beaucoup de recherche et d'étude et prendrait donc un temps considérable.
- On pourrait rajouter la possibilité de manipuler l'entièreté du programme de monitoring à partir de son interface en ligne, sur le site de BRAMS.
- Actuellement, le programme de monitoring surveille les fichiers de façon à détecter des erreurs qui ont déjà eu lieu. Cependant, il est possible que des nouveaux problèmes surviennent, qui ne sont pas détectés par le logiciel de surveillance. Une amélioration intéressante serait donc de réfléchir à des potentiels erreurs qui ne seraient pas détectés et en tenir compte dans le programme.
- Il serait intéressant d'avoir une interface graphique sur le site web de BRAMS permettant de manipuler le programme qui recherche les échos venant d'un météore. Cette interface pourrait alors afficher ces résultats sur un spectrogramme en plus de générer un fichier CSV.
- Pour le système de détection d'anomalies dans le programme de monitoring, il serait intéressant d'effectuer plus de tests et de recherches afin d'améliorer la méthode utilisée. En effet, comme expliqué avant, il n'est pas parfait et alerte parfois l'utilisateur d'un problème alors qu'il n'y en a pas.

Références

- [1] *Site internet du projet BRAMS*, consulté en janvier 2022
<https://brams.aeronomie.be/>
- [2] *Documentation de la fonction scipy.signal.spectrogram*, consulté en février 2022
<https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.spectrogram.html>
- [3] *Hands-On Tutorial on Visualizing Spectrograms in Python*, consulté en février 2022
<https://analyticsindiamag.com/hands-on-tutorial-on-visualizing-spectrograms-in-python/>
- [4] *Cutting unused frequencies with specgram, matplotlib*, consulté en février 2022
<https://stackoverflow.com/questions/19468923/cutting-of-unused-frequencies-in-specgram-matplotlib>
- [5] *Documentation Matplotlib*, consulté en février 2022
https://matplotlib.org/stable/api/mlab_api.html#matplotlib.mlab.specgram
- [6] *Set spectrogram Parameters*, consulté en février 2022
<https://stackoverflow.com/questions/29321696/what-is-a-spectrogram-and-how-do-i-set-its-parameters>
- [7] *Documentation de la fonction numpy.convolve*, consulté en mars 2022
<https://numpy.org/doc/stable/reference/generated/numpy.convolve.html>
- [8] *How to convolve two 2-dimensional matrices in python with scipy*, consulté en mars 2022
<https://moonbooks.org/Articles/How-to-do-a-simple-2D-convolution-between-a-kernel-and-an-image-in-python-with-scipy-/>
- [9] *Slice 2D array in smaller 2D arrays*, consulté en mars 2022
<https://stackoverflow.com/questions/16856788/slice-2d-array-into-smaller-2d-arrays>
- [10] *Compute a confidence interval from sample data*, consulté en mars 2022
<https://stackoverflow.com/questions/15033511/compute-a-confidence-interval-from-sample-data>

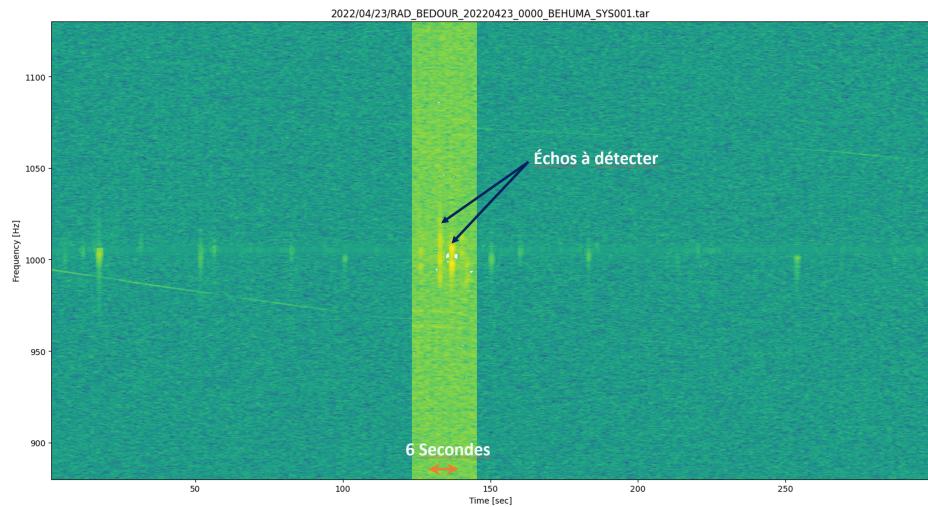
- [11] *Microsoft WAVE soundfile format*, consulté en mars 2022
<http://soundfile.sapp.org/doc/WaveFormat/>
- [12] *Python - Sending Email using SMTP*, consulté en mars 2022
https://www.tutorialspoint.com/python/python_sending_email.htm
- [13] *The fastest 2D convolution in the world*, consulté en mars 2022
<https://laurentperrinet.github.io/sciblog/posts/2017-09-20-the-fastest-2d-convolution-in-the-world.html>
- [14] *Documentation de la fonction scipy.ndimage.convolve*, consulté en mars 2022
<https://docs.scipy.org/doc/scipy/reference/generated/scipy.ndimage.convolve.html>
- [15] *Power Spectral Density - an overview*, consulté en avril 2022
<https://www.sciencedirect.com/topics/computer-science/power-spectral-density>
- [16] *What is a Power Spectral Density*, consulté en avril 2022
<https://community.sw.siemens.com/s/article/what-is-a-power-spectral-density-psd>
- [17] *How to add time onto a datetime object in Python*, consulté en avril 2022
<https://www.adamsmith.haus/python/answers/how-to-add-time-onto-a-datetime-object-in-python>
- [18] *Fourier Transforms With scipy.fft : Python Signal Processing*, consulté en avril 2022
<https://realpython.com/python-scipy-fft/#why-would-you-need-the-fourier-transform>
- [19] *Documentation de la fonction scipy.fft.rfft*, consulté en avril 2022
<https://docs.scipy.org/doc/scipy/reference/generated/scipy.fft.rfft.html>
- [20] *Reading and Writing CSV Files in Python - Real Python*, consulté en mai 2022
<https://realpython.com/python-csv/>
- [21] *A Fast, Extensible Progress Bar for Python and CLI - TQDM*, consulté en mai 2022
<https://github.com/tqdm/tqdm>
- [22] *Documentation du module tarfile*, consulté en mai 2022
<https://docs.python.org/3/library/tarfile.html>

- [23] *Documentation de la librairie simplejson*, consulté en mai 2022
<https://pypi.org/project/simplejson/>
- [24] *Python Command Line Arguments*, consulté en mai 2022
<https://realpython.com/python-command-line-arguments/>
- [25] *Documentation du module argparse*, consulté en mai 2022
<https://docs.python.org/3/library/argparse.html>
- [26] *Artificial neural network for bubbles pattern recognition on the images - Scientific Figure on ResearchGate*, consulté en mai 2022
https://www.researchgate.net/figure/a-Illustration-of-the-operation-principle-of-the-convolution-kernel-convolutional-layer_fig2_309487032

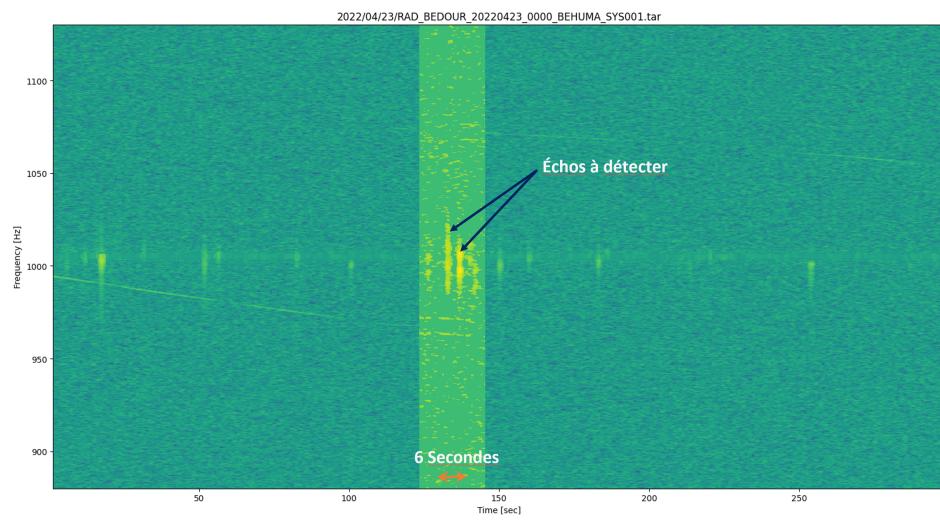
A Kernel de Convolution Permettant d'Amplifier les Signaux de Météores

```
1      [[ 0.   0.   0.   50.   0.   0.   0.   ]  
2      [ 0.   0.   0.   50.   0.   0.   0.   ]  
3      [ 0.   0.   0.   0.   0.   0.   0.   ]  
4      [ 0.   0.   0.   0.   0.   0.   0.   ]  
5      [ 0.   0.   0.   0.   0.   0.   0.   ]  
6      [ 0.   0.   0.   0.   0.   0.   0.   ]  
7      [ 0.   0.   0.   0.   0.   0.   0.   ]  
8      [ 0.   0.   0.   0.   0.   0.   0.   ]  
9      [ 0.   0.   0.   0.   0.   0.   0.   ]  
10     [ 0.   0.   0.   0.   0.   0.   0.   ]  
11     [ 0.   0.   0.   0.   0.   0.   0.   ]  
12     [ 0.   0.   0.   0.   0.   0.   0.   ]  
13     [-1.5  0.   0.   0.   0.   0.   -1.5]  
14     [-1.5  0.   0.   0.   0.   0.   -1.5]  
15     [-1.5  0.   0.   0.   0.   0.   -1.5]  
16     [ 0.   0.   0.   0.   0.   0.   0.   ]  
17     [ 0.   0.   0.   0.   0.   0.   0.   ]  
18     [ 0.   0.   0.   0.   0.   0.   0.   ]  
19     [ 0.   0.   0.   0.   0.   0.   0.   ]  
20     [ 0.   0.   0.   0.   0.   0.   0.   ]  
21     [ 0.   0.   0.   0.   0.   0.   0.   ]  
22     [ 0.   0.   0.   0.   0.   0.   0.   ]  
23     [ 0.   0.   0.   0.   0.   0.   0.   ]  
24     [ 0.   0.   0.   0.   0.   0.   0.   ]  
25     [ 0.   0.   0.   0.   0.   0.   0.   ]  
26     [ 0.   0.   0.   50.   0.   0.   0.   ]  
27     [ 0.   0.   0.   50.   0.   0.   0.   ]]
```

B Résultat du filtre pour amplifier les échos de météores



C Résultat du filtre à percentile



D Résultat des éléments qui sont trop petits pour pouvoir être un écho

