Informatique

Faculté des sciences

Université de Sherbrooke

Travail Pratique n°1 : **La transformée de Fourier locale**

par

Nicolas Auclair-Labbé

15 068 524

et

Félix Lussier

19 132 760

Remis à

Marie-Flavie Auclair-Fortier

Dans le cadre de l’activité pédagogique

IMN318 : Traitement de l’audionumérique

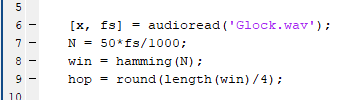
24 octobre 2020

**No.1 - Glock.wav**

Dans ce numéro nous devions trouver les notes jouées par un glockenspiel sur un extrait audio. On peut réduire les tâches à effectuer au nombre de trois :

1. Préparer l’information pour calculer la transformée de Fourier du signal et pour avoir un filtre pour le fenêtrage.
2. Calculer le spectrogramme de l’extrait audio à l’aide des méthodes établies en 1.
3. Modifier les informations du spectrogramme pour y retrouver plus facilement la fréquence des notes jouées par le glockenspiel.
4. **Transformée de Fourier et filtre**

Pour cette étape nous avons pas mal suivit l’énoncé du TP. On a mis en le signal et la fréquence d’échantillonnage de l’extrait audio en variable. Ensuite on se sert de cette fréquence d’échantillonnage et d’un scalaire pour créer une fenêtre de Hamming (*win*) qui servira au fenêtrage. On a aussi créé une variable (*hop*) qui contient le nombre d’échantillon à sauter par mouvement de fenêtre. À partir de là on avait toutes les informations nécessaires pour effectuer les transformées de Fourier pour le spectrogramme.



1. Les variables de bases

1. **Le spectrogramme**

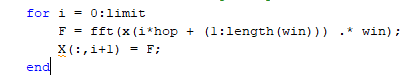
Pour avoir le spectrogramme de l’extrait on a dû ajouter plusieurs tranches de TF pour obtenir un spectre temporel. On a donc dû commencer par déterminer le nombre de tranche à calculer.



2. Formule pour la borne de la boucle

le *-length(win)* et *-1* est pour être sûr qu’on ne va pas accidentellement dépasser la taille du vecteur de signal *x*.

À chaque itération de la boucle on calcul la TF d’une section plus avancée dans l’extrait audio et on l’ajoute à notre matrice de spectrogramme *X*. Ne pas pouvoir ajouter en paramètre la fréquence d’échantillonnage nous force à décimer notre signal. Nous avons fait une bonne partie de ce numéro avec 44100 comme deuxième paramètre de la fonction *fft()*. C’est seulement plus tard qu’on avait réalisé qu’il faut vivre avec la décimation pour ce TP.

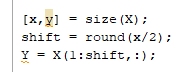


3. Boucle qui permet d’avoir un spectrogramme de l’extrait

1. **Modification du spectrogramme pour trouver les notes**

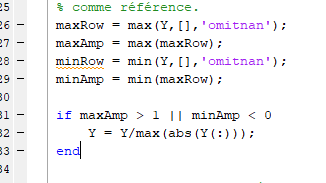
Le principal problème du spectrogramme de 2. est la présentation de l’information qui n’est pas adéquate pour déterminer quelles notes sont jouées dans l’extrait audio. Pour ça on doit appliquer quatre modifications au spectrogramme :

1. Retirer la fréquence de Nyquist.
2. Convertir l’intensité en décibel négatif.
3. Restreindre le signal selon un ambitus fréquentiel et d’intensité donné.
4. Modifier les valeurs de l’axe temporel et de l’axe fréquentiel.
   1. Pour cela on a dû soustraire la 2e moitié de l’ambitus de fréquence. On a donc créé une nouvelle matrice dans laquelle on a passé les variables de notre spectrogramme de base jusqu’à la moitié de la fréquence maximale. Ça nous a permis de nous débarrasser de la symétrie de fréquence qui ne nous est pas utile.



4. La nouvelle matrice ne contient que la première moitié des fréquences

* 1. Ici on s’assure que l’intensité a bien été normalisé entre 0 et 1 (ce qui devrait être le cas comme le spectrogramme passé en paramètre dans *DispSTFT* est déjà normalisé), si ce n’est pas le cas on le normalise.



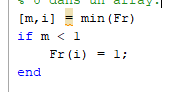
5. Vérifie si l'intensité est normalisée entre 0 et 1

Ensuite, on calcule le *log10* de la matrice. Nous en avons profité aussi pour multiplier le résultat par deux pour compenser la fréquence de Nyquist qui a été supprimé à l’étape 3.1.



6. Calcul de décibel et multiplication pour compenser la moitié symétrique supprimée

* 1. Dans cette étape on a restreint l’ambitus fréquentiel (Hz) et d’intensité (dB) selon les variables passées en paramètre dans la fonction. Pour l’ambitus fréquentiel, on vérifie si la variable minimale passée est supérieure à zéro pour permettre de l’utiliser comme index pour un array plus tard (comme il n’y a pas de position 0 dans les structures de données dans MatLab).

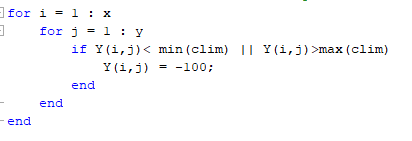


7. Vérifie si on peut respecter le domaine de fréquence demandé et le modifie si nécessaire.



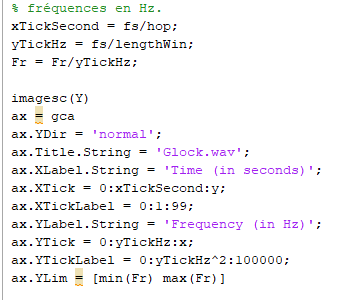
8. Dessine le graphique en respectant les limites demandées

Pour l’ambitus d’intensité, on passe l’ensemble du spectrogramme dans une boucle. À chaque point on vérifie si leur intensité fait partie du domaine d’intensité demandé. Les points dont l’ambitus ne passe pas la condition se font assigner une intensité minime (-100dB). Cela permet de ne garder que les intensités plus fortes pour mieux faire ressortir les notes de glockenspiel.



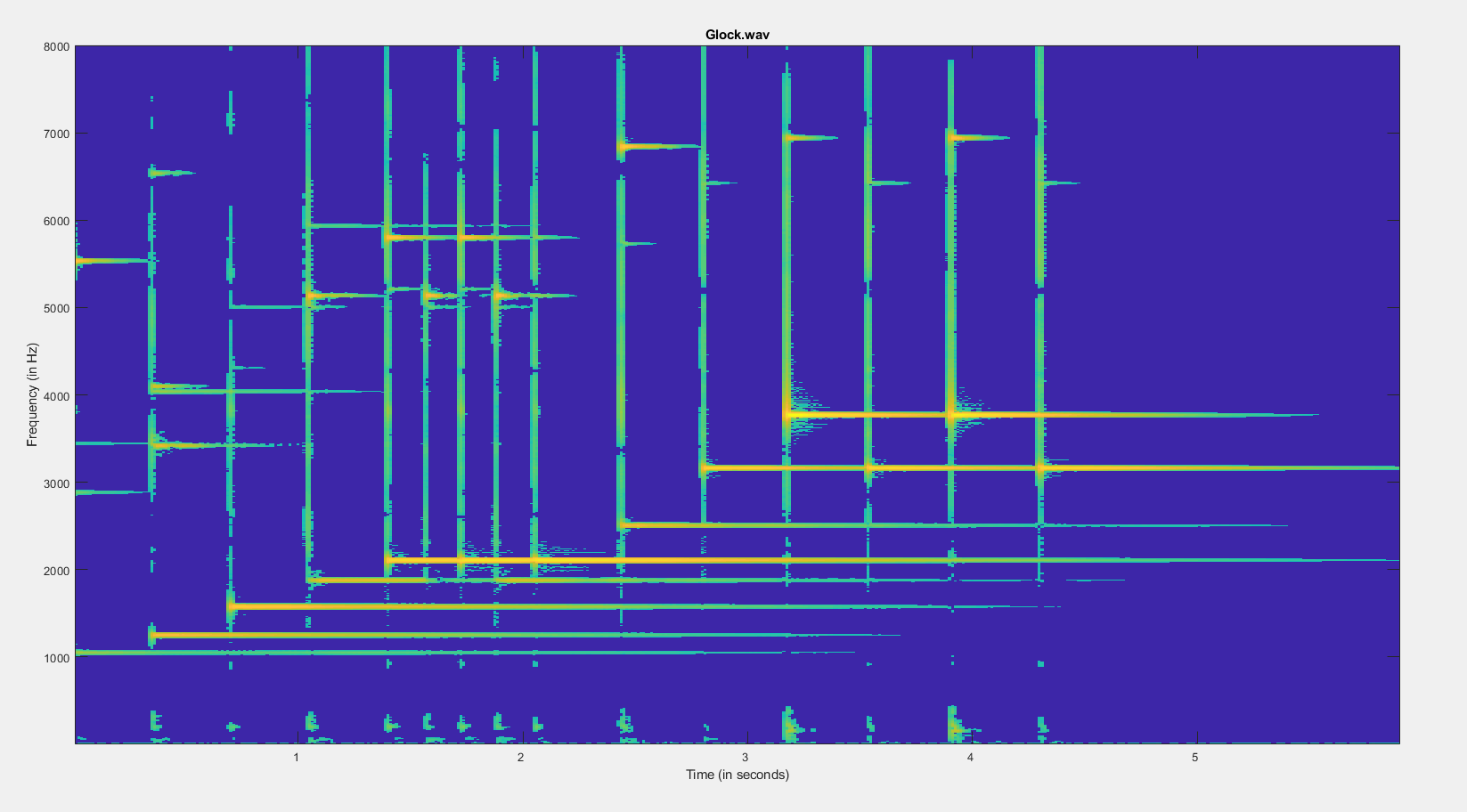
9. Boucle qui vérifie l’intensité des points et qui les réduits s’ils ne font pas partie du domaine demandé.

* 1. Dans cette étape finale, nous avons simplement donné quelques commandes de formatage pour mieux présenter l’information du spectrogramme final. Nous avons notamment changé ce qui détermine une unité pour l’axe des x pour obtenir des secondes au lieux des fenêtres d’échantillons. On a fait ça simplement en désignant qu’une unité en x est égale au nombre de fenêtre par seconde. Pour convertir les fréquences « bins » (basé sur la taille de la fenêtre Hamming) à des fréquences en Hz on a fait la même chose, sauf que cette fois-ci la fréquence est égale à la fréquence d’échantillonnage sur le nombre d’échantillon utilisé pour faire la transformée de Fourier (ce nombre étant égal à la largeur de la fenêtre de Hamming).

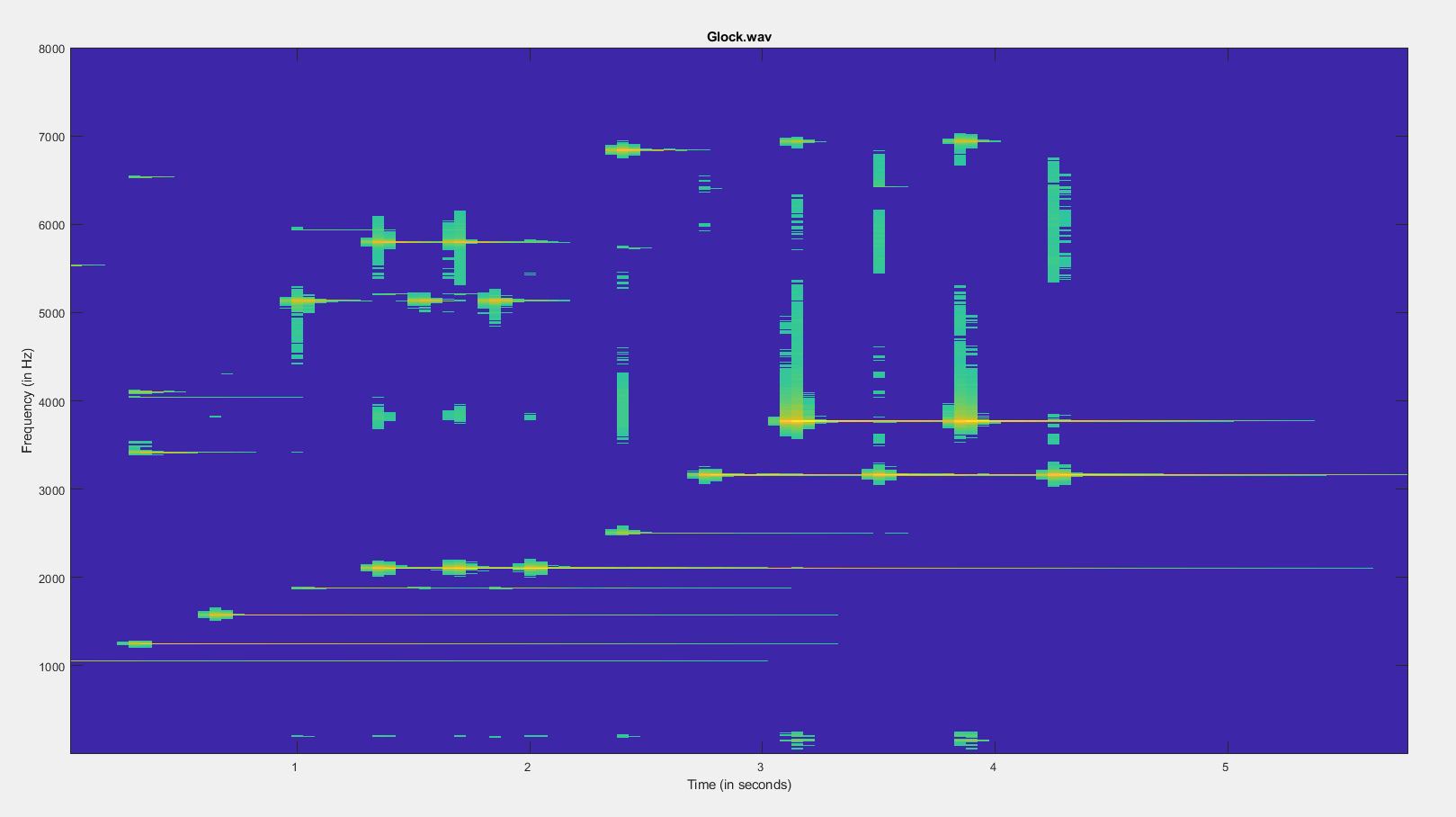


10. Les commandes de formatage pour le spectrogramme ainsi que le calcul de ratio pour les secondes.

Tout ça nous permet finalement d’obtenir ce spectrogramme pour l’extrait audio.

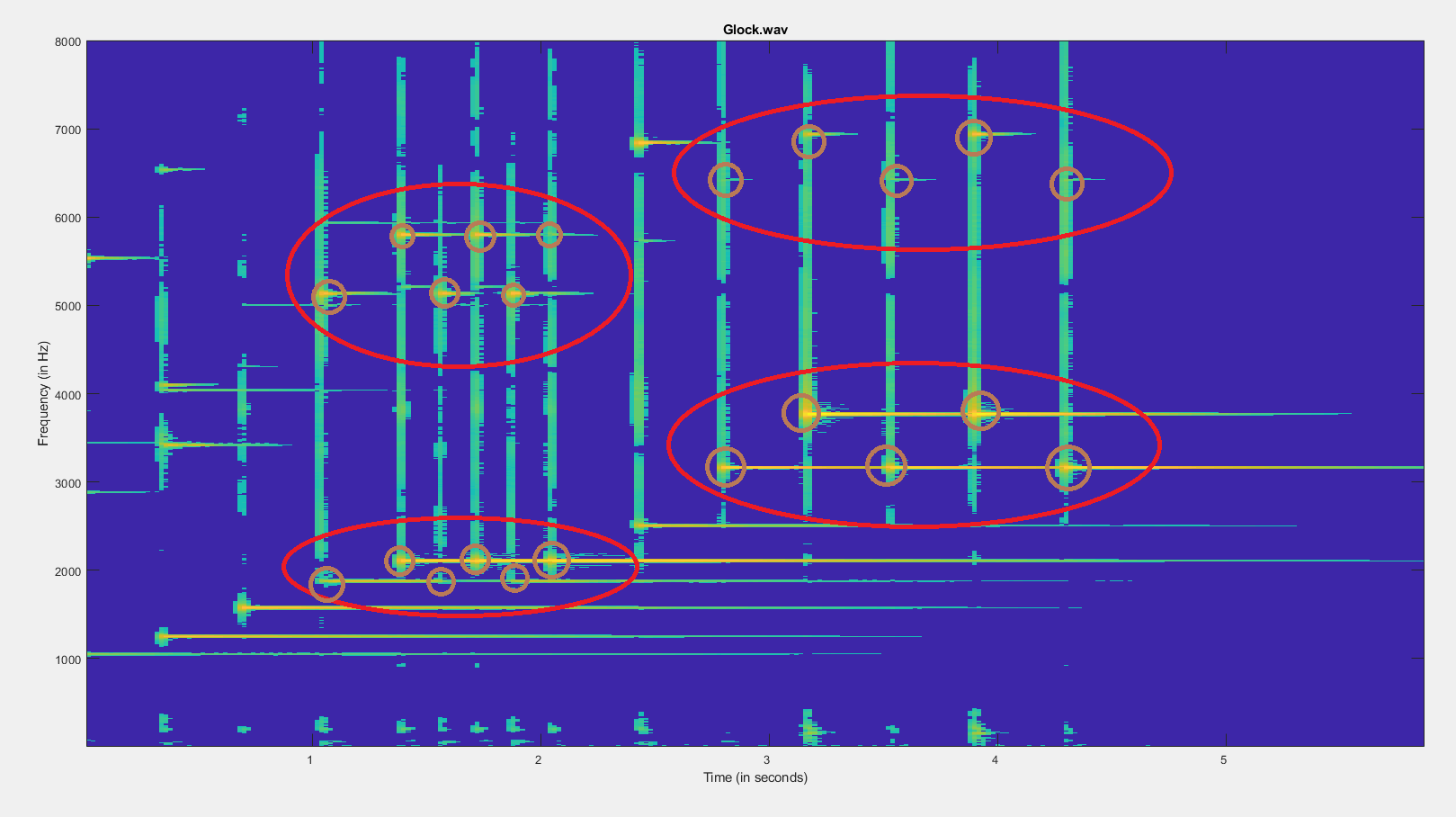


11. Spectrogramme avec une limite de -50dB et une fenêtre de 50ms



12. Spectrogramme utilisé pour trouver les notes. Limite à -45dB et fenêtre de 200ms

Un passage qui a été un peu plus difficile à trouver est celui avec les double-croches entre la première et deuxième seconde. Nous avons réussi à déterminer qu’il s’agit d’une broderie autour des mêmes deux notes à l’aide du motif similaire entre cette partie et celle entre la troisième et la quatrième seconde. Dans celle-là, le mouvement rythmique est plus lent, mais le mouvement mélodique est similaire. Nous avons donc remarqué la corrélation entre ces deux parties à l’aide de leur harmonique qui suivent le même mouvement mélodique.



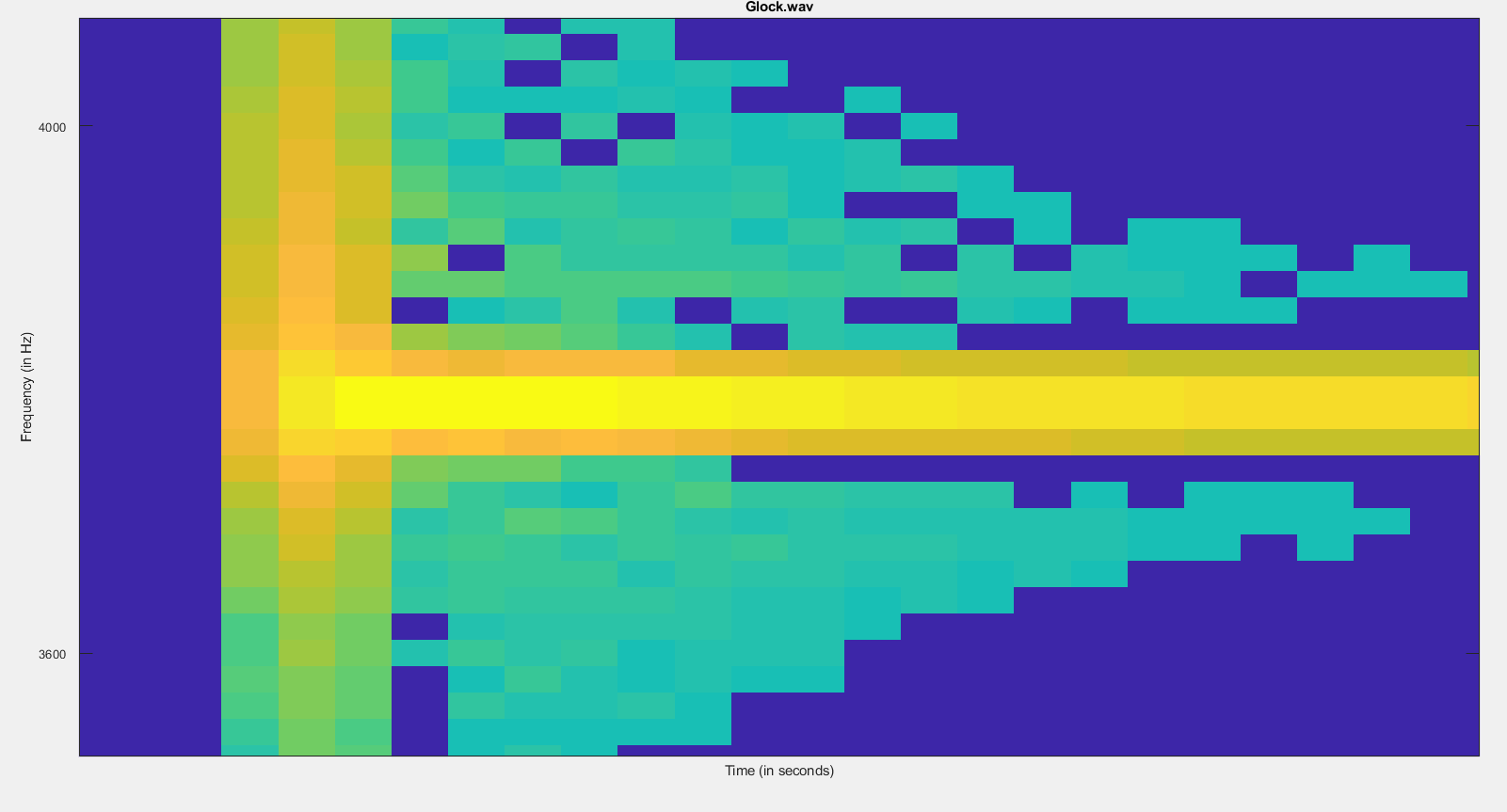
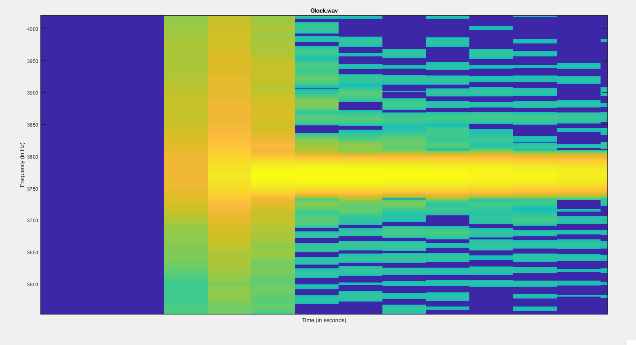
13. Fenêtre de 80ms et limite à -50dB

Si on agrandit le spectrogramme de la figure 12, on peut trouver les notes suivantes (présentés dans l’ordre) :

|  |  |
| --- | --- |
| **≈ 1053**  **=**  **C6** |  |
| **≈ 1250Hz**  **=**  **Eb6** |  |
| **≈ 1580Hz**  **=**  **G6** |  |
| **≈ 1880Hz**  **=**  **Bb6** |  |
| **≈2108Hz**  **=**  **C7** |  |
| **Les deux notes identifiées sont le Bb6 et le C7 qu’on vient de voir. On peut facilement remarquer que la broderie est sur les mêmes deux notes.** |  |
| **≈2505Hz**  **=**  **Eb7** |  |
| **≈3166Hz**  **=**  **G7** |  |
| **≈3771Hz**  **=**  **Bb7** |  |
| **Même chose que pour plus haut, une broderie avec les mêmes deux notes.** |  |

Bref, on peut voir que l’extrait de glockenspiel nous joue une montée de Cm7 sur deux octaves. Il y a aussi un rythme assez constant, sauf pour les notes cinq, six, sept et huit qui sont deux fois plus rapide que les autres. Le rythme est assez stable et comme on peut voir qu’il y a un tout petit peu moins qu’un temps et demi dans une seconde, le tempo doit être entre 80 et 85.

Une petite note, lorsqu’on a travaillé sur ce numéro notre premier réflexe a été de manuellement mettre la fréquence d’échantillonnage comme paramètre de la *fft()*. Par exemple, avec une fenêtre de 50ms, on coupe une fréquence sur vingt. On peut voir le résultat plus clairement ici.

   
*Les deux images ont été calculé avec une fenêtre de 50ms. À gauche, on a seulement conservé 2205 variables fréquentielles par TF alors qu’à droite on a conservé l’ensemble des 44100 variables.*

**No.2 - Restore.wav**