TP Traitements Numériques Avancés

Arthur GARON

Hamza ARIOUICH

Sample-Rate Converter (SRC)

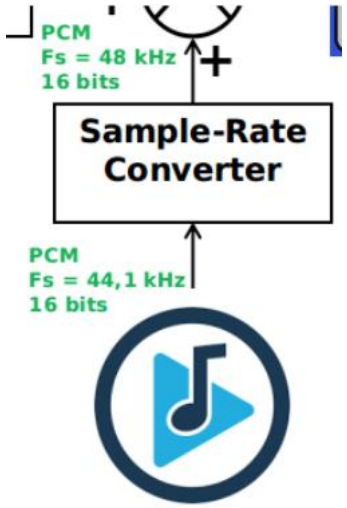


Fig1 : SRC

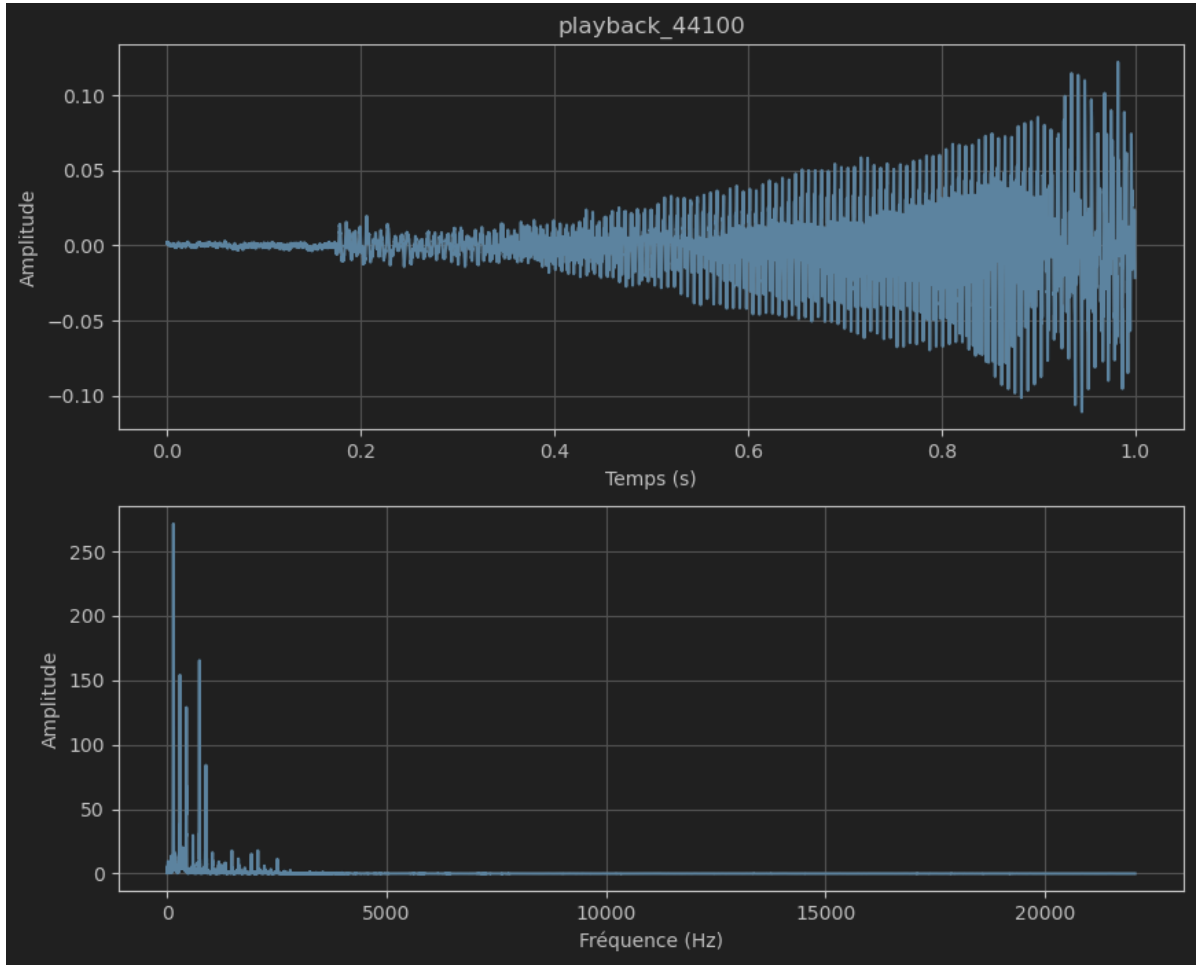
- ***Rappel de l'objectif de cette partie :*** concevoir un convertisseur de taux d'échantillonnage (SRC) pour passer de 44,1 kHz à 48 kHz.
- ***Contexte :*** importance de l'interpolation et de la décimation dans le traitement du signal audio.
- ***Défis principaux :*** qualité audio, minimisation des erreurs et optimisation de la puissance de calcul.

Le sample-Rate Converter se constitue de trois étapes majeures :

1. **Augmentation de la fréquence d'échantillonnage**
2. **Réduction de la fréquence d'échantillonnage**
3. **Filtrage anti-repliement**

Sample-Rate Converter (SRC)

Signal d'entrée



Le signal d'entrée est un signal discret encodé sur 16 bits. À l'analyse temporelle, on observe que le début du signal est dominé par du bruit, puis le son devient progressivement audible.
À l'aide de la FFT, on constate que le spectre est principalement concentré dans les basses fréquences, où les raies spectrales sont plus marquées. En revanche, dans les hautes fréquences, l'amplitude des raies est très faible, voire nulle.

Mise en place d'une version naïve

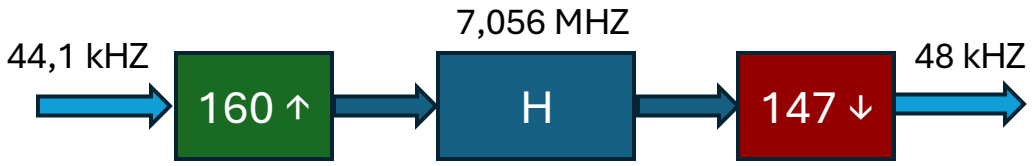
Pour convertir un signal d’une fréquence d’échantillonnage fs_in=44100 Hz à une fréquence de sortie fs_out=48000 , voici les étapes détaillées :

1. Calcul des facteurs L et M:

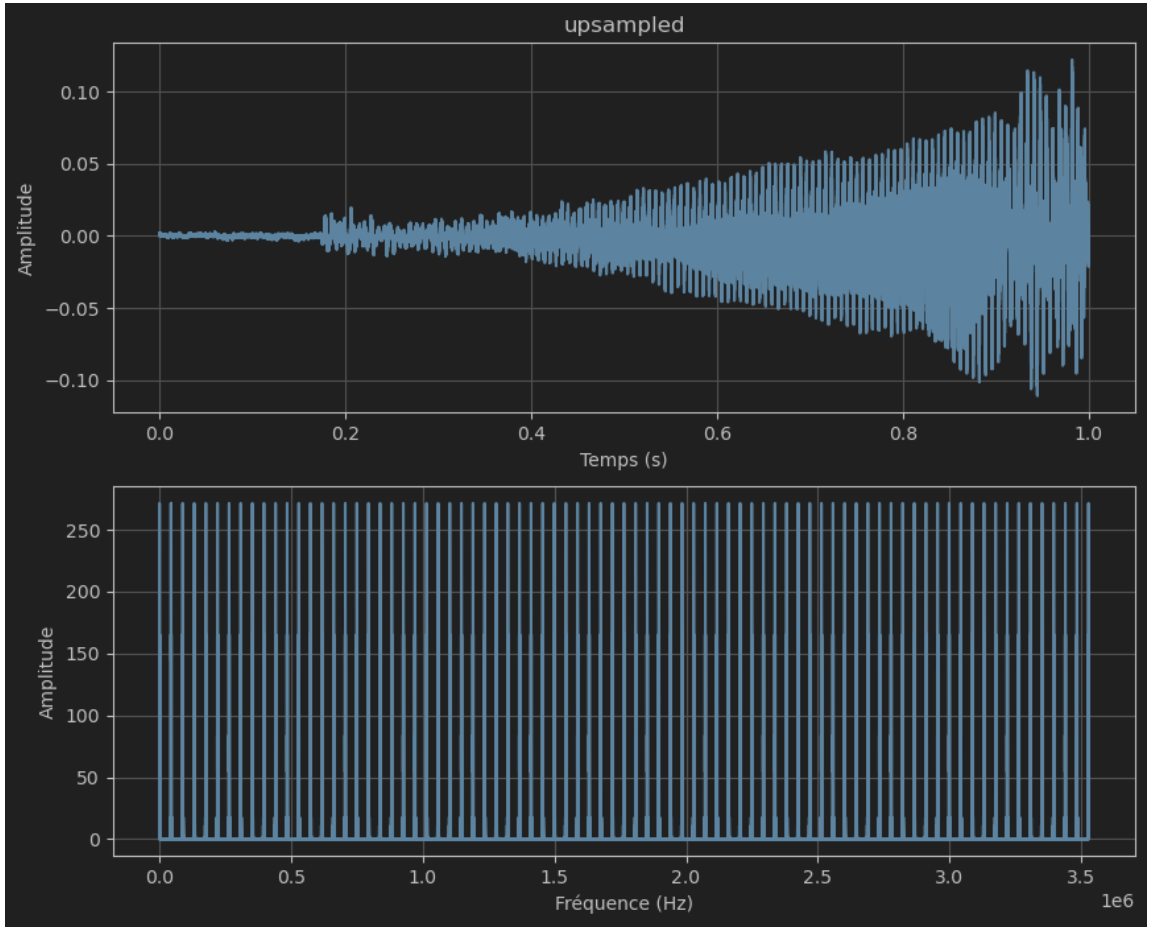
- Facteur commun : factor = pgcd(fs_in, fs_out) =300
- Calcul des coefficients:
  - L = fs_out/factor = 160
  - M = fs_in/factor = 147

2. Étapes de filtrage

- Upsampling (L)
- Filtrage passe-bas
- Downsampling (M)



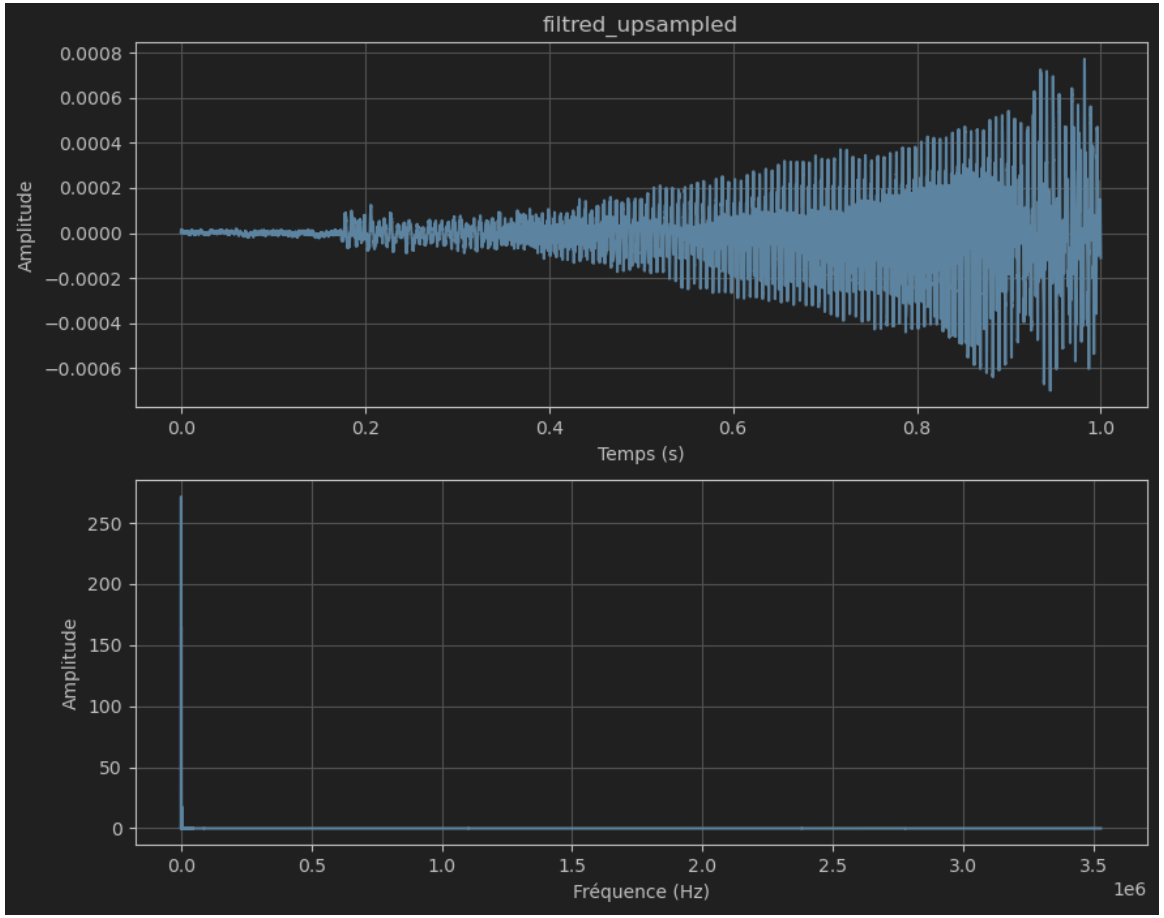
Upsampling



L'upsampling consiste à insérer L−1 zéros entre chaque échantillon du signal d'entrée. Cela crée des copies spectrales du signal autour des multiples de la nouvelle fréquence d'échantillonnage (L·44100).

Dans notre cas, l'upsampling avec L=160 a généré L−1=159 répliques spectrales qu'il a fallu supprimer efficacement avec le filtrage.

Filter upsampled

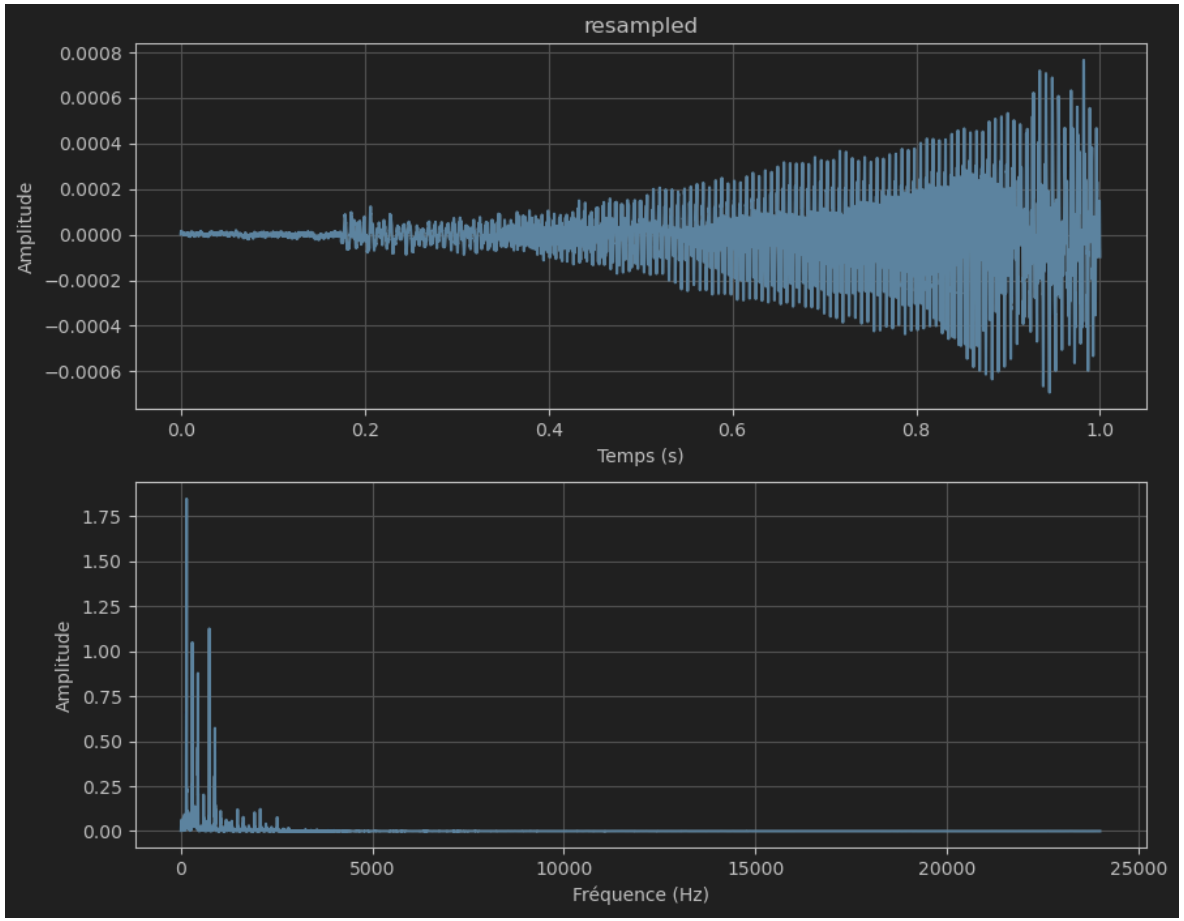


Un filtre passe-bas est appliqué pour supprimer les répliques spectrales introduites par l'upsampling.

- **Type de filtre** : FIR (Finite Impulse Response).
- **Ordre** : 101 dans un premier temps.
- **Fréquence de coupure** : 0.5/max(L,M), soit environ 1/2L, pour garantir une suppression correcte des répliques spectrales.

Nous travaillons ici a une fréquence de 7,056 MHz, ce qui est très couteux en termes de calcules

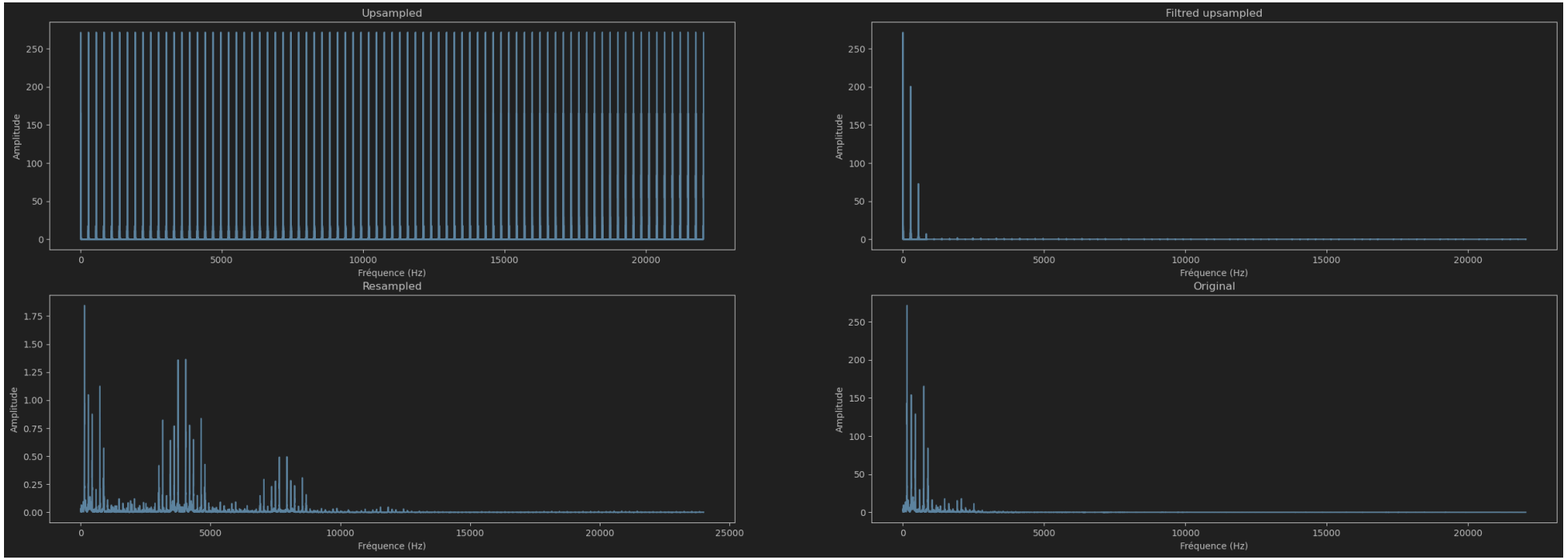
Downsampling



Le downsampling consiste à réduire la fréquence d’échantillonnage en gardant un échantillon sur M. Cela diminue la fréquence finale à 48000 Hz.

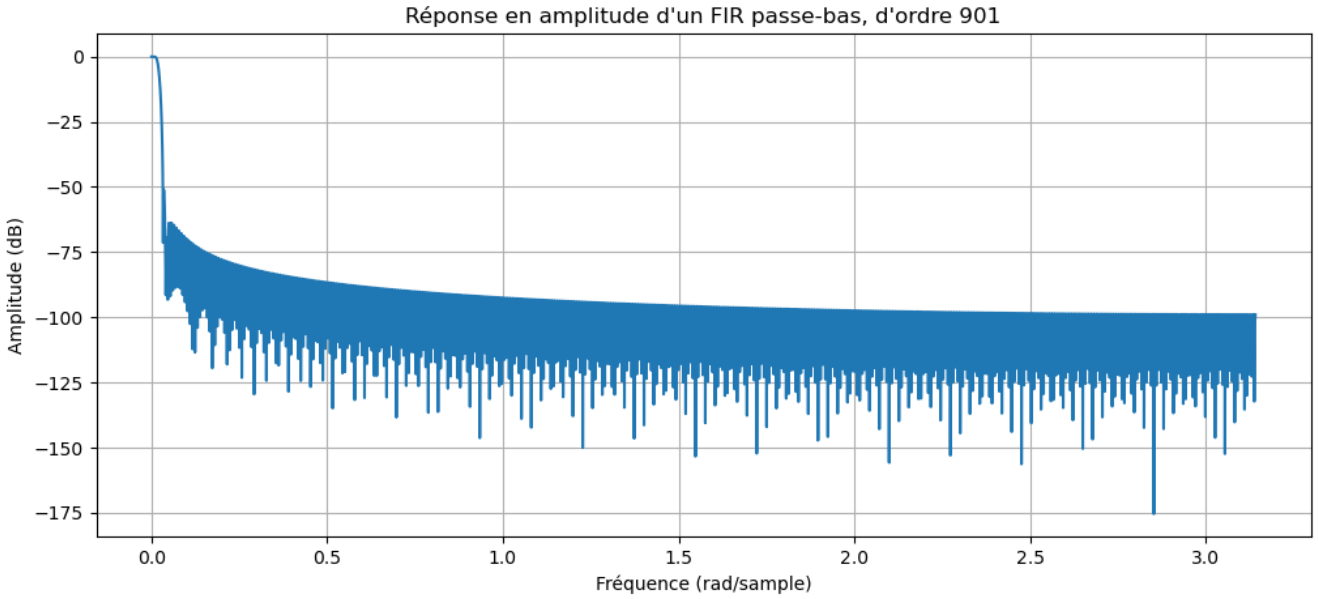
On retrouve alors notre signal original à la fréquence d'échantillonnage souhaitée

Ordre du filtre pas assez élevé



Dans un premier temps, nous avons utilisé un filtre d'ordre insuffisamment élevé, ce qui n'a pas permis de supprimer toutes les répliques spectrales, comme nous pouvons le voir sur le résultat final.

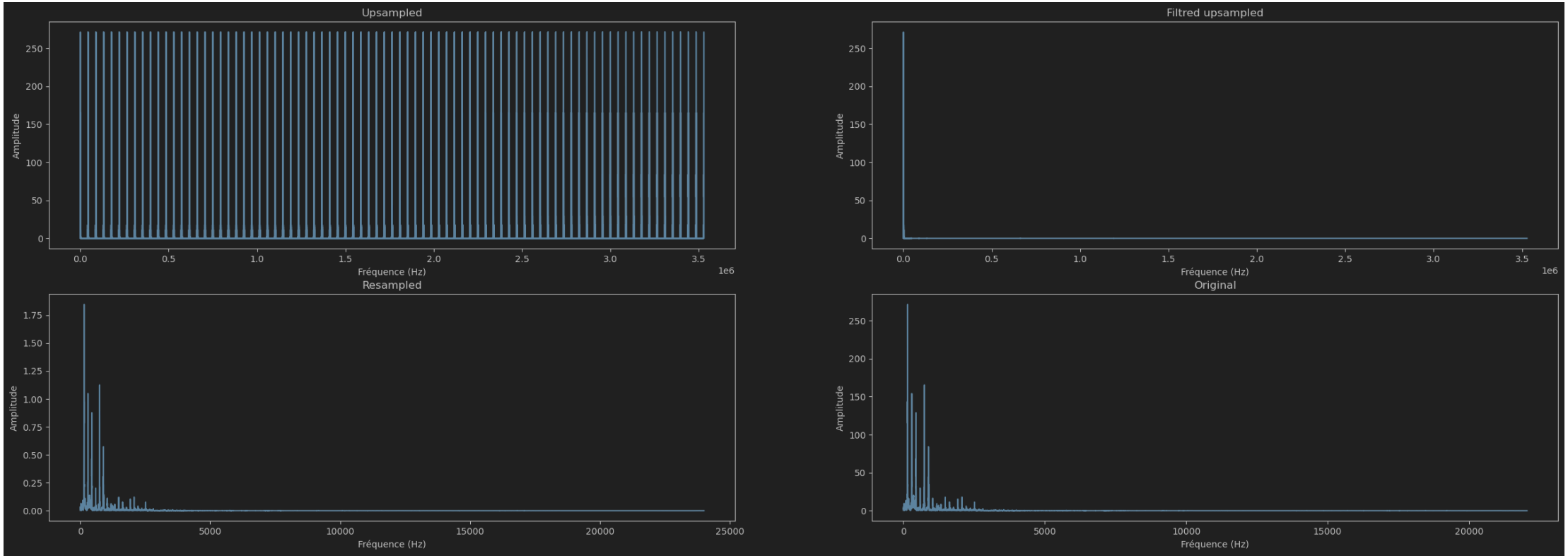
Design du filtre



Cela était dû à l'ordre du filtre qui était trop bas, nous avons donc choisit un nouveau filtre de la manière suivantes:

- La contrainte est d'assurer une dynamique de 96 dB correspondant à une précision de 16 bits.
- Nous avons donc choisi un filtre FIR d'ordre 901 pour assurer –96 dB dans la bande filtrée.
- Cependant, cet ordre est très élevé, ce qui entraîne une complexité de calcul importante.

Ordre correct



Le signal est correctement resampler

Mise en place d'une version améliorer

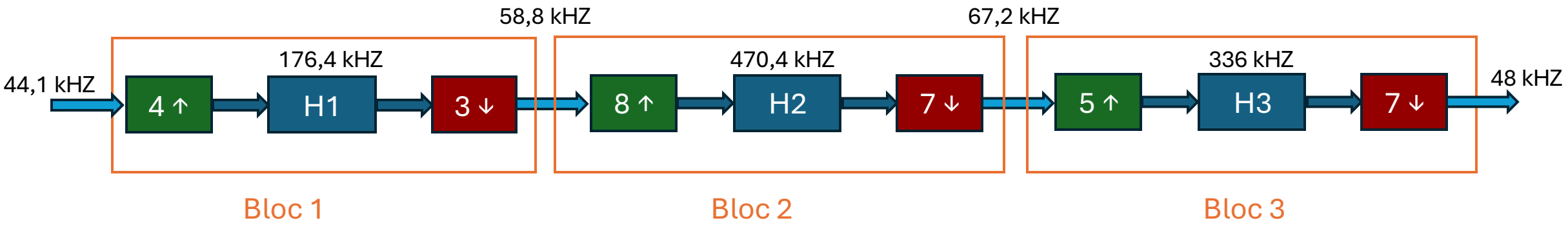
Comme nous l’avons vu précédemment, l’ordre très élevé du filtre rend les calculs complexes, et l’upsampling occupe beaucoup d’espace en mémoire avec une liste de taille :

- L× taille de la séquence d’entrée.

Pour optimiser ce processus, nous allons effectuer le rééchantillonnage en cascade.
En décomposant LLL et MMM en facteurs premiers :

- L=160=2^5 × 5=4×8×5
- M=147=3×7^2=3×7×7

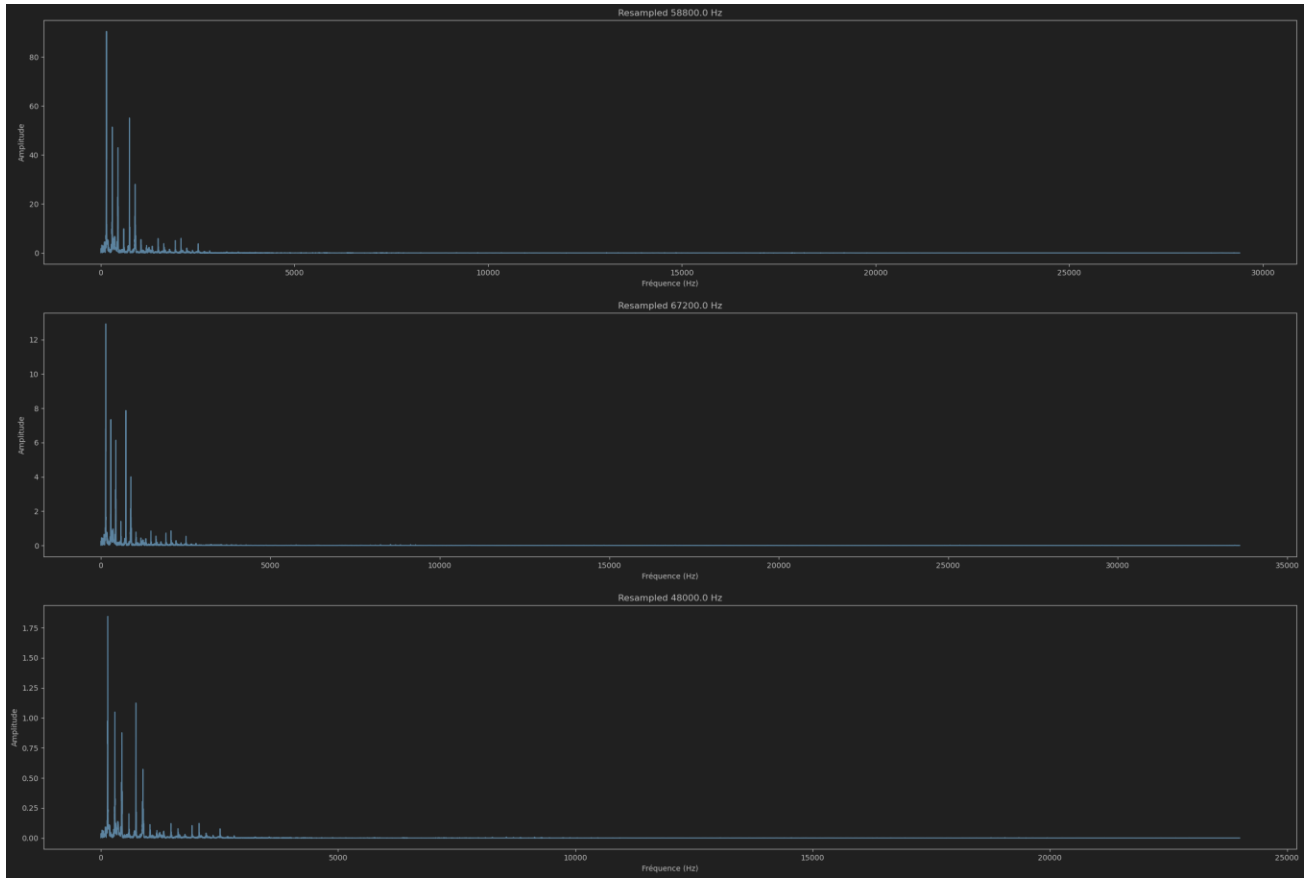
Nous pouvons alors regrouper le rééchantillonnage en plusieurs blocs de transformations successives. Cela permet de réduire la complexité des calculs et la mémoire utilisée à chaque étape.



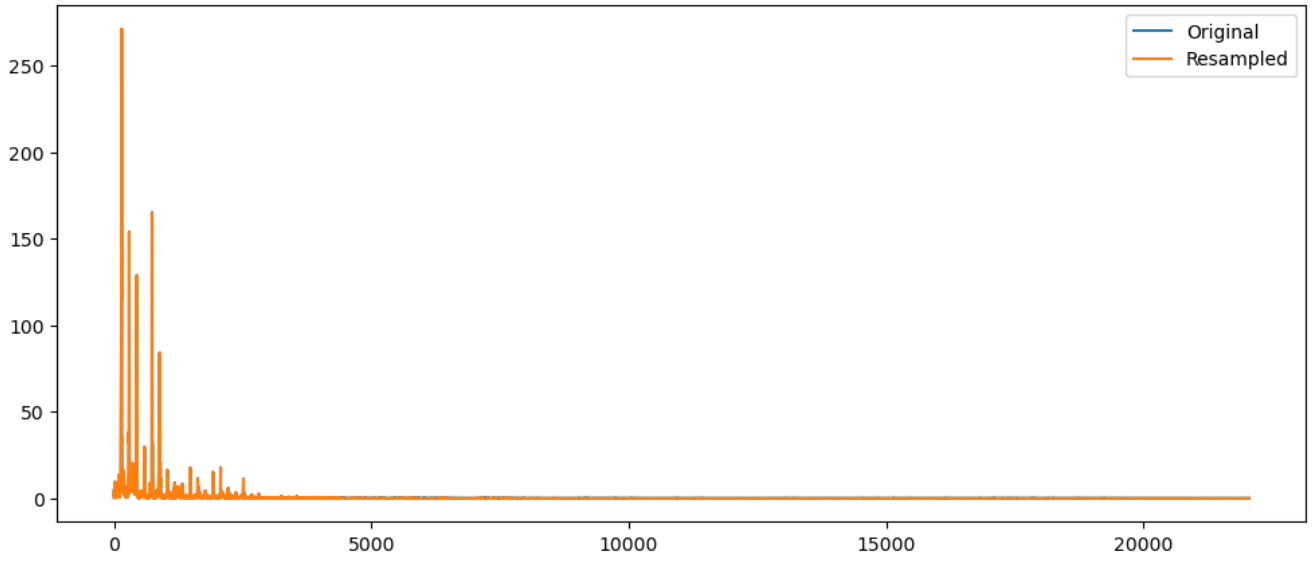
Mise en place d'une version améliorer

En plus de gagner de l’espace mémoire, cette méthode permet aux filtres de travailler à des fréquences plus basses, ce qui réduit considérablement le temps de calcul. De plus, nous avons pu abaisser l’ordre des filtres sans observer de grandes différences dans les résultats, en passant d’un filtre d’ordre 901 à un filtre d’ordre 31. Cela a également contribué à diminuer la charge computationnelle.

Les résultats sont sans équivoque : le temps d’exécution pour le **NaiveResampler** est de 1,64 s, tandis que pour le rééchantillonneur optimisé en cascade, il est de 40,7 ms, soit une amélioration de **40 fois plus rapide**.



Erreur



Nous avons calculé l'erreur en utilisant le **MSE** (Mean Squared Error) entre le spectre du signal original et celui rééchantillonné.

Les résultats obtenus sont les suivants :

- **NaiveResampler** : 0,00100
- **Resampler amélioré** : 0,00158

Au vu de l'ordre de grandeur des valeurs dans le spectre (issue de la FFT), ces erreurs semblent négligeables.

De plus, il est intéressant de privilégier le **Resampler optimisé** au **NaiveResampler**, malgré une erreur 58 % plus élevée, car le gain de temps obtenu est considérable et bien plus bénéfique dans un contexte pratique.

Conclusion

Dans ce TP, nous avons exploré les différentes étapes de la conception d'un système de conversion de fréquence d'échantillonnage (SRC) en passant par deux versions successives :

- **NaiveResampler** : Cette approche a permis de valider les principes fondamentaux du rééchantillonnage, incluant l'upsampling, le filtrage passe-bas, et le downsampling. Cependant, cette méthode s'est révélée coûteuse en mémoire et en temps de calcul, en grande partie à cause de l'upsampling direct et de l'utilisation d'un filtre d'ordre élevé (901).
- **Resampler amélioré**: Grâce à une décomposition des facteurs LLL et MMM, nous avons découpé le rééchantillonnage en plusieurs étapes successives. Cette optimisation a permis de travailler à des fréquences plus basses et d'obtenir un temps de calcul beaucoup plus rapide (40 fois plus rapide que le NaiveResampler), tout en maintenant une erreur de reconstruction négligeable.

En comparant les performances, le Resampler amélioré offre un compromis idéal entre qualité et efficacité. Bien que l'erreur du spectre soit légèrement supérieure à celle du NaiveResampler (MSE de 0,00158 contre 0,00100), elle reste négligeable par rapport à l'ordre de grandeur du spectre FFT.

Ce TP illustre ainsi l'importance de l'optimisation algorithmique et des compromis entre précision et efficacité computationnelle dans la conception de systèmes embarqués ou temps réel.