SAE31

Groupe Maxime Boison, Alerig Cuvillier, Pierre Hebert, Marc TAHON-DUJARDIN

Les Filtres passe bas

Un filtre passe-bas (ou filtre de coupure) est un circuit électronique ou un algorithme qui permet de passer les fréquences basses d'un signal tout en atténuant les fréquences hautes. Cela peut être utilisé pour réduire le bruit haute fréquence dans un signal ou pour accentuer les graves dans un son.

Un filtre passe-bas est généralement caractérisé par sa fréquence de coupure, qui détermine à quelle fréquence le filtre commence à atténuer les hautes fréquences. Plus la fréquence de coupure est basse, plus le filtre atténuera les hautes fréquences.

Un Filtre passe bas dans SuperCollider

Pour créer un filtre passe-bas dans SuperCollider, vous pouvez utiliser la classe `LPF`. Voici comment vous pouvez utiliser cette classe pour créer un filtre passe-bas et l'appliquer à un signal audio :

1. Créez une instance de la classe `LPF` en utilisant la syntaxe suivante :

```
lpf = LPF.ar(in, freq, mul: 1, add: 0)
```

- `in` est le signal audio que l'on souhaite filtrer.
- `freq` est la fréquence de coupure du filtre, c'est-à-dire la fréquence à laquelle le filtre commence à atténuer le signal.
- `mul` et `add` sont des paramètres optionnels qui permettent de régler l'amplitude et le décalage temporel du signal filtré.

1. Pour appliquer le filtre à un signal audio, vous pouvez simplement utiliser l'objet `lpf` comme n'importe quel autre objet audio dans SuperCollider. Par exemple, vous pouvez

utiliser la syntaxe suivante pour jouer le signal filtré :

```
lpf.play
```

Voici un exemple complet qui montre comment créer et utiliser un filtre passe-bas dans SuperCollider :

```
(
// Créer un générateur de bruit blanc
bruit = WhiteNoise.ar;

// Créer un filtre passe-bas avec une fréquence de coupure de 1000 Hz
lpf = LPF.ar(bruit, 1000);

// Jouer le signal filtré
lpf.play;
)
```

Un Filtre passe bas créé de toute pièces

```
~buffer=Buffer.read(s,"C:/Users/Maxime/Documents/IUT/travail/semestre
2/SAÉ22/AM_BaStatA_120A.wav")

~buffer.play(loop:true)
Ndef.clear
Ndef(\filtre).play

(
Ndef(\filtre,{
    var in,out,a0,a1,b1,omega0,tau;
    tau=1.0/s.sampleRate;
    omega0=2pi100;
    a0=tauomega0/((tauomega0)+2.0);
    a1=a0;
    b1=(2-(tauomega0))/((tauomega0)+2.0);
    in=PlayBuf.ar(1,~buffer,BufRateScale.ir(~buffer),loop:1);
    out=FOS.ar(in,a0,a1,b1);
    2*out;
});
)
```

1. Tout d'abord, le code lit un fichier audio nommé "AM_BaStatA_120A.wav" dans le répertoire "C:/Users/Maxime/Documents/IUT/travail/semestre 2/SAÉ22/ " et le charge dans un buffer nommé "~buffer" en utilisant la fonction `Buffer.read()`.
2. Ensuite, le code joue le contenu du buffer en boucle en utilisant la fonction `play()` avec l'argument "loop: true"
3. Après cela, le code utilise la fonction `Ndef.clear` pour supprimer tous les Ndef existants et ensuite utilise `Ndef(\filtre).play` pour créer un nouveau Ndef nommé "filtre" et le jouer.
4. Dans la prochaine section de code, on définit un nouveau Ndef nommé "filtre" qui contient un script de synthèse.
5. Le script de synthèse déclare plusieurs variables: "in", "out", "a0", "a1", "b1", "omega0" et "tau". "tau" est égal à l'inverse de la fréquence d'échantillonnage de SuperCollider

(s.sampleRate), "omega0" est égal à 2 pi fois la fréquence de coupure souhaitée (100 Hz), "a0" et "a1" sont les coefficients de filtrage pour les entrées du filtre, et "b1" est le coefficient de filtrage pour la sortie du filtre.

6. La variable "in" est égale à la sortie de la fonction `PlayBuf.ar()`, qui joue le contenu du buffer ~buffer à une fréquence de lecture définie par `BufRateScale.ir(~buffer)` avec un argument "loop:1" pour jouer en boucle.
7. La variable "out" est égale à la sortie de la fonction `FOS.ar()`, qui applique le filtre passe-bas en utilisant les coefficients de filtrage définis précédemment.
8. Enfin, le script retourne la sortie du filtre multipliée par 2.

Les Filtres passe haut

Un filtre passe-haut est un type de filtre utilisé pour atténuer les fréquences basses d'un signal, en laissant passer les fréquences élevées. Il est utilisé pour améliorer les sons aigus dans les applications audio, pour supprimer les bruits de fond dans les communications, pour accentuer les détails dans les images numériques, etc.

Les filtres passe-haut ont une réponse en fréquence qui est le complémentaire de la réponse en fréquence d'un filtre passe-bas de même ordre. La fréquence de coupure est le point où le filtre passe-haut coupe les fréquences basses et laisse passer les fréquences élevées.

Un Filtre passe haut dans SuperCollider

Pour créer un filtre passe-haut dans SuperCollider, vous pouvez utiliser la classe `HPF`. Voici comment vous pouvez utiliser cette classe pour créer un filtre passe-haut et l'appliquer à un signal audio :

1. Créez une instance de la classe `HPF` en utilisant la syntaxe suivante :

```
hpf = HPF.ar(in, freq, mul: 1, add: 0)
```

- `in` est le signal audio que l'on souhaite filtrer.
- `freq` est la fréquence de coupure du filtre, c'est-à-dire la fréquence à laquelle le filtre commence à atténuer le signal.
- `mul` et `add` sont des paramètres optionnels qui permettent de régler l'amplitude et le décalage temporel du signal filtré.

1. Pour appliquer le filtre à un signal audio, vous pouvez simplement utiliser l'objet `hpf` comme n'importe quel autre objet audio dans SuperCollider. Par exemple, vous pouvez utiliser la syntaxe suivante pour jouer le signal filtré :

```
hpf.play
```

Voici un exemple complet qui montre comment créer et utiliser un filtre passe-haut dans SuperCollider :

```
(
// Créer un générateur de bruit rose
bruit = PinkNoise.ar;

// Créer un filtre passe-haut avec une fréquence de coupure de 1000 Hz
hpf = HPF.ar(bruit, 1000);

// Jouer le signal filtré
hpf.play;
)
```

Les Filtres passe bande

Un filtre passe-haut est un type de filtre utilisé pour atténuer les fréquences élevées d'un signal, en laissant passer les fréquences basses. Il est utilisé pour améliorer les sons graves dans les applications audio, pour accentuer les bruits de fond dans les communications, pour supprimer les détails parasites dans les images numériques, etc.

Les filtres passe-haut ont une réponse en fréquence qui est le complémentaire de la réponse en fréquence d'un filtre passe-bas de même ordre. La fréquence de coupure est le point où le filtre passe-haut coupe les fréquences élevées et laisse passer les fréquences basses. Dans SuperCollider pour créer un filtre passe-haut on utilise la classe `HPF` . Il est utilisé de la même manière que le filtre LPF, c'est à dire, on crée une instance de la classe en définissant les paramètres d'entrée, de fréquence de coupure et on peut ajouter des paramètres optionnels pour régler l'amplitude et le décalage temporel du signal filtré.

Un Filtre passe bande dans SuperCollider

Pour créer un filtre passe-bande dans SuperCollider, vous pouvez utiliser la classe `BPF`. Voici comment vous pouvez utiliser cette classe pour créer un filtre passe-bande et l'appliquer à un signal audio :

1. Créez une instance de la classe `BPF` en utilisant la syntaxe suivante :

```
bpf = BPF.ar(in, centerFreq, bw, mul: 1, add: 0)
```

- `in` est le signal audio que l'on souhaite filtrer.
- `centerFreq` est la fréquence centrale du filtre passe-bande, c'est-à-dire la fréquence autour de laquelle le filtre passe-bande laisse passer le signal.
- `bw` est la bande passante du filtre, c'est-à-dire la largeur de la bande de fréquences autour de la fréquence centrale qui est laissée passer par le filtre.
- `mul` et `add` sont des paramètres optionnels qui permettent de régler l'amplitude et le décalage temporel du signal filtré.

1. Pour appliquer le filtre à un signal audio, vous pouvez simplement utiliser l'objet `bpf` comme n'importe quel autre objet audio dans SuperCollider. Par exemple, vous pouvez utiliser la syntaxe suivante pour jouer le signal filtré :

```
bpf.play
```

Voici un exemple complet qui montre comment créer et utiliser un filtre passe-bande dans SuperCollider :

```
(
// Créer un générateur de bruit rose
bruit = PinkNoise.ar;

// Créer un filtre passe-bande avec une fréquence centrale de 500 Hz et une bande
passante de 100 Hz
bpf = BPF.ar(bruit, 500, 100);

// Jouer le signal filtré
bpf.play;
)
```

Les Filtres à peigne

Un filtre à peigne est un type de filtre utilisé pour sélectionner des bandes de fréquences spécifiques d'un signal, en laissant passer les fréquences souhaitées et en atténuant les autres. Il est utilisé pour séparer des signaux multiples dans les applications audio, pour extraire des informations spécifiques dans les communications, pour accentuer des détails dans les images numériques, etc.

Les filtres à peigne ont une réponse en fréquence qui est caractérisée par des bandes de fréquences passantes et des bandes de fréquences atténuées. Les fréquences de coupure sont les points où les bandes de fréquences passantes et atténuées se croisent. Dans SuperCollider pour créer un filtre à peigne on utilise la classe `CombN` ou `CombC`. Il est utilisé en définissant les paramètres d'entrée, de fréquences de coupure et les coefficients de résonance. On peut ajouter des paramètres optionnels pour régler l'amplitude et le décalage temporel du signal filtré.

Un Filtre à peigne dans SuperCollider

Pour créer un filtre à peigne dans SuperCollider, vous pouvez utiliser la classe `CombN` ou `CombC`. Voici comment vous pouvez utiliser cette classe pour créer un filtre à peigne et l'appliquer à un signal audio :

1. Créez une instance de la classe `CombN` ou `CombC` en utilisant la syntaxe suivante :

```
comb = CombN.ar(in, delaytime, decaytime, mul: 1, add: 0)
```

- `in` est le signal audio que l'on souhaite filtrer.
- `delaytime` est le temps de délai du filtre à peigne, c'est-à-dire le temps entre l'entrée et la sortie du signal filtré.
- `decaytime` est le temps de décroissance du filtre à peigne, c'est-à-dire le temps pour lequel le signal filtré est atténué.
- `mul` et `add` sont des paramètres optionnels qui permettent de régler l'amplitude et le décalage temporel du signal filtré.

1. Pour appliquer le filtre à un signal audio, vous pouvez simplement utiliser l'objet `comb` comme n'importe quel autre objet audio dans SuperCollider. Par exemple, vous pouvez utiliser la syntaxe suivante pour jouer le signal filtré :

```
comb.play
```

Voici un exemple complet qui montre comment créer et utiliser un filtre à peigne dans SuperCollider :

```
(
// Créer un générateur de bruit blanc
scie = Saw.ar(440);

// Créer un filtre à peigne avec un temps de délai de 0.2 secondes et un temps de décroissance de 1 seconde
comb = CombN.ar(scie, 0.2, 0.2, 4);

// Jouer le signal filtré
comb.play;
)
```

Les Filtres passe tout

Un filtre passe-tout est un type de filtre utilisé pour laisser passer un certain intervalle de fréquences d'un signal, tout en atténuant les fréquences en dehors de cet intervalle. Il est utilisé pour accentuer des bandes de fréquences spécifiques dans les applications audio, pour extraire des informations spécifiques dans les communications, pour accentuer des détails dans les images numériques, etc.

Un Filtre passe tout créé de toute pièces

```
~buffer=Buffer.read(s,"C:/Users/Maxime/Documents/IUT/travail/semestre 2/SAÉ22/AM_BaStatA_120A.wav")

~buffer.play(loop:true)
Ndef.clear
Ndef(\FTP).play

(
Ndef(\FPT,{
    var in,out,a0,a1,b1,omega0,tau;
    tau=1.0/s.sampleRate;
    omega0=2pi2000;
    a0=((tauomega0)-2.0)/((tauomega0)+2.0);
    a1=1;
    b1=((((tauomega0)-2.0)/((tauomega0)+2.0))(-1));
    in=PlayBuf.ar(1,~buffer,BufRateScale.ir(~buffer),loop:1);
    out=FOS.ar(in,a0,a1,b1);
    2out;
});
)
```

Ce code utilise des fonctions de SuperCollider pour lire un fichier audio, le jouer en boucle, puis appliquer un filtre passe-tout sur ce signal audio.

Les lignes 1 à 2 chargent un fichier audio à partir d'un emplacement spécifique sur l'ordinateur, puis démarre la lecture de ce fichier en boucle.

Ligne 4 à 5 utilise Ndef.clear pour effacer tous les Ndef existants et Ndef(\FTP) pour jouer un Ndef existant.

Lignes 6 à 22 définissent un nouveau Ndef nommé \FPT. Il définit des variables pour stocker les paramètres du filtre passe-tout, comme le tau, omega0, a0, a1, b1. Il utilise ensuite la fonction PlayBuf pour jouer le buffer chargé dans la variable in. Il utilise ensuite FOS (filtrage par ordre supérieur) pour appliquer le filtre passe-tout sur le signal audio en utilisant les paramètres définis précédemment. Enfin, le signal filtré est amplifié par 2 avant de sortir.

Équation pour retrouver les coefficients

$$H(w) = \frac{1 - \mathrm{j}\frac{\omega}{\omega_0}}{1 + \frac{\omega}{\omega_0}}$$

$$H(w) = \frac{\mathrm{I} - \frac{2}{\tau\omega_0} * \frac{I - \Delta}{I + \Delta}}{\mathrm{I} + \frac{2}{\tau\omega_0} * \frac{I - \Delta}{I + \Delta}}$$

$$H(w) = \frac{\tau\omega_0(I + \Delta) - 2(I - \Delta)}{\tau\omega_0(I + \Delta) + 2(I - \Delta)}$$

$$H(w) = \frac{(\tau\omega_0 - 2)I + 2(\tau\omega_0 + 2)\Delta}{(\tau\omega_0 + 2)I + 2(\tau\omega_0 - 2)\Delta}$$

$$(\tau\omega_0 + 2)Y_n + (\tau\omega_0 - 2)Y_{n-1} = (\tau\omega_0 - 2)X_n + (\tau\omega_0 + 2)X_{n-1}$$

$$Y_n = a_0X_n + a_1X_{n-1} + b_1Y_{n-1}$$

$$a_0 = \frac{\tau\omega_0 - 2}{\tau\omega_0 + 2}$$

$$a_1 = \frac{\tau\omega_0 + 2}{\tau\omega_0 + 2} = 1$$

$$B_1 = -\frac{\tau\omega_0 - 2}{\tau\omega_0 + 2} = -a_0$$

Les filtres passe bas et passe haut résonnant

Un filtre passe-bas résonnant est un type de filtre utilisé pour accentuer les fréquences basses d'un signal en utilisant un effet de résonance. Il est souvent utilisé dans les applications audio pour ajouter de la profondeur et de la chaleur aux sons basse fréquence. Il est caractérisé par une fréquence de coupure spécifique qui détermine la plage de fréquences qui seront accentuées, ainsi qu'un paramètre de résonance qui peut être utilisé pour ajouter de la profondeur à l'effet de renforcement des fréquences basses.

Un filtre passe-haut résonnant est un type de filtre utilisé pour accentuer les fréquences aigües d'un signal en utilisant un effet de résonance. Il est souvent utilisé dans les applications audio pour ajouter de l'éclat et de la précision aux sons haute fréquence. Il est caractérisé par une fréquence de coupure spécifique qui détermine la plage de fréquences qui seront accentuées, ainsi qu'un paramètre de résonance qui peut être utilisé pour ajouter de la profondeur à l'effet de renforcement des fréquences aigües.

```
~buffer=Buffer.read(s,"C:/Users/Maxime/Documents/IUT/travail/semestre
2/SAÉ22/AM_BaStatA_128D.wav")

~buffer.play(loop:true)
Ndef.clear
Ndef(\RLPF).play
Ndef(\RHPF).play

//filtre passe-bas à résonnance
(
Ndef(\RLPF,{
    var in;
    in=PlayBuf.ar(1,~buffer,BufRateScale.ir(~buffer),loop:1);
    RLPF.ar(in,1000);
});
)

//filtre passe-haut à résonnance
(
Ndef(\RHPF,{
    var in;
    in=PlayBuf.ar(1,~buffer,BufRateScale.ir(~buffer),loop:1);
    RHPF.ar(in,10000);
});
)
```