Traitement du signal audio

TP - 3SYM : réalisation d'effets audio sous matlab

24 novembre 2017

Ce TP a pour objet l'étude de quelques aspects de la chaîne d'acquisition, de traitement et de restitution du signal audio. Seront étudiés successivement :

- L'acquisition d'un signal (musical ou parole) à partir d'un microphone
- La mise en forme de ce signal destinée à en optimiser la dynamique et l'occupation spectrale
- L'ajout d'ambiance réverbérante

Il y a beaucoup de choses à réaliser : de fait, si vous êtes à cours de temps, vous pouvez sauter certaines parties.

Le développement des traitements et effets sont réalisés sous Matlab.

1 Acquisition - Prise de son

Il s'agit de réaliser l'acquisition d'un signal d'une trentaine de secondes, soit de parole, soit de musique si vous disposez d'un instrument. L'objectif est que vous acquerriez un signal **brut** sur lequel il sera intéressant de réaliser des traitements. Inutile donc d'utiliser un fichier MP3 issu du net !!! Ce signal sera utilisé par la suite comme objet de traitement, il convient donc toutefois d'en réaliser l'acquisition de manière soignée.

Matériel à votre disposition :

- microphones électrodynamiques
- deux microphones LEM électrostatique (alimentation par pile : à vérifier, donc)
- un microphone AT2020 électrostatique (attention : polarisation 48V alimentation phantom)
- bonnette et filtre antipop
- convertisseur jack 3.5/6.5 ou 3.5/XLR
- pieds de micro

— carte son PC (entrée mic avec gain réglable via le panneau de contrôle de Windows)
— Logiciel : Audacity permet d'enregistrer un son de manière simple et conviviale.
— Chambre anéchoïde le cas échéant pour tester l'absence de réverbération.

Quelques fonctions matlab utiles en traitement audio : `wavread, wavwrite, spectrogram, colorera, mp3read`

Instructions/consignes de travail :

— Vous veillerez en particulier à régler le gain de l'entrée mic de manière à disposer de la meilleure dynamique (plage maxi sans écrêtage). Vous vous pouvez réaliser plusieurs enregistrements avec le micro plus ou moins loin de la source sonore (5cm à 50cm), afin de tester l'influence de *l'effet de proximité* sur le spectre du signal.
— la prise de son sera l'occasion de tester l'influence de deux paramètres essentiels en audionumérique : la fréquence d'échantillonnage et la profondeur de quantification. Vous pouvez varier fs de $8kHz$ à $48kHz$, et la profondeur de quantification de 8 bits à $24bits$. Comparer le résultat, d'une part à l'écoute (restitution), d'autre part sur le spectre (vous pouvez utiliser la fonction `spectrogram` de Matlab).
— Estimation du bruit de fond : enregistrer avec le microphone débranché ; puis avec le micro connecté, mais avec un chiffon ou mouchoir devant la capsule (vous aurez alors le bruit total incluant l'électronique du micro) ; amplifier sous audacity afin de pouvoir entendre confortablement ; étudier le spectrogramme du bruit seul (répartition énergétique : bruit blanc ? rose ?) ; comparer avec le niveau attendu du bruit de quantification
— Comparer le bruit de fond dû à la chaîne d'acquisition, avec le bruit ambiant (c'est à dire, hors signal utile, parole ou musique). Ce point sera utile pour la suite (noise gate).

2 Réalisation d'un noise-gate

Il s'agit habituellement de la première étape de traitement d'un signal audio enregistré en condition réelles, c'est-à-dire avec du bruit ambiant. Le noise-gate est un effet qui coupe (=réduit au silence) les parties du signal se trouvant en-dessous d'un certain niveau, dans le but de supprimer le bruit. L'idée clé est qu'en présence de signal utile (parole ou musique), le bruit est généralement masqué (cf. cours psychoacoustique sur le masquage fréquen-

tiel), et l'auditeur ne le perçoit pas ; en revanche il est clairement audible lors des plages de silence musical ou de pause entre deux phrases. Lorsque le signal sera mélangé à d'autres signaux lors de l'étape de mixage, il est souvent primordial (sauf raisons artistiques particulières) que le moins de bruit ambiant subsiste.

On dit que la porte est ouverte lorsque le signal passe tel quel au travers du traitement ; qu'elle est fermée si le signal est réduit (ou atténué).

Réalisation naïve Ecrire une fonction `xs=noiseGate1(xe, seuil, attenuation)` qui prend le signal audio sous forme d'un vecteur en entrée xe, et restitue le signal après application d'une noise gate très simplifiée, consistant à atténuer d'une quantité fixée (en dB) les échantillons se trouvant sous le seuil (paramètre *seuil*). Il est intéressant de normaliser le signal avant de le traiter afin de s'affranchir des problèmes de rapport niveau/seuil.

Tester l'effet de différents seuils et différentes atténuations (ne pas se contenter de spectrogrammes, mais écouter ! ! !). Commentaires ?

Réalisation avancée Une noise gate améliorée dispose de paramètres dynamiques supplémentaires (cf. figure 1) :

- attack time : c'est le temps que met la porte à s'ouvrir lorsque le signal dépasse le seuil
- holding time : c'est le temps minimal pendant lequel la porte reste ouverte (même si le signal passe sous le seuil) ; ceci permet notamment d'éviter les effets de hachage, la porte ne pouvant pas se fermer tant que ce temps n'est pas écoulé.
- release time : c'est le temps (supérieur au précédent, donc) que met la porte à se fermer lorsque le signal passe sous le seuil (à condition que le holding time ait été atteint).

Pour faciliter l'écriture de la fonction `xs=noiseGate2(xe, seuil, attenuation, attack, hold, release)`, je vous suggère d'utiliser un programmation par état en utilisant une variable *state* valant le cas échéant :

- CLOSED : porte fermée
- OPENING : le signal vient de passer le seuil, la porte est en train de s'ouvrir (prise en compte de l'attaque)
- OPEN : la porte est ouvre
- CLOSING : le signal vient de passer sous le seuil, la porte est (si le hold time est respecté) en train de se fermer (prise en compte du release time).

Vous choisirez des pentes linéaires (cf. figure 1) pour l'ouverture et la fermeture (des pentes exponentielles sont généralement plus esthétiques, mais

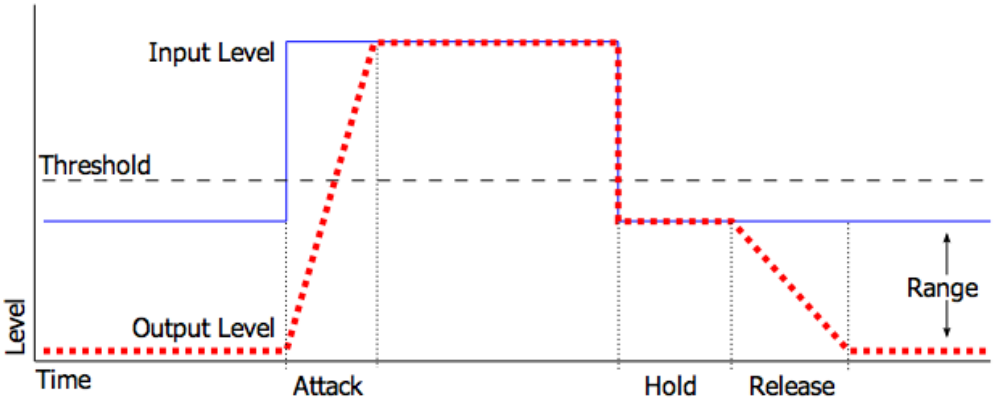

FIGURE 1 – Paramètres de la noise-gate (range représente l'atténuation).

aussi plus compliquées à programmer). Dans ce cas, un simple compteur suffira pour l'ouverture et la fermeture.

A suivre... avec l'étape suivante, la compression dynamique et l'égalisation.

3 Réalisation d'un compresseur dynamique

Afin d'optimiser le nombre de bits utilisés pour numériser le signal, il est souvent intéressant de rehausser les faibles niveaux, ou, alternativement, de réduire les forts niveaux. Cette opération est appelée *compression dynamique*, et consiste à appliquer la courbe de gain suivante au signal d'entrée :

Un autre intérêt du compresseur est esthétique : il permet, en réduisant la dynamique globale du signal, de faire ressortir certains passages ou instruments qui seraient à peine audible autrement. Il donne également une impression de "lissage" qui, dans certains styles musicaux, est appréciée.

Les paramètres d'un compresseur dynamique sont :

— compression ratio : la réduction de gain au-dessus du seuil
— threshold : le seuil (en dB sur le signal d'entrée) au-dessus duquel le signal est compressé.
— attack : le temps nécessaire au compresseur pour se déclencher
— release : le temps nécessaire, lorsque le signal repasse sous le seuil, pour que le gain reprenne sa valeur unitaire.

Ces quatre paramètres standards sont parfois complétés d'un paramètre "hold" (comme pour la noise-gate), d'un paramètre "knee" adouciçant la courbe de compression afin que le compresseur rentre plus ou moins progressivement

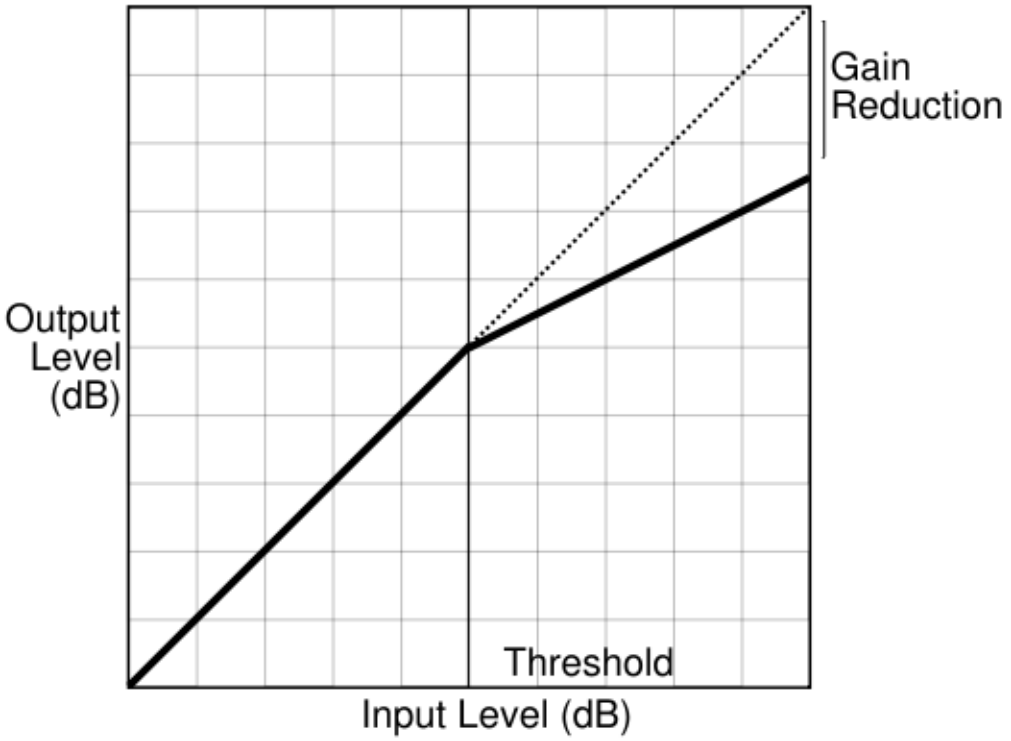

FIGURE 2 – Le compresseur dynamique atténue tous les signaux dont l'amplitude est supérieure à un certain seuil (threshold paramètre). Ici le rapport de compression (compression ratio) vaut 2 :1.

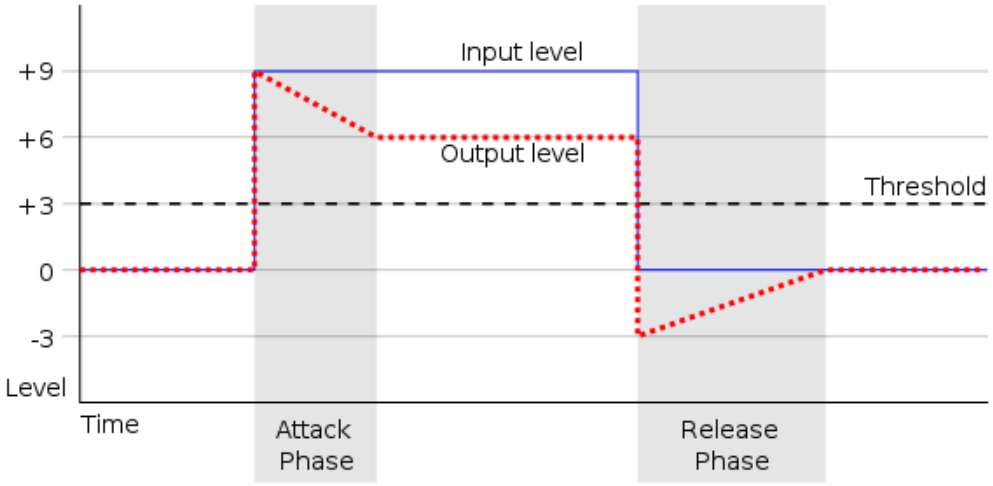

FIGURE 3 – Le changement de gain n'est pas immédiat, mais suit au contraire une enveloppe similaire à la noise-gate (attack/release).

en action, et enfin de la possibilité de faire agir un filtre passe-bande sur le signal de compression (cf. de-esser).

Remarque : un compressor avec un taux de compression infini, est appelé un **limiteur**.

Le fonctionnement standard d'un compresseur est le suivant :

— le niveau RMS du signal d'entrée est mesuré régulièrement
— lorsque ce niveau dépasse le seuil, le gain est réduit à sa valeur donnée par le taux de compression
— cette réduction de gain devient effective au bout d'un temps donné par le paramètre d'attack
— lorsque le niveau RMS du signal passe sous le seuil, le gain revient à sa valeur initiale (avec une courbe temporelle dépendant du paramètre "release")

Le paramètre "make up gain" est un paramètre additionnel, permettant après compression de redonner au signal une amplitude telle qu'il est perçu avec la même intensité que le signal non compressé. Il s'agit simplement d'une amplification additionnelle.

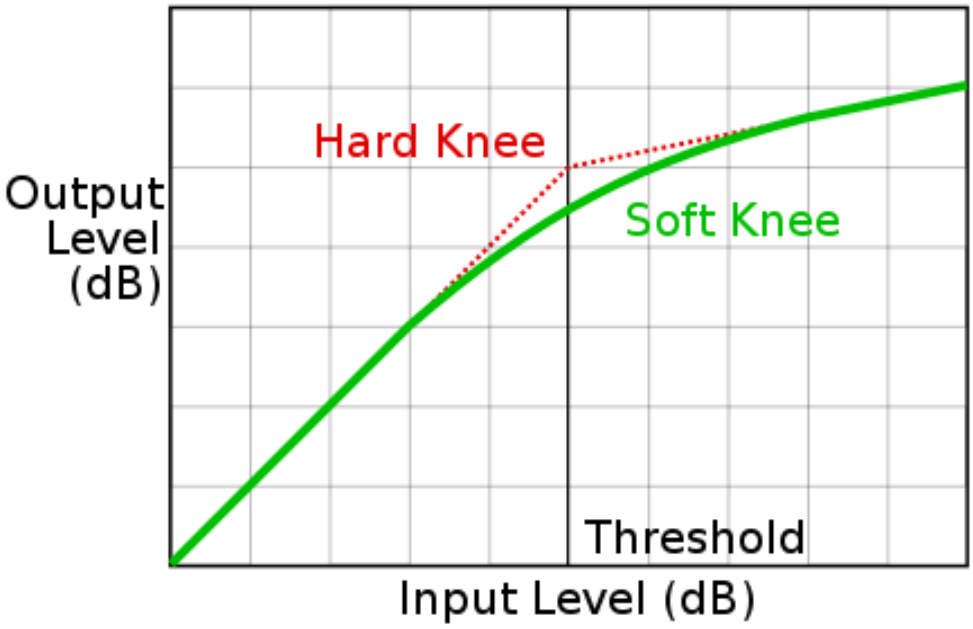

FIGURE 4 – Un compresseur à Soft Knee voit le compresseur entrer progressivement en action. La compression est plus douce.

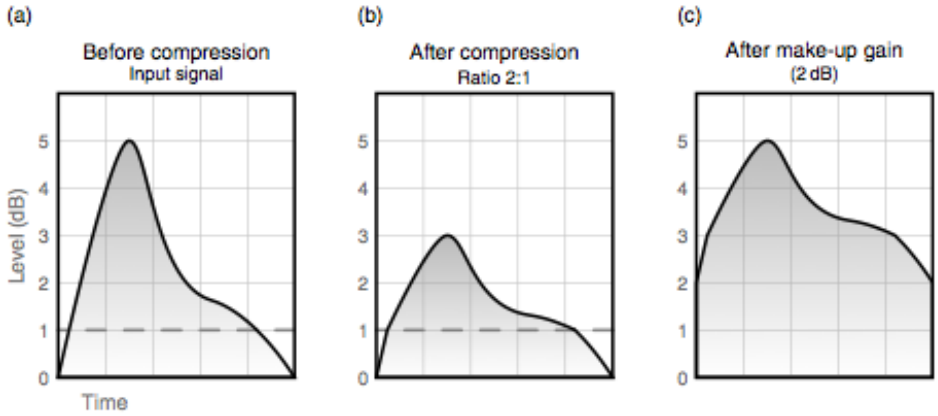

FIGURE 5 – Le make-up gain permeant de redonner un niveau acceptable au signal.

4 Réalisation d'une spatialisation par délai et réverbération

Dans cette partie vous devez implémenter un algorithme de spatialisation à partir de délais et de réverbération. L'objectif est de donner une impression subjective de spatialisation, comme si la voix avait été enregistrée dans une pièce à l'acoustique réverbérante.

Implémentation d'un délai : le signal d'entrée est retardé, puis réinjecté à l'entrée après atténuation. Les paramètres sont :

— le retard, en mille-secondes

— le feedback, ou taux de réinjection (atténuation de la boucle de réaction).

Testez plusieurs valeurs de délai sur votre enregistrement vocal (de quelques ms à quelques centaines de ms), et commentez sur la perception spatiale de l'enregistrement. Testez également la superposition de deux ou trois délais avec des paramètres très différents entre eux.

L'implémentation de la réverbération qui vous est demandée repose sur l'algorithme de convolution par transformée de Fourier. Des réponses impulsionnelles se salles typiques sont fournies sur mon site (fichiers SDIR). La sortie de la réverbération est obtenues en convoluant le signal d'entrée $x(n)$ avec la réponse impulsionnelle de la salle $h(n)$: $y = x * h$. Pour accélérer le calcul (qui est proportionnel au carré de la longueur N de la réponse impulsionnelle), on passe dans le domaine fréquentiel : $Y = X \cdot H$. Il est utile de calculer la transformée de Fourier de h une fois pour toute, et celle de x, par FFT (algorithme en temps $N \ln N$).

Remarque : attention au choix du fenêtrage (hamming, blackman, hanning,...) lors de l'application de la FFT. Attention également au fait que le signal réverbéré est plus long que le signal original (de N exactement).

5 Réalisation d'un égaliseur

Cette ultime partie consiste à réaliser un égaliseur sous forme de filtres à réponse impulsionnelle infinie. Ce système est composé de :

— un filtre passe-haut non-résonant à 4 pôles, et l'équivalent en filtre passe-bas ;

— un filtre "shelf", équivalent au filtre analogique de fonction de transfert

$$T(p) = \frac{1 + \tau_1 p}{1 + \tau_2 p}$$

— un filtre paramétrique passe-bande, de fréquence centrale f_0 et de coefficient de qualité Q, tous deux variables.

Exemple de filtre passe-bas d'ordre 4 coupant à $f_e/10$:

$$y[n] = (1 * x[n-4]) + (4 * x[n-3]) + (6 * x[n-2]) + (4 * x[n-1]) + (1 * x[n-])$$

$$+(-0.85 * y[n-4]) + (3.53 * y[n-3]) + (-5.52 * y[n-2]) + (3.84 * y[n-1])$$