À la recherche du rééchantillonneur parfait

Laurent de Soras

2005.10.14

web : http://ldesoras.free.fr

ABSTRACT

Cet article technique traite de la synthèse sonore numérique. De toutes les méthodes de synthèse connues, il y en a une qui ne se démodera jamais : la reproduction d'échantillons sonores. Nous présentons dans ce document une méthode efficace pour rejouer des échantillons à une vitesse variable, sans perte de qualité perceptible. Cette technique est particulièrement adaptée aux synthétiseurs tournant sur des ordinateurs personnels.

MOTS CLÉS

Synthèse sonore, échantillons, ré-échantillonnage, interpolation, repliement

0. Structure du document

0.1 Table des matières

0.2 Historique des révisions

Version	Date	Modifications
1.0	2003.06.20	Première publication, en Anglais
1.4	2005.10.14	Version française

0.3 Glossaire

CPU	Central Processing Unit – Unité centrale
DAW	Digital Audio Workstation – Station de travail audio-numérique
FFT	Fast Fourier Transform – Transformée de Fourier rapide
FIR	Finite Impulse Response – Filtre à réponse finie
IFFT	Inverse Fast Fourier Transform – Tranformée de Fourier inverse rapide
IIR	Infinite Impulse Response – Filtre à réponse infinie
MAC	Multiply and Accumulate – Multiplication et Accumulation
MIP	Ici, Multum In Parvo
SIMD	Single Instruction, Multiple Data. Ex : AltiVec, SSE, MMX...
SNR	Signal to Noise Ratio – Rapport signal/bruit
TPDF	Triangular Probability Density Function – Fonction de densité de probabilité triangulaire

1. Introduction

Le noyau d'un synthétiseur à lecture d'échantillons (appelé *sampler*) est l'algorithme de ré-échantillonnage, utilisant un filtre d'interpolation. Ces filtres ont été étudié en long, en large et en travers dans la littérature. Pour nous, la principale difficulté réside dans la maximisation de la fidélité de la réponse en bande passante et l'optimisation du SNR, qui est dégradé par le repliement spectral (*aliasing*) introduit par l'interpolation. Nous allons présenter dans cet article une méthode, ou plus exactement un ensemble de méthodes, pour interpoler efficacement les données échantillonnées.

2. Récapitulatif des techniques classiques de ré-échantillonnage

2.1 Différents types d'interpolateurs

Les interpolateurs tombent généralement dans deux catégories : polynomiale et FIR. La différence essentielle réside dans l'implémentation, car les interpolateurs polynomiaux peuvent être décrits par leur réponse impulsionnelle.

Les interpolateurs d'ordre faible donnent des résultats tout à fait décents à pour coût de calcul relativement bas [1]. Par exemple, l'interpolateur Hermite du 3ème ordre à 4 points effectue un très bon boulot pour très peu de multiplications et d'additions. On peut représenter le calcul de l'interpolation comme une multiplication matricielle. L'interpolateur Hermite 4,3 s'exprime ainsi de la façon suivante :

$$f_{3,4\,H}(t+k)=\begin{bmatrix}1 & t & t^2 & t^3\end{bmatrix}\times\frac{1}{2}\begin{bmatrix}0 & 2 & 0 & 0\\ -1 & 0 & 1 & 0\\ 2 & -5 & 4 & -1\\ -1 & 3 & -3 & 1\end{bmatrix}\times\begin{bmatrix}x_{k-1}\\ x_k\\ x_{k+1}\\ x_{k+2}\end{bmatrix} \qquad (2.1\text{-}1)$$

Pour augmenter la qualité, il est nécessaire d'augmenter l'ordre du polynôme et le nombre de points utilisés. Malheureusement, les calculs nécessaires croissent beaucoup plus rapidement que les gains en qualité, ce qui limite l'intérêt de l'interpolation polynomiale.

Nous disposons également des convolutions par filtre FIR. Il s'agit de l'application directe du théorème d'échantillonnage, auquel on ajoute une fonction fenêtre pour limiter le terme `sinc` théoriquement infini. Cela conduit à des algorithmes polyphases multi-fréquences comme ceux décrits en [4]. Ils ont de nombreux avantages et inconvénients :

- Possibilité de choisir précisément la réponse en fréquence du filtre.
- Technique simple à mettre en oeuvre marchant bien pour tout type de de ré-échantillonnage.

- Ne sont gérés que les facteurs de ré-échantillonnage en fractions entières (N/M).
- L'impulsion correspondant à un unique facteur N/M prend beaucoup de mémoire.
- Elle doit avoir un nombre suffisant de passages par zéro pour obtenir une qualité décente (faible repliement, bande passante plate).
- Le coût des calculs augmente proportionnellement avec le facteur de ré-échantillonnage.

Les interpolateurs FIR de haute qualité sont bien moins coûteux que les polynomiaux de qualité équivalente. Ce design est très efficace pour des facteurs fixes. Malheureusement avec les synthétiseurs, il y a au moins un facteur par note, sans compter les effets plus ou moins subtiles de variation de hauteur de la note. Une méthode plus avancée consiste à interpoler linéairement les coefficients de l'impulsion de façon à troquer un peu de temps de calcul et de qualité contre d'importantes économies de mémoire. Cela résout le problème des taux de ré-échantillonnage, mais l'implémentation n'est pas aussi efficace que ce qu'elle pourrait être à cause de l'interpolation linéaire sur l'impulsion. Le pas de celle-ci varie en fonction du taux de ré-échantillonnage. Ce point spécifique sera traité un peu plus loin dans le document.

Nous pouvons décrire rapidement quelques attributs caractérisants la réponse d'un filtre d'interpolation, qu'il soit polynomial ou FIR :

- La largeur de la bande-fréquence de transition. Sa partie empiétant dans la bande passante affecte sa platitude. Celle au contact de la bande coupée génère du repliement.

- L'amplitude du premier lobe latéral. Les lobes sont les images de la bande passante répliquées dans les zones supérieures du spectre au cours du processus, générant du repliement lors du ré-échantillonnage.
- La pente de la partie du lobe principal formant la bande de transition. Elle peut souvent être mesurée en dB/octave.

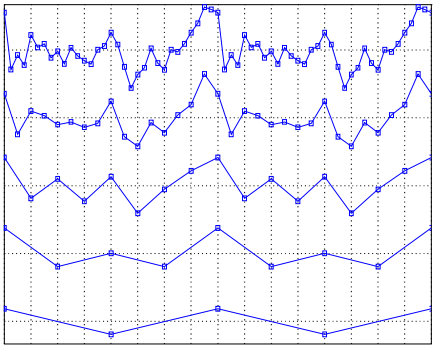
2.2 Le MIP-mapping

En outre, les interpolateurs sont souvent utilisés avec des échantillons MIP-mappés. Cette technique est empruntée au monde de l'image de synthèse dans lequel elle est utilisée pour réduire le moiré apparaissant lors du tracé de textures sur des objets en trois dimensions.

Le document [8] nous donne une bonne définition, que nous traduisons ici : « Le MIP mapping consiste en plusieurs images d'une même texture à différentes résolutions pour représenter la texture figurant à différentes distances du spectateur : l'image aux dimensions les plus grandes est placée au premier plan, remplacée par les plus petites au fur et à mesure que l'on s'éloigne vers l'arrière-plan. Chaque résolution est définie comme un niveau de MIP map. Le MIP mapping aide à éviter les bords en escaliers dans une image, qui peuvent être la conséquence du redimensionnement d'une image bitmap. MIP vient du Latin multum in parvo, qui veut dire beaucoup de substance dans un petit volume. »

Image bitmap sous forme de MIP-map.

Ainsi, le moteur de rendu peut faire ses calculs sur un petit nombre de pixels d'une texture distante afin de calculer la couleur d'un pixel à l'écran. La bonne moyenne de couleur est pré-calculée pour chaque niveau, les calculs restants demeurent toujours légers quelle que soit la distance. C'est exactement le même principe que nous utiliserons avec les sons. L'échantillon original existe sur plusieurs niveau. Chaque niveau est filtré par un passe-bas quasi-optimal et décimé à une fréquence d'échantillonnage donnée. Généralement, les niveaux sont espacés régulièrement d'une octave.



Cinq niveaux de MIP-map audio, numérotés de 0 à 4.

Le MIP mapping permet une réduction massive du repliement quand on ré-échantillonne à un fort taux avec un filtre de ré-échantillonnage de faible complexité. Cependant, les haut taux de ré-échantillonnage deviennent de moins en moins nécessaires avec les banques de sons « multi-layer », dont les échantillons sont enregistrés pour plusieurs notes de l'instrument. Ces haut taux restent en outre utiles pour la synthèse de formes d'onde simples dont l'aspect est invariant quelque soit la fréquence du signal. Toutefois il ne faut pas attendre de miracle du MIP mapping qui ne retire pas complètement le repliement spectral, en particulier celui créé par l'interpolateur.

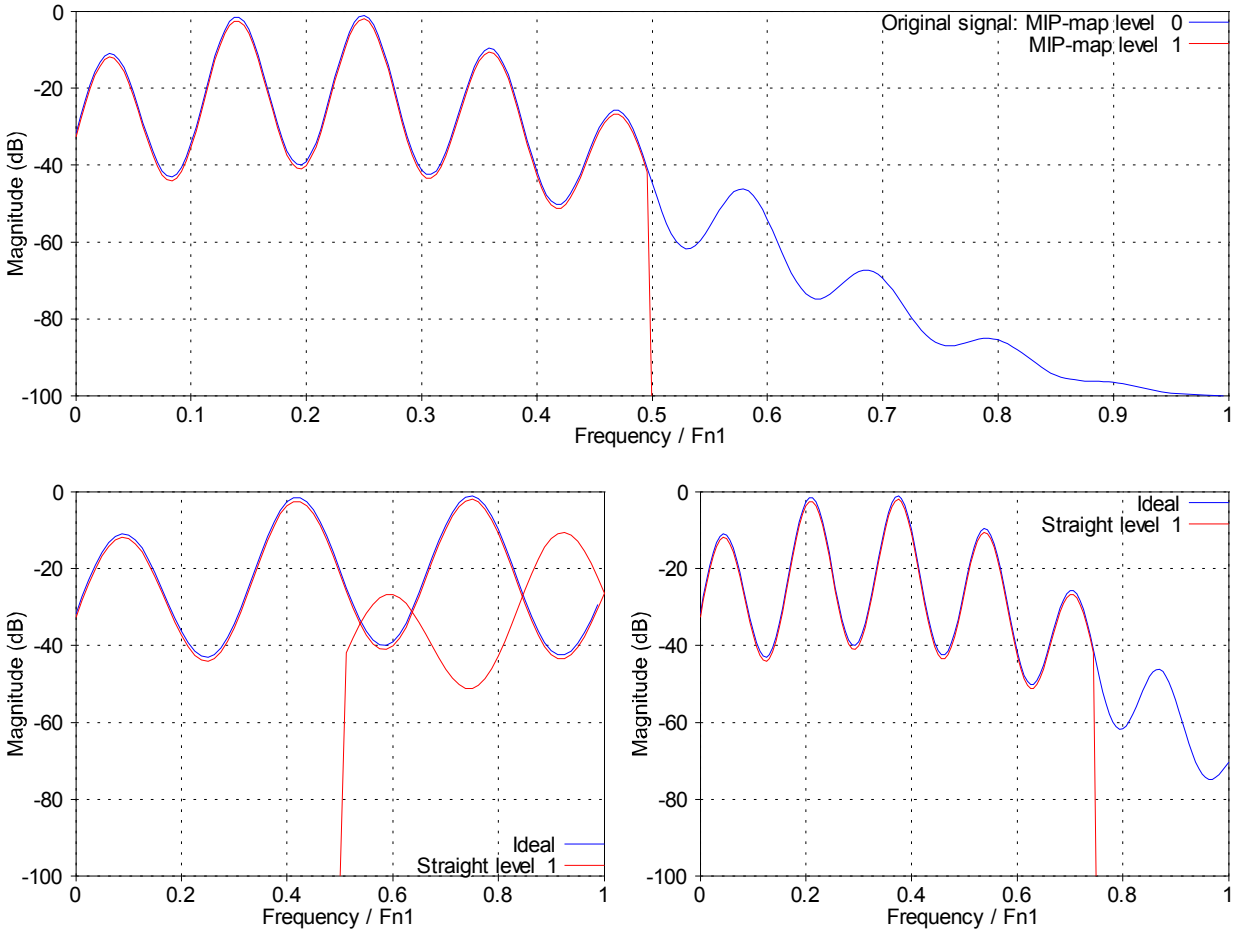
2.3 Sur-échantillonnage de l'échantillon source

Comme décrit dans les documents [1] et [2], les performance d'un interpolateur polynomial sont considérablement augmentées quand les données d'entrée sont sur-échantillonnées ; seule une petite partie du spectre de la source est occupée. Cela donne une bande passante utile bien plus plate et des lobes latéraux (miroirs du spectre original) réduits. Plus la source est sur-échantillonnée, meilleures sont les performances. Le désavantage est une consommation de mémoire d'autant plus importante. Le sur-échantillonnage de la source est souvent utilisé conjointement avec le MIP mapping, augmentant le SNR obtenu.

3. La solution proposée

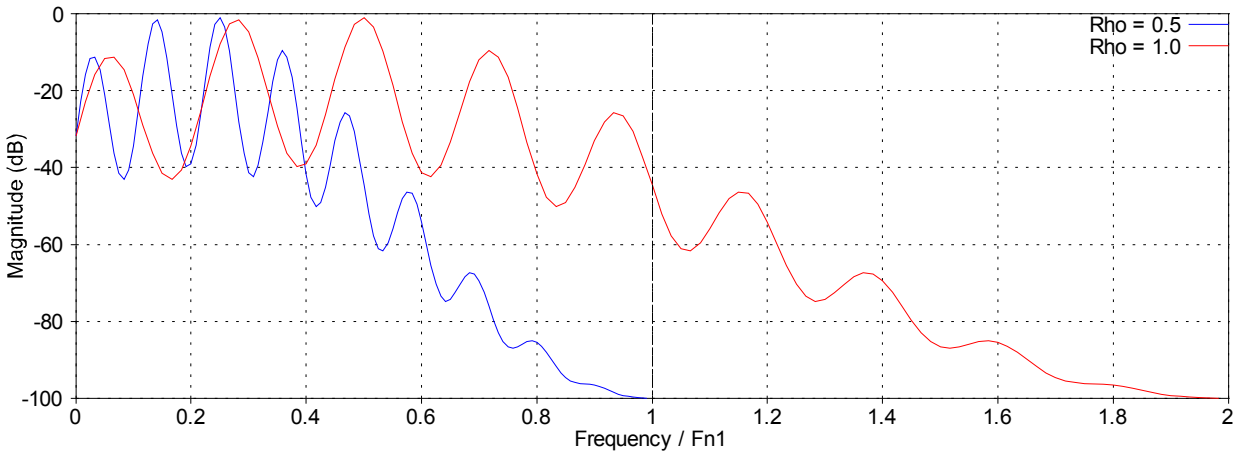
3.1 Vue d'ensemble

La méthode que nous allons décrire est un processus en trois étapes, consistant en un MIP mapping, une interpolation et un sous-échantillonnage. Supposons que nous voulons ré-échantillonner notre son d'un facteur r. Le MIP mapping à l'octave pré-filtre et décime le sample, permettant de grandes valeurs pour r tout en évitant de coûteux calculs. S'il était appliqué directement sur un niveau de MIP map, un interpolateur idéal créerait du repliement en ré-échantillonnant par un facteur local $\rho > 1$, ou donnerait un contenu fréquentiel tronqué pour $\rho < 1$.



La figure du haut montre les spectres de MIP maps pour les niveaux 0 et 1. En dessous, les figures montrent un ré-échantillonnage idéal et celui donné en interpolant le niveau 1 pour différentes valeurs de r. À gauche : $r = 3$ ($\rho = 3/2$) et à droite : $r = 3/2$ ($\rho = 3/4$).

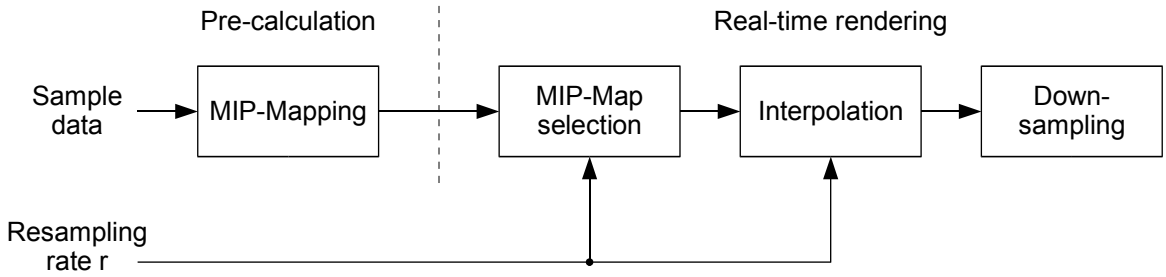
Dans notre méthode, l'interpolateur fonctionne à deux fois la fréquence d'échantillonnage de sortie, doublant la capacité de contenu fréquentiel. À partir de maintenant, nous appellerons ρ le taux utilisé pour interpoler un niveau donné d'une MIP map ; il dépend de ce niveau et du facteur global de ré-échantillonnage. En choisissant ainsi un niveau de MIP map de façon à obtenir $\rho \leq 1$, le principe de Shannon est respecté. Le repliement peut théoriquement être évité par un filtrage passe-bas pendant la décimation final. Et parce que la bande passante finale est la moitié de la bande passante du signale sur-échantillonné d'un facteur deux, choisir $\rho \geq 1/2$ ne tronque pas le contenu fréquentiel utile. De cette façon nous pouvons poser comme contrainte $\rho \in [1/2 ; 1]$. Il est alors suffisant d'espacer les niveaux de MIP maps d'une octave pour trouver un niveau et un ρ satisfaisant cette contrainte pour un taux r donné.



Interpolation à un sur-échantillonnage x2. La ligne en pointillé montre où se trouve la fréquence de Nyquist du signal de sortie.

L'interpolateur est un FIR. Pour des performances optimales, il a une bande passante fixe et sa fréquence de coupure ne baisse pas dans le cas $\rho > 1$, chose rendu superflue par le sur-échantillonnage, comme il a été montré précédemment. Ce FIR est stocké en mémoire de manière polyphase comme impulsion fenêtrée. Ses coefficients sont linéairement interpolés en temps réel pour produire un interpolateur final de haute précision.

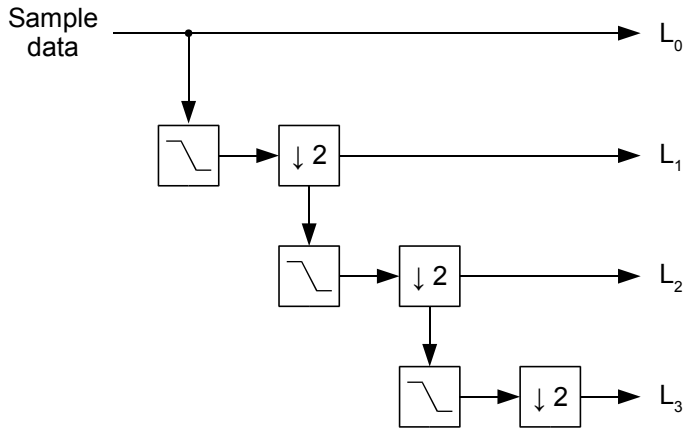
La dernière étape est le sous-échantillonnage x2. Nous utilisons une implémentation polyphase d'un filtre demi-bande IIR elliptique, décrit dans le document [9]. C'est la seule partie du traitement qui n'est pas linéaire en phase.



Vue d'ensemble du ré-échantillonnage.

3.2 Le MIP mapping

Le MIP mapping est la première étape du procédé. Pour une efficacité maximum, il doit être effectué off-line, une fois pour toute avant les traitements en temps-réel. Comme nous l'avons montré précédemment, le rendement optimal de notre système est atteint quand les niveaux de la MIP map sont espacés d'une octave. Le premier niveau est évidemment l'échantillon original. Pour obtenir les niveau suivant, nous filtrons le niveau courant avec un passe-bas demi-bande – c'est-à-dire dont la fréquence de coupure est la moitié de celle de Nyquist, et nous décimons le résultat par deux en rejetant un échantillon sur deux. De manière simple, on peut montrer que la mémoire totale nécessaire aux niveaux n'excède pas le double de celle utilisée par l'échantillon original (notons que cette assertion est en toute rigueur fausse, à vous de trouver pourquoi).



Cascade de filtres demi-bande et de decimateurs produisant les niveaux de la MIP-map

Il est important que le filtre demi-bande soit à phase linéaire. En effet, quand on passe d'un niveau de MIP map à un autre à cause du changement de taux de ré-échantillonnage, une différence de phase entre les deux niveaux causerait une discontinuité dans le signal. Ainsi, le filtrage par FIR est la méthode la plus simple pour traiter notre problème. Comme ce filtrage se fait une seule fois et en pré-calcul, nous pouvons utiliser un filtre coûteux, de très haute qualité. Cela nous permet de garder une certaine marge d'erreur pour les traitements en temps réel. Le filtre doit être conçu avec une bande de transition aussi fine que possible, une bande passante très plate et une bande rejetée un peu en dessous du rapport signal/bruit désiré. Plusieurs méthodes peuvent être utilisées pour la conception, tel que l'algorithme de Parks-McClellan [10] (aussi connu sous le nom de Remez) ou une impulsion sinc tronquée par une fenêtre Dolph-Chebyshev [7]. Une ou deux centaines de coefficients devraient faire l'affaire.

Comme l'ordre du filtre est particulièrement élevé, il est recommandé d'implémenter la convolution par FFT à l'aide des méthodes overlap-add ou overlap-save [11]. De plus, les problèmes de résolution des données doivent être pris en compte. En effet, stocker les échantillons d'un niveau avec la résolution minimum nécessaire et les utiliser pour générer le niveau suivant peut entraîner l'accumulation des erreurs d'arrondis, entraînant un bruit de calcul important. Pour palier à ceci, nous proposons de traiter tous les niveaux simultanément avec une résolution maximale, et de réduire la résolution des données une fois qu'elles ont été utilisées. D'ailleurs à ce stade, le tramage (*dithering*) peut aider à économiser quelques bits de stockage [12]. Cependant la mise en forme du bruit de quantification (*noise shaping*) n'est pas recommandée pour les raisons données dans le document [13].

Le filtrage par FIR induit un retard dans le signal. Ce retard doit être compensé soit en tronquant le début de chaque niveau de MIP map, soit en ré-indexant les tableaux d'échantillons, de façon à garder une référence temporelle uniforme entre les niveaux.

Lors de la lecture du son, l'index du niveau MIP map peut être défini par :

$$l(r) = max(0,\ floor(\log_2(r))) \qquad (3.2\text{-}1)$$

Où r est le facteur de ré-échantillonnage et $floor(x)$ dénote le plus grand entier inférieur ou égal à x – la fonction seuil. 0 est l'index du niveau constitué de l'échantillon original, 1 est sous-échantillonné par un facteur 2, 2 est sous-échantillonné par un facteur 4, etc. À partir de cela, nous pouvons déduire le facteur de ré-échantillonnage local ρ :

$$\rho = \frac{r}{q \cdot 2^{l(r)}} \qquad (3.2\text{-}2)$$

Où q est le facteur de sur-échantillonnage de l'interpolateur, nous l'avons fixé à 2. Nous pouvons également déduire le nombre de niveaux à générer en fonction de la valeur maximale que peut prendre r.

Dans le cas extrême où le nombre de niveaux est si important que le dernier se réduit à un simple échantillon, le MIP mapping n'occupe que deux fois la place prise par l'échantillon original. En effet :

$$S^n = S + \frac{S}{2} + \frac{S}{4} + \frac{S}{8} + \ldots = S \sum_{k=0}^{+\infty} \left(\frac{1}{2}\right)^k = S \frac{1}{1 - \frac{1}{2}} = 2S \quad \text{(3.2-3)}$$

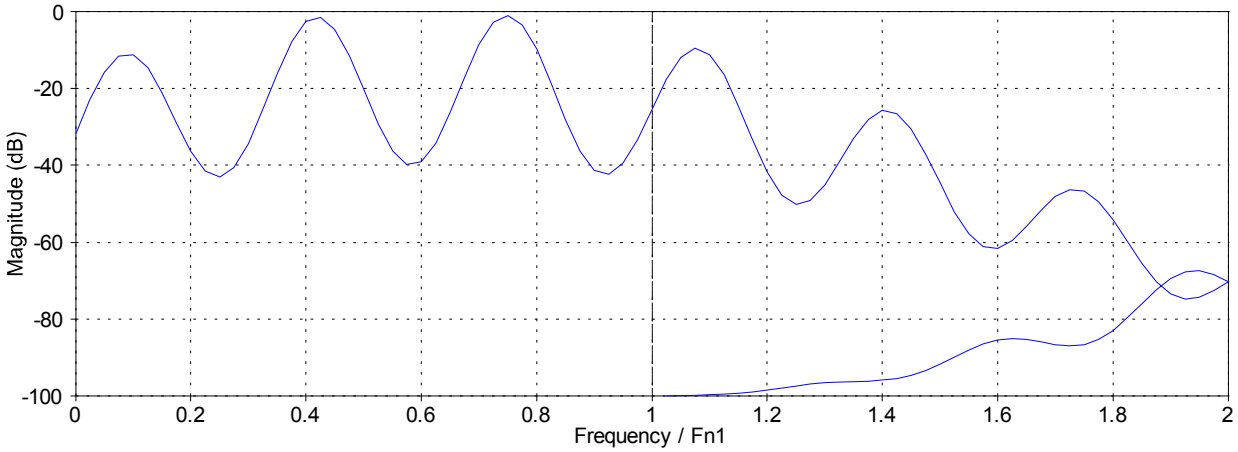
Dans les cas d'utilisation courante, et avec l'arrivée des banques de sons d'instruments *multi-layer*, l'algorithme n'a pas besoin d'atteindre des taux de ré-échantillonnage aussi élevés. La plupart du temps, seuls deux niveaux sont requis (0 et 1), occupant en mémoire une fois et demie la place prise par l'échantillon original.

3.3 Conception de l'interpolateur

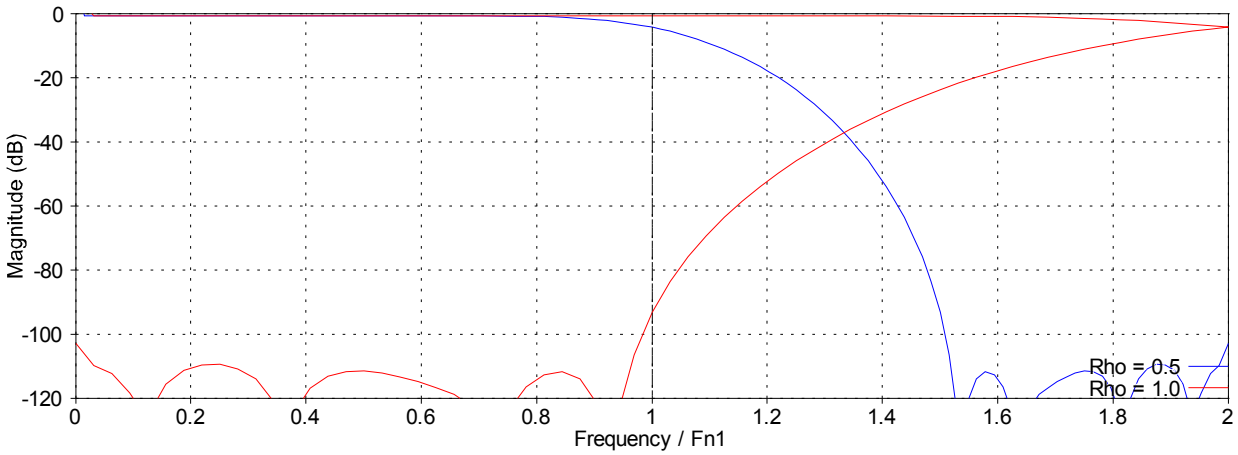
Nous avons choisi d'implémenter l'interpolateur sous forme d'un FIR à cause de la flexibilité qu'offre ce type de filtre en terme de réponse fréquentielle. De plus, les jeux d'instruction SIMD des CPU modernes peuvent améliorer sensiblement la vitesse de calcul en exécutant plusieurs multiplications et additions simultanément. Cette amélioration est beaucoup plus nette avec les FIR qu'avec les interpolateurs polynomiaux.

Contrairement à de nombreuses réalisations d'interpolateurs sous forme FIR, notre implémentation a une fréquence de coupure fixe. Plus précisément, le terme passe-bas n'est pas très approprié car la fréquence de coupure est fixée à la fréquence de Nyquist de l'échantillon source (en fait celle du niveau de MIP map utilisé). En effet, nous avons besoin de préserver l'intégralité du spectre. À cause du sur-échantillonnage dans lequel fonctionne l'interpolateur, le contenu de l'échantillon interpolé ne dépassera jamais la fréquence de Nyquist du son final (F_{N1}). Ici, le but est de reconstruire de manière exacte le signal source sous sa forme continue et de l'échantillonner au taux relatif ρ .

Notons que $\rho \leq 3/2$ est théoriquement la borne supérieure pour éviter le repliement spectral en sur-échantillonnage 2x. Avec cette hypothèse plus forte que celle que nous avons initialement envisagée, le contenu fréquentiel le plus haut atteint virtuellement $3F_{N2}/2$ et se replie sur $F_{N2}/2$. Ainsi, la partie repliée du signal sur-échantillonné reste toujours au-dessus de $F_{N2}/2 = F_{N1}$ et sera complètement écartée lors de l'étape suivante de sous-échantillonnage. Ou inversement, on peut dire que $3/2$ est un facteur suffisant de sur-échantillonnage pour utiliser des niveaux de MIP map espacés d'une octave. Cependant, nous avons choisi $\rho \leq 1$ et allons utiliser cette marge pour la bande de transition de l'interpolateur, de façon à ce qu'elle puisse s'étendre jusqu'à $3F_{N2}/2$ sans effet audible. De plus, le sur-échantillonnage x $3/2$ demande un nombre fractionnaire d'échantillons source pour constituer un échantillon de sortie, ce qui peut compliquer un peu les choses si nous devons produire la sortie sous forme de blocs de taille impaire. Ce problème n'apparaît pas avec les sur-échantillonnage x2.



Interpolation idéale à $\rho = 3/2$. Le repliement s'arrête au-dessus de F_{N1} .



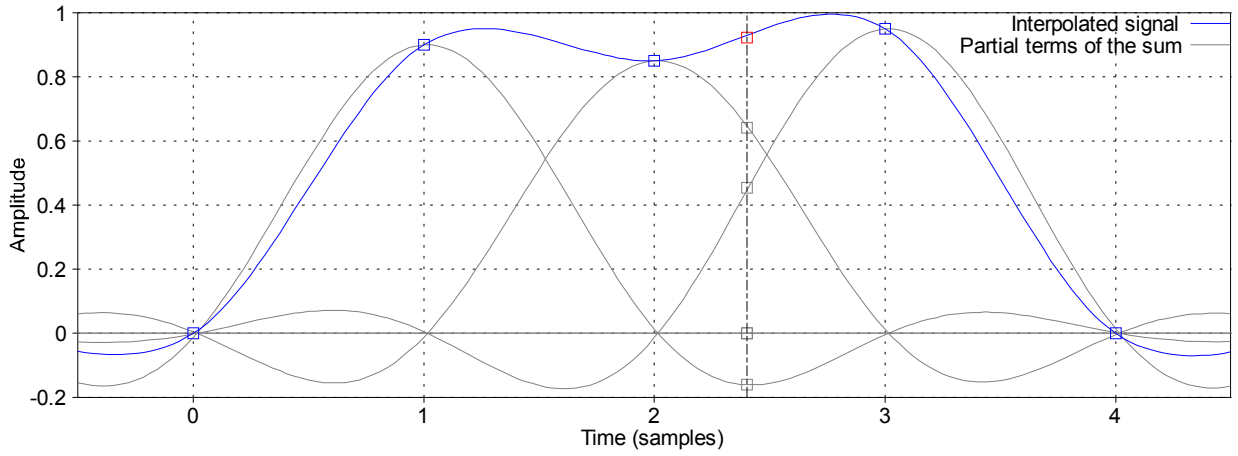
Réponse d'un interpolateur réel à $\rho=1$ et utilisation de la marge sur le facteur de ré-échantillonnage pour placer la bande de transition.

Ainsi le FIR est assimilé à un retard fractionnaire censé reconstituer les données de l'échantillon pour n'importe quelle position fractionnaire. Nous posons :

- $p=\text{floor}(t)$ et $d=t-p$ les positions respectivement entières et fractionnaires d'une position $t=p+d$ dans l'échantillon source. Évidemment, p est entier et $d\in[0\,;1[$,
- N un entier pair positif représentant la longueur de l'impulsion du FIR — approximativement le nombre de croisements de sa courbe avec 0,
- $G_C(t)$ la réponse impulsionnelle de l'interpolateur, centré sur $N/2$ et définie dans l'intervalle $[0\,;N[$,
- $x[p]$ les échantillons sources pour le niveau de MIP map utilisé, et
- $x_C(t)$ la fonction continue correspondant aux données interpolées.

Nous obtenons :

$$x_C(p+d)=\sum_{k=1}^{N} x\left[p+k-\frac{N}{2}\right]\cdot G_C(N-k+d) \quad (3.3\text{-}1)$$



Interpolation des échantillons pour $\{p,d\}=\{2,0.4\}$.
La courbe grise montre les contributions des échantillons sources à la somme donnant l'échantillon résultat.

$G_C(t)$ Est une fonction continue. En fait, nous pouvons la considérer comme un ensemble de fonctions discrètes dépendant d'un paramètre continu d :

$$G_d[k]=G_C(k+d) \quad (3.3\text{-}2)$$

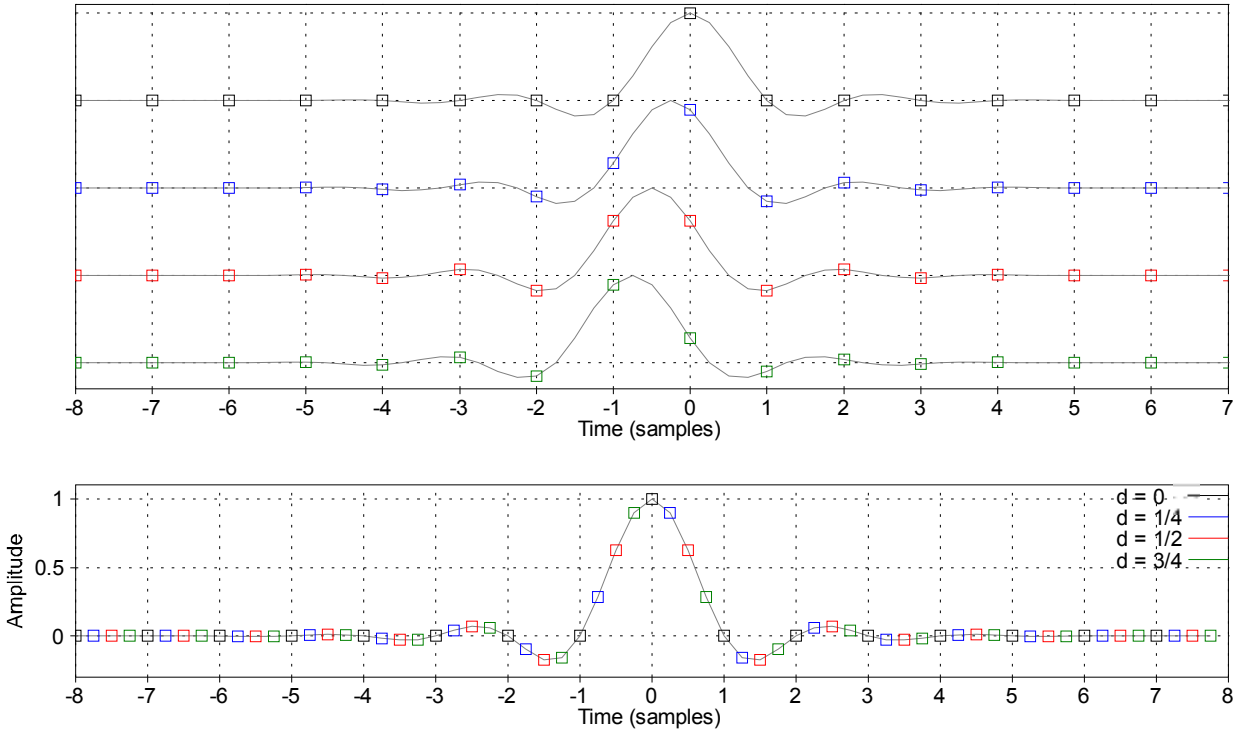
Ainsi nous utilisons une seule fonction $G_d[k]$ pour calculer la valeur à une position donnée. La formule (3.3-1) devient :

$$x_C(p+d)=\sum_{k=1}^{N} x\left[p+k-\frac{N}{2}\right]\cdot G_d[N-k] \quad (3.3\text{-}3)$$

La variable d est continue. Dans une implémentation concrète, nous ne pouvons pas stocker en mémoire un nombre infini de fonctions $G_d[k]$. Dans une implémentation pratique, nous quantifions d et lui assignons deux paramètres de base :

- Le nombre de coefficients par phase. Pour une position fractionnaire donnée, une phase du FIR est sélectionnée et convoluée avec les échantillons source. C'est le paramètre N mentionné précédemment.
- Le nombre M de phases, équidistantes dans l'intervalle $[0\ ;1[$.

Le paramètre N impacte directement sur la raideur de la courbe de réponse fréquentielle dans la bande de transition, ainsi que sur le niveau des oscillations dans les bandes passantes et bloquées, comme lors de la conception de n'importe quel FIR statique. Quand toutes les phases sont entrelacées et classées par date croissante, on obtient une impulsion globale dont la fréquence de coupure est F_N/M (fréquence relative au niveau de MIP-map).

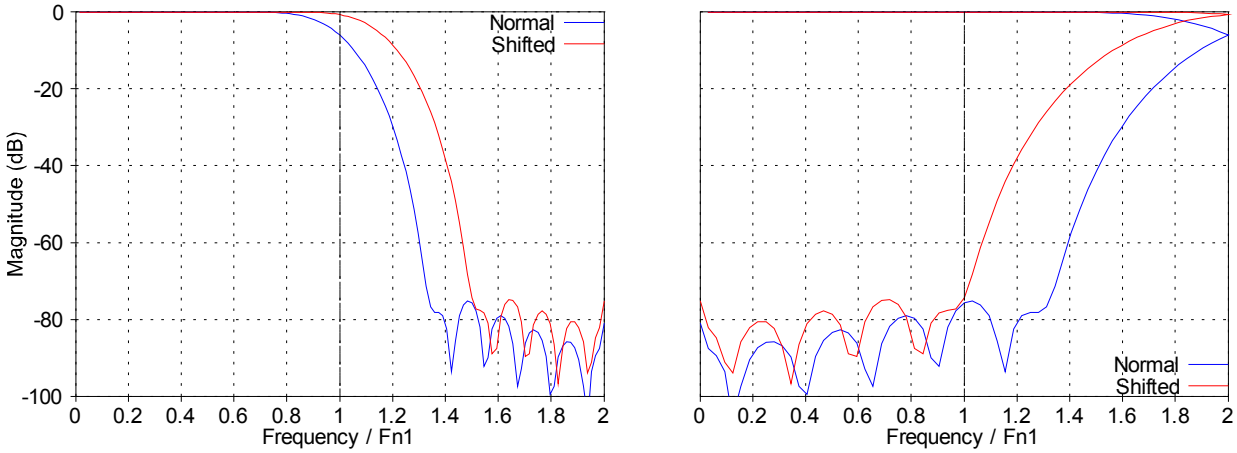


En haut : toutes les courbes $G_d[k]$ pour $M=4$. En bas, les impulsions entrelacées.

Donc nous allons maintenant nous concentrer sur la conception d'un filtre passe-bas de longueur $N\times M$. Il y a plusieurs méthodes. La plus performante est probablement l'algorithme de Parks-McClellan, mais la plus simple est le sinc fenêtré et marche bien pour les FIR de grande longueur. La fenêtre de Dolph-Chebyshev offre des oscillations de niveau constant dans la bande coupée et de bonnes propriétés pour la bande de transition. Notons que ce que nous obtenons avec les spécifications a une atténuation garantie en bande coupée, mais dans les cas concrets, on peut obtenir un peu trop par rapport à ce qui était nécessaire. Si on veut améliorer cette phase, il faudrait vérifier la réponse finale avec une FFT et ajuster récursivement les spécifications de la bande coupée pour optimiser la bande de transition.

Une autre chose à considérer : nous avons laissé de la marge de bande passante en contraignant le taux de sur-échantillonnage à 2 et l'intervalle de valeurs de ρ. Cela permet au

lobe principal du filtre de s'étendre jusqu'à $3F_{N2}/2$. Dans le cas du sinc fenêtré, nous pouvons légèrement remonter la fréquence de coupure de façon à ce que le lobe principal touche cette fréquence. Ainsi nous maximisons la platitude de la bande passante.



Mise à profit de la marge pour améliorer la platitude de la réponse en bande passante.
Figure de gauche : $\rho=1/2$, figure de droite : $\rho=1$.

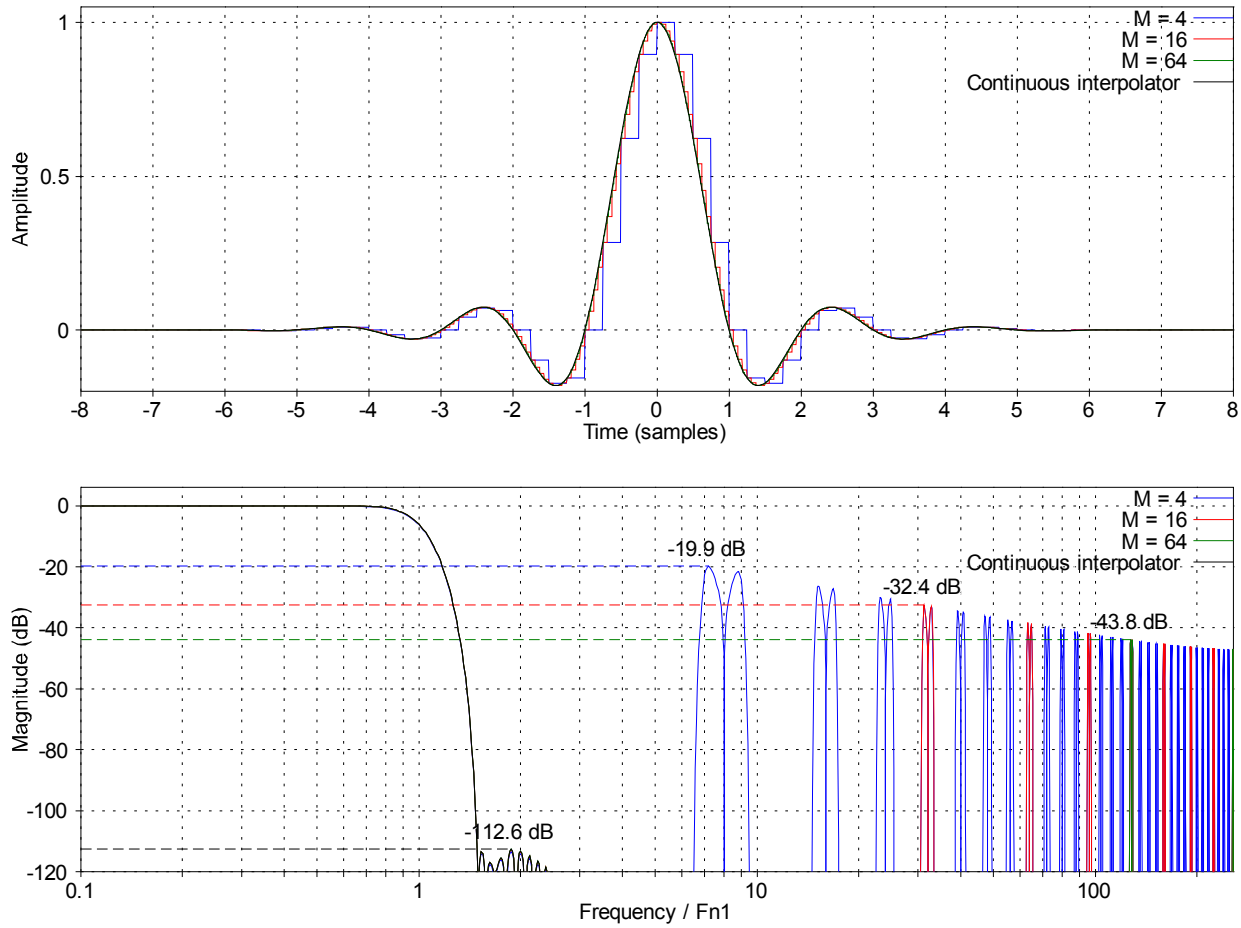
Il est encore possible d'améliorer le système en considérant que la bande passante ne va pas exactement jusqu'à la fréquence de Nyquist. Des spécifications réalistes incluraient effectivement une petite marge entre la fin de la bande passante et la fréquence de Nyquist. En effet, la zone la plus haute du spectre est généralement encombrée de signaux parasites, venant du fait que les filtres anti-repliement ne sont et ne peuvent pas être parfaits. Nous posons F_{CP} la fréquence supérieure limitant la bande passante, comme fraction de la fréquence de coupure, ici F_{N1}. En prenant en compte le repliement pour $\rho=1$, nous pouvons calculer la fréquence inférieure optimale pour la bande coupée :

$$F_{CS}=\frac{4-F_{CP}}{2} \quad (3.3\text{-}4)$$

Avec $F_{CP}=0.9$, qui est une spécification courante, nous obtenons $F_{CS}=1.55$. Si l'algorithme de Parks-McClellan est utilisé, ces valeurs peuvent être utilisées pour délimiter les bandes passantes et coupées.

3.4 Interpoler l'interpolateur

Évidemment, quantifier le nombre de fonctions G est équivalent à quantifier la position fractionnaire d. Selon le document [3], la réduction du nombre de phases (M) dégrade le SNR à un taux de 6 dB/bit. La figure ci-dessous montre les réponses comparées de tels interpolateurs quantifiés, avec $N=16$ et différentes valeurs pour M :



Réponses impulsionnelles d'interpolateurs discrets et continus

Nous voyons sur les réponses en fréquence une séparation claire entre les versions discrètes et continues de l'interpolateur. Les lobes latéraux de l'interpolateur continu sont causés uniquement par le fenêtrage de l'impulsion, alors que ceux de l'interpolateur discret ont pour origine la réduction en bits de la phase. Dans ces conditions, nous devons augmenter M à une valeur inacceptable pour obtenir un SNR décent. C'est pourquoi il est nécessaire d'interpoler correctement les fonctions $G_d[k]$. Les différentes phases constituant la totalité de l'impulsion peuvent être numérotées de 0 à $M-1$. Nous introduisons alors la variable entière q :

$$q=\text{floor}(d\cdot M) \quad (3.4\text{-}1)$$

La façon dont nous arrondissons $d\cdot M$ n'est probablement pas la plus appropriée pour utiliser directement $G_{q/M}[k]$ — il aurait été mieux d'arrondir à l'entier le plus proche, divisant par deux l'erreur de phase maximum ce qui aurait permis de gagner 6 dB de SNR. Mais comme notre choix s'est porté sur l'interpolation linéaire, l'arrondi par défaut est une décision logique. Finalement, nous définissons la fonction d'interpolation $G_{Id}[k]$ comme :

$$G_{Id}[k]=G_{\frac{q}{M}}[k]+(d\cdot M-q)\left(G_{\frac{q+1}{M}}[k]-G_{\frac{q}{M}}[k]\right) \quad (3.4\text{-}2)$$

La formule (3.4-2) utilise $G_1[k]$, qui n'est pas défini en 1. Pour pour assurer la cohérence de (3.4-2), nous étendons l'intervalle de la classe de fonctions G selon la définition (3.3-2) :

$$G_1[k]=G_C(k+1) \quad (3.4\text{-}3)$$

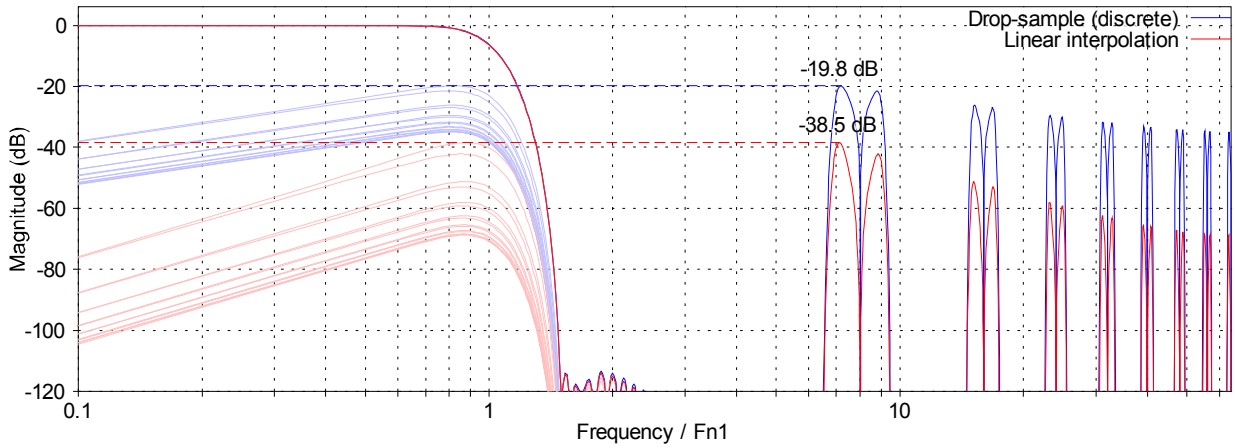
Ainsi :

$$G_1[k]=G_0[k+1] \quad (3.4\text{-}4)$$

Cette classe de fonctions $G_{Id}[k]$ nous donne la possibilité de définir une nouvelle fonction continue pour l'interpolateur :

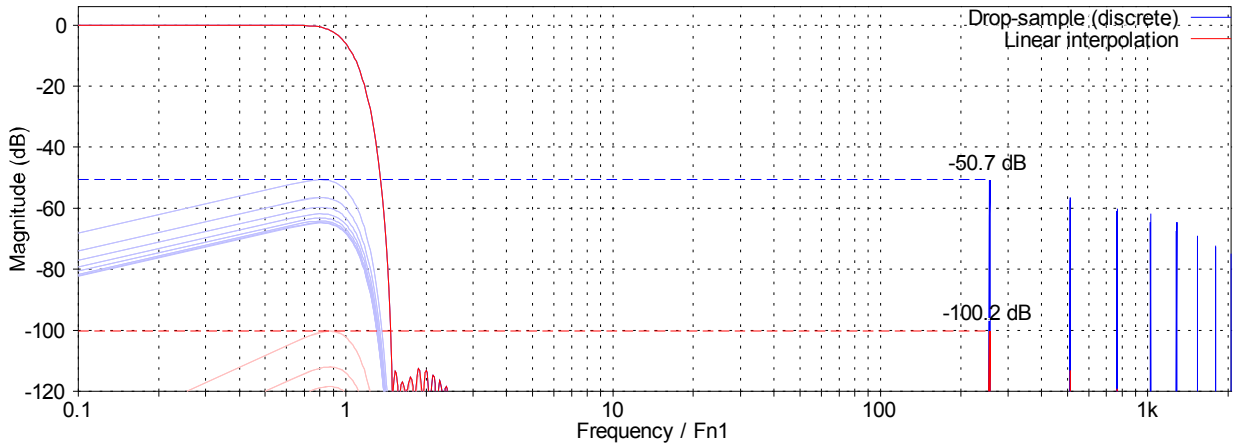
$G_{C2}(k+d)=G_{Id}[k]$ (3.4-5)

Ci-dessous, $G_{C2}(t)$ pour $N=16$ et $M=4$. Nous évaluons sa réponse en fréquence et la comparons à celle de l'interpolateur quantifié. Évidemment $M=4$ est trop petit pour une application réelle, le but ici est plutôt de montrer clairement le principe et ses bénéfices.



Réponse en fréquences des $G_{C2}(t)$ est de ses versions discrètes pour $N=16$ et $M=4$. Les couleurs délavées montrent les effets du repliement des lobes latéraux.

Avec $M=128$, le repliement est rejeté dans les profondeurs abyssales du -100 dB :



Les réponses en fréquence de $G_{C2}(t)$ et ses versions discrètes pour $N=16$ et $M=128$.

Ce nouvel algorithme d'interpolation est équivalent à un interpolateur polynomial travaillant sur un échantillon source sur-échantillonné, où M est le facteur de sur-échantillonnage. Si vous avez besoin d'un repliement extrêmement faible et que vous vous souciez plus de la mémoire que du coût de calcul, l'interpolation linéaire peut être trop juste pour parvenir à cette fin. Référez-vous aux documents [1] et [2] pour une discussion plus approfondie sur ce sujet particulier.

3.5 Le cas r < 1

Nous avons essentiellement traité du repliement causé par le cas où r > 1. Nous avons profité du sur-échantillonnage pour abaisser nos exigences sur la largeur de la bande de transition du filtre d'interpolation. En l'associant au MIP mapping, nous avons pu obtenir un filtre à pente très raide pour un faible coût en ressources temps-réel.

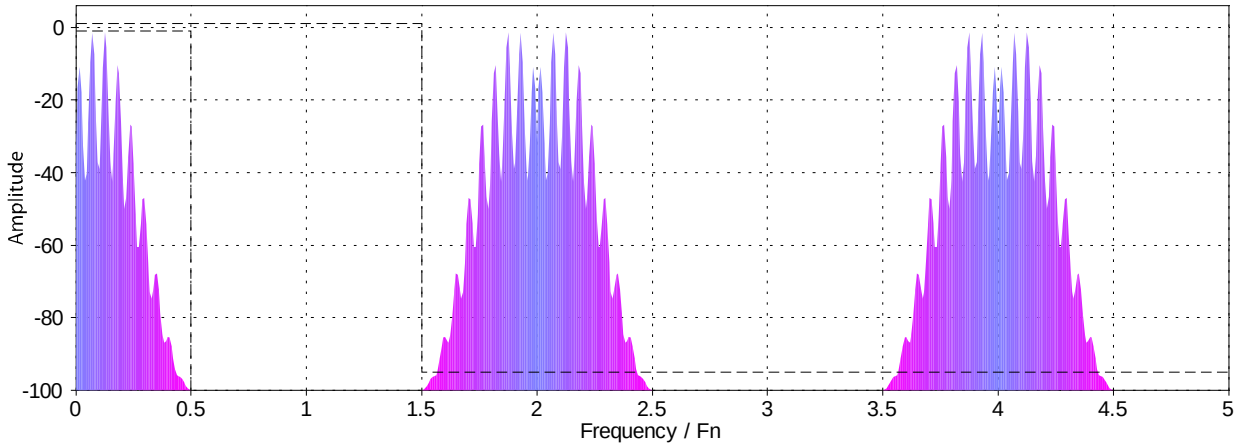
Cependant les choses sont un peu différentes quand r est inférieur à 1. En effet, la pente du filtre d'interpolation n'est pas aussi raide qu'elle le devrait pour annuler totalement les réplications spectrales. La partie supérieure du spectre, qui est naturellement reflétée et répliquée au-dessus de $r \cdot F_N$, n'est pas assez filtrée. De fait, nous devons utiliser un filtre plus raide. Nous avons deux solutions : le post-filtrage du signal signal avec un filtre IIR passe-bas efficace (elliptique par exemple), ou un filtre d'interpolation plus coriace de façon à affiner la bande de transition.

La première option est très attrayante. Les filtres elliptiques sont assez peu coûteux ; nous pouvons atteindre une pente très raide pour un nombre de MACs relativement faible. Mais il y a des inconvénients : la perte de la continuité des phases avec le cas $r>1$, les problèmes avec les fréquences de coupure proches de Nyquist et une structure récursive rédhibitoire pour la parallélisation des opérations. Aucun de ces problèmes ne sont insolvables, mais ils ajoutent une complexité significative à la solution.

La second option peut paraître au premier abord outrageusement gourmande en calculs. Ce n'est pas tout à fait exact. Avec $r<1$, nous ne sommes plus confrontés au repliement du spectre original se manifestant quelle que soit la perfection du filtre d'interpolation. Ici, le repliement aurait été causé par un filtre imparfait, laissant passer trop de reflets du spectre, certains dépassant alors Nyquist. Si la pente du filtre est suffisamment raide, nous n'avons pas besoin du sur-échantillonnage. Nous pouvons ainsi économiser la moitié du temps de calcul. Ce temps peut être réinvesti dans la raideur du filtre dont nous avons besoin.

Si la solution est simple et relativement efficace, elle est loin d'être parfaite. En effet, la perfection de notre filtre a ses propres limites. Avec deux fois plus de coefficients, nous ne pouvons réduire la bande de transition que d'un facteur deux environ, pas beaucoup plus. Cette solution est acceptable pourvu que nous ayons spécifié une bande de transition assez large, ou si nous sommes prêts à faire un compromis sur les oscillations en bande passante ou coupée.

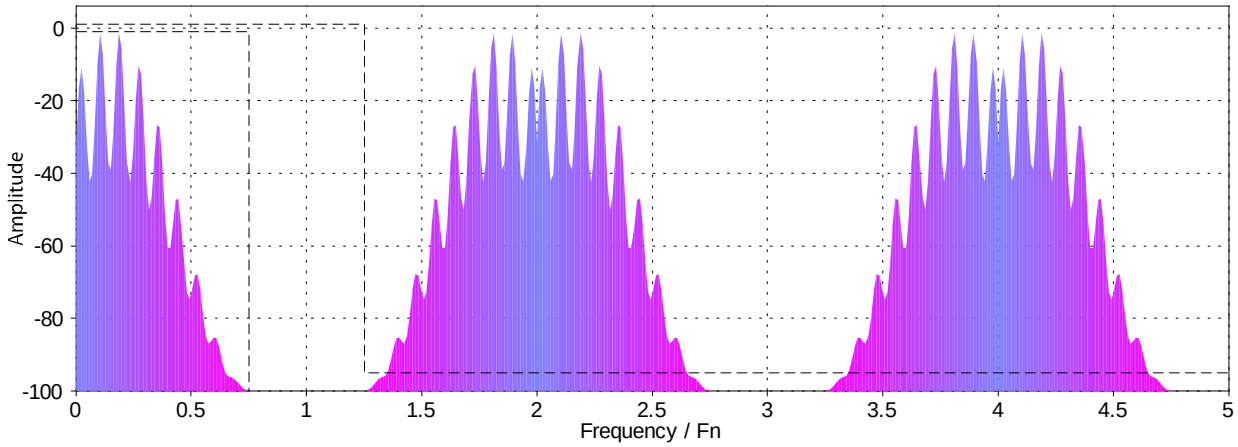
Cependant nous avons un autre atout dans notre jeu, un compromis supplémentaire. Il s'agit de la mémoire. Les spécifications de notre interpolateur sont relatives au signal original, pas directement à son contenu fréquentiel. Si ce dernier n'occupe que la moitié de la bande passante, le spectre du résultat interpolé aura des « trous » correspondant à l'absence de contenu dans le signal original. En sur-échantillonnant à l'avance l'échantillon, nous pouvons réduire les zones dans lesquelles le filtre doit réellement filtrer quelque chose, obtenant ainsi de meilleurs interpolateurs. Les articles [1] et [2] traitent de ce sujet à travers l'interpolation polynomiale, mais nous pouvons appliquer cette théorie aux interpolateurs FIR.



Spécifications d'un filtre d'interpolation pour des données pré-sur-échantillonnées x2.

Par quel facteur devons nous sur-échantillonner la source ? Pas beaucoup. Pour un contenu spectral identique, chaque augmentation de la fréquence d'échantillonnage bénéficie immédiatement à la bande de transition du filtre. Le sur-échantillonnage double est alors largement suffisant. Si l'occupation mémoire est un problème crucial, nous pouvons réduire le facteur de sur-échantillonnage à $4/3$. En effet, le filtre conçu pour $r>1$ a une bande de

transition dont la largeur vaut approximativement $F_N/2$. C'est la même largeur que la bande libre laissée par la réplication du spectre après l'interpolation, entre $3F_N/4$ et $5F_N/4$.



Spécifications d'un filtre d'interpolation pour des données pré-sur-échantillonnées x $4/3$.

Nous ajoutons ainsi un nouveau niveau MIP map, ici sur-échantillonné et non sous-échantillonné comme nous les avons utilisés précédemment. Nous pourrions être tentés d'utiliser des niveaux de MIP map sur-échantillonnés partout, en particulier pour les cas $r>1$. Mais en sur-échantillonnant la source, nous devons utiliser des valeurs de ρ de plus en plus élevées, ce qui compense en fin de compte les gains escomptés sur la bande de transition.

Nous pouvons bien sûr mentionner une troisième possibilité : ne rien changer à l'algorithme. Il est possible de vivre avec la réplique d'un bout du haut du spectre. Avec notre modèle de filtre, cela donne la possibilité de se débarrasser du sur-échantillonnage en temps réel dès que $r<2/3$; théoriquement le contenu filtré passe sous la limite de repliement.

3.6 Sous-échantillonnage final

La dernière chose que nous devons faire est de passer du sur-échantillonnage double à la fréquence d'échantillonnage finale. Nous utiliserons la structure de filtre IIR polyphase décrite dans le document [9]. La réponse en magnitude de cette structure est impressionnante, comparée au temps de calcul nécessaire. La distortion de phase est jugée peu importante ici, mais quiconque a besoin d'une phase linéaire peut utiliser des filtres polyphase classiques FIR. Ceux-ci ont été abondamment abordés dans la littérature, en particulier dans le document [4].

Théoriquement, nous n'avons plus de marge pour la bande de transition de notre filtre. Cependant, un signal sonore n'est jamais échantillonné de manière critique, c'est-à-dire que la fréquence de Nyquist et toujours un peu au-dessus de la dernière fréquence utile. Celle-ci peut arbitrairement être choisie à 20 kHz, limite communément admise pour les signaux audibles. La bande de fréquence restante peut alors être altérée sans effet audible notable. Par exemple, une fréquence d'échantillonnage de 44.1 kHz nous laisse une marge de 2.05 x 2 = 4.1 kHz pour la transition entre la bande passante et la bande coupée. Notons que cette marge peut également être utilisée pour élargir un peu les bandes de transition des filtres d'interpolation et de création des niveaux de MIP-map.

4. Implémentation

4.1 Un exemple complet

4.1.1 Cahier des charges

Dans cette section, nous allons concevoir un interpolateur construit sur les spécifications suivantes, en ce qui concerne la qualité sonore :

- Bande passante définie de 0 à $\frac{9}{10} F_{NI}$ Hz
- Atténuation du repliement dans la bande passante en-dessous de -85 dB
- Oscillations de la réponse de la bande passante en-dessous de 0.1 dB d'amplitude

Évidemment, nous pouvons aussi ajouter ces contraintes floues :

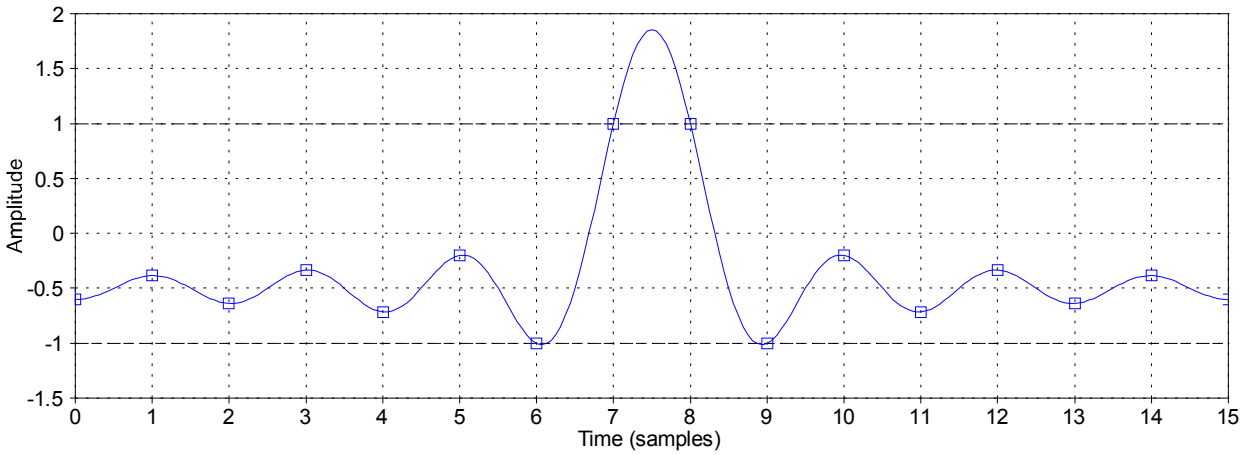
- Faible usage de la mémoire
- Faible temps de calcul CPU
- Rendu en temps réel
- Taux de ré-échantillonnage variable dans le temps

À une fréquence d'échantillonnage de 44.1 kHz, la bande passante s'étend jusqu'à 19.8 kHz, ce qui est bien suffisant pour des applications de type audio. Nous n'allons pas donner une formule magique pour produire le meilleur interpolateur possible pour une spécification donnée. Il est préférable de procéder par essais et erreurs. Cela marche bien dans le cadre de la méthode décrite précédemment. Le résultat ne sera probablement pas optimal, mais aura une implémentation simple et efficace.

4.1.2 MIP maps et stockage des données

Comme suggéré dans la méthode décrite précédemment, nous travaillerons avec des MIP map dont les niveaux successifs sont espacés d'une octave. Le filtre de ré-échantillonnage fonctionnera au double de la fréquence d'échantillonnage finale.

Le type de stockage des données des niveaux de MIP map dépend de l'échantillon source. Dans le cas d'entiers sur 16 bits, il n'y a pas un réel besoin de passer en 32 bits ou en virgule flottante car nos spécifications de SNR sont assez inférieures à celle permise par une résolution de 16 bits. Cependant nous devons faire attention. Si l'amplitude des données originales atteint les bornes de l'intervalle permis par les 16 bits, le son ré-échantillonné va très probablement dépasser cet intervalle. Prenons le cas qui est probablement le pire de ce point de vue : une fonction $sinc$ centrée exactement entre deux échantillons et tenant pile dans la plage de représentation des données. Une interpolation parfaite produirait d'importantes oscillations du côté positif (du côté négatif aussi, mais moins), atteignant une valeur proche de 2 ! Il est donc recommandé de garder au moins 6 dB de marge pour ce genre de problème, ou plus simplement, d'augmenter la capacité de stockage finale d'un bit par échantillon. Cette considération vaut aussi bien pour le calcul de l'interpolation en temps réel que pour le pré-calcul des MIP maps.



Marge nécessaire pour l'interpolation.

4.1.3 Conception du filtre des MIP maps

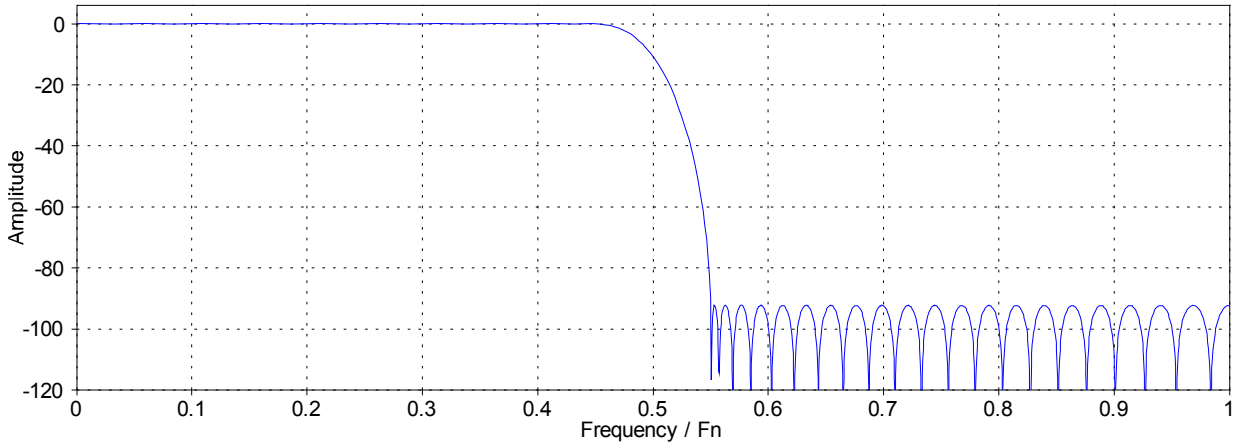
Nous utiliserons l'algorithme de Parks-McClellan pour concevoir le filtre demi-bande ($F_C = F_N/2$) en respectant les spécifications de réponse en fréquence suivantes :

- 1 entre 0 et $\frac{9}{10} F_C$, la bande passante.
- 0 entre $\frac{11}{10} F_C$ et F_N, la bande coupée.
- Un rapport de 100 entre les oscillations en bande passantes et celles en bande coupée.

Nous obtenons les caractéristiques suivantes :

- Nombre de coefficients : 81
- Largeur de la bande de transition : $F_N/10$
- Oscillations en bande passante : 0.04 dB
- Atténuation de la bande coupée : -92.2 dB
- Retard de groupe général : 40 échantillons (phase linéaire)

Les oscillations sont plus faibles que ce qui était requis parce que nous devons prendre en compte la pollution de la bande passante causée par les traitement suivants.



Réponse en fréquence du filtre MIP map

Lors du filtrage de l'échantillon, nous effectuerons la convolution par l'intermédiaire d'une FFT (algorithme overlap-add). Ainsi nous pourrons minimiser le temps d'initialisation. Nous

devons utiliser une FFT de longueur supérieure à 81. Avec 128, nous ne pouvons traiter que 48 échantillons par passe, ce qui n'est pas optimal en terme de performance. Une FFT de 256 points semble plus appropriée. Si l'algorithme de FFT utilisé est disponible en radix-3, 192 points conviennent à merveille. Le filtre peut être stocké dans le domaine fréquentiel afin d'être immédiatement disponible pour la multiplication avec les données fréquentielles du son à traiter. Encore une fois, nous devons faire une remarque sur la résolution utilisée lors des calculs. Une FFT en entiers 16 bits ne sera pas suffisante, il est nécessaire de tout calculer au moins en entiers 24 bits ou en virgule flottante 32 bits, et de revenir en 16 bits au dernier moment. La réduction de bits peut se faire avec un simple tramage à bruit à densité de probabilité triangulaire.

4.1.4 Conception du filtre d'interpolation pour r > 1

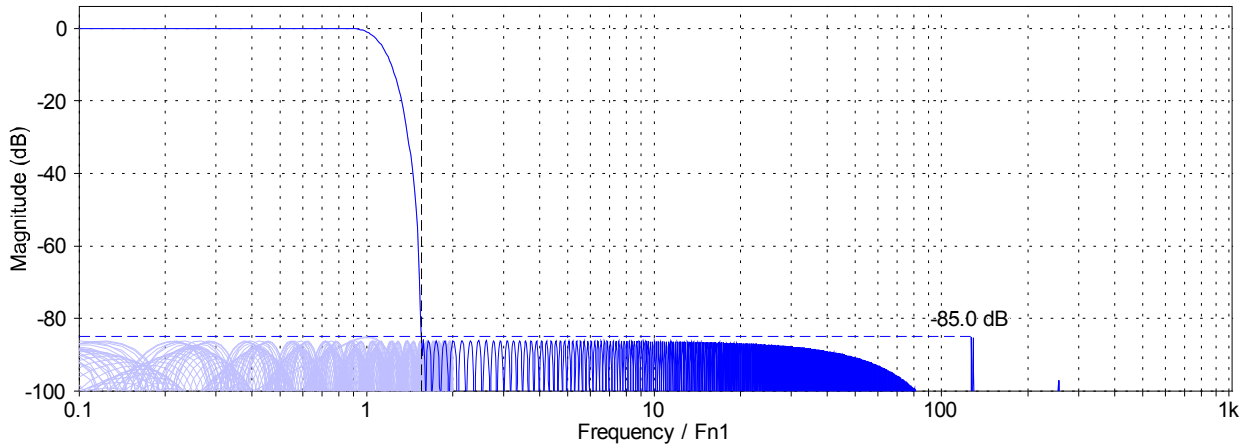
Nous devons maintenant définir les paramètres les plus importants, N et M. N influence la qualité générale de l'interpolateur – plus précisément la bande de transition. M régule le niveau du repliement des images spectrales causé par l'interpolation. Après quelques essais, nous avons trouvé que la combinaison $N=12$ et $M=64$ convient parfaitement à nos exigences.

Nous utiliserons une fois de plus l'algorithme de Parks-McClellan pour concevoir le filtre d'interpolation dont la fréquence de coupure vaut $F_C=F_N/M$. Nous déduisons de l'équation (3.3-4) les spécifications du filtre et injectons dans l'algorithme le prototype suivant :

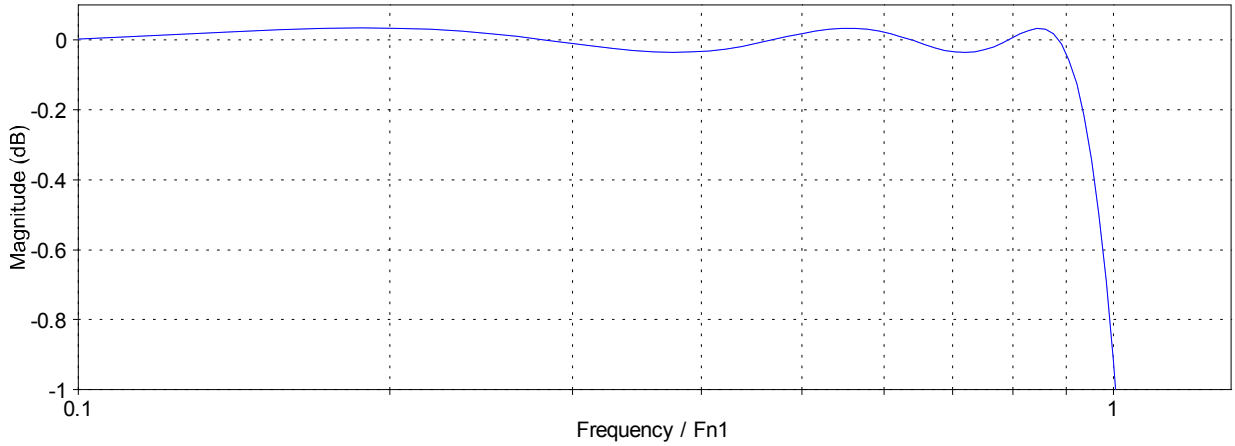
- Nombre de coefficients : $N\cdot M-1=767$
- La bande passante entre le fréquence 0 et $9F_C/10$.
- La bande coupée entre $31F_C/20$ et F_N.
- Les niveaux d'oscillations en bande passante et en bande coupée sont respectivement -0.08 dB et -85 dB.

Nous spécifions seulement 767 coefficients à l'algorithme. Ce détail est motivé par un problème de phase : il est plus facile de manipuler une filtre dont le nombre de coefficient est impair parce que le retard de groupe est alors un nombre entier d'échantillons. Il est possible d'atteindre le même résultat avec un filtre dont le nombre de coefficients serait pair, mais cela entraîne une manipulation supplémentaire de la phase durant l'interpolation pour que les transitions de MIP maps restent fluides. The resulting filter has these properties:

- Bande de transition entre $9F_C/10$ et $31F_C/20$
- Oscillations en bande passante : 0.08 dB
- Oscillations en bande coupée : -85.0 dB
- Retard de groupe : 384 échantillons (phase linéaire) sur le filtre désentrelacé, ce qui donne 6 échantillons pour chaque fonctions $G_d[k]$.



Réponse en fréquence obtenue avec interpolation linéaire du filtre.

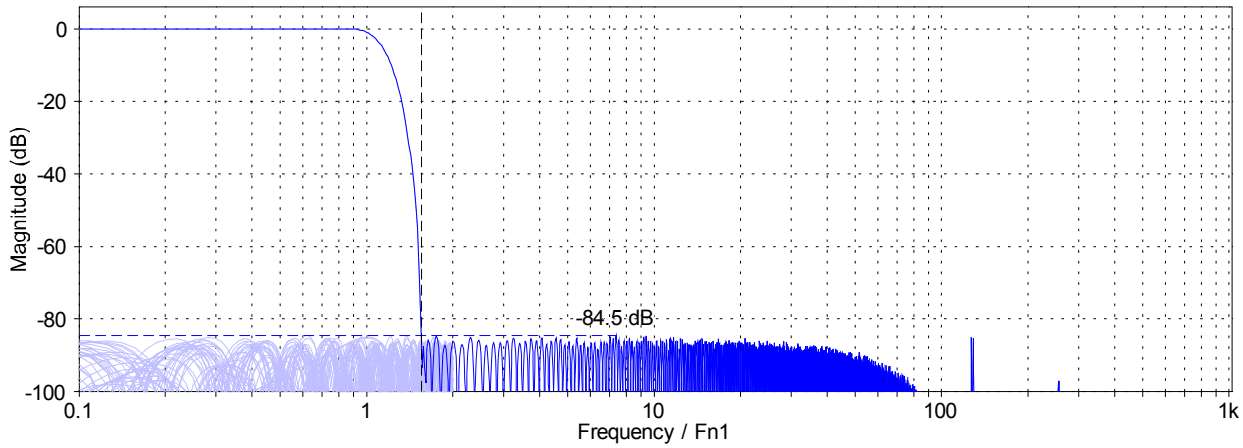


Zoom sur la bande passante.

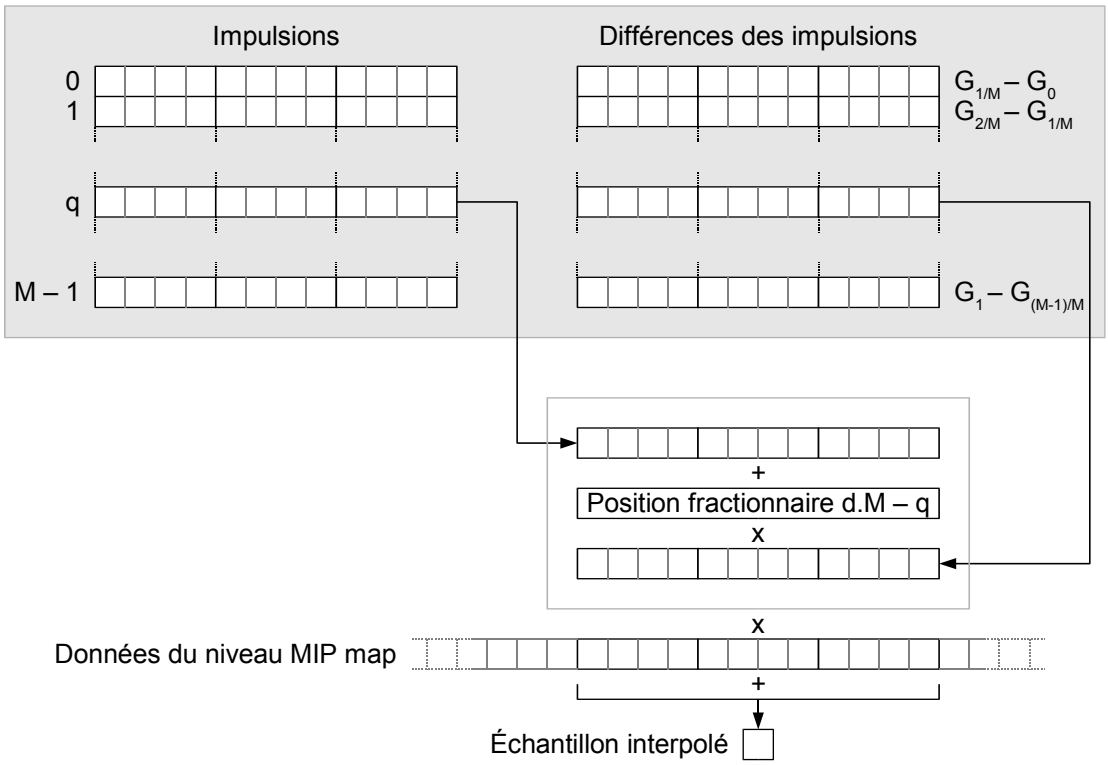
Maintenant, nous devons nous occuper du stockage en mémoire du filtre. Une méthode efficace consiste à le stocker sous forme de fonctions $G_d[k]$ désentrelacées. Chacune représente l'impulsion d'un filtre fonctionnant à la fréquence d'échantillonnage de traitement. Ces impulsions peuvent être lues en mémoire et associées en bloc aux données des MIP maps en utilisant le jeu d'instruction SIMD du processeur. Comme nous devons effectuer une interpolation linéaire — équation (3.4-2), il est avisé de précalculer les différences entre deux fonctions $G_d[k]$ consécutives. La dernière fonction, correspondant à $G_1[k]$, est obtneue en décalant $G_0[k]$ d'un échantillon, comme indiqué dans l'équation (3.4-4), et complétée par un 0 à la fin.

Nous stockons les tables en intercalant les fonctions-différences entre les fonctions-impulsions de façon à avoir une lecture séquentielle des données, ce qui utilise avantageusement le mode « burst » des contrôleurs mémoire. En effet la plupart des DAW font tourner divers effets et instruments logiciels en plus des échantillonneurs. Les éléments du réseau audio ainsi constitué sont traités cycliquement, les uns après les autres. Chaque élément traite un bloc de N échantillons, qu'il passe aux suivants dans la chaîne des traitements. Ainsi les données internes d'un échantillonneur (impulsion du filtre par exemple) ne sont pas forcément présentes en mémoire cache au début du traitement d'un bloc, générant un « cache miss » et requérant un accès à la mémoire centrale. Le groupement des données accédées simultanément aide à minimiser le ralentissement généré à cette occasion.

Concernant le format des données, l'impulsion peut être stockée en entiers 16 bits si elle est normalisée correctement, de façon à maximiser la dynamique. Nos tests ont montré que cette faible précision n'a qu'un impact mineur sur la bande rejetée, augmentant le bruit d'un demi Décibel dans cette zone, ce qui est ici acceptable. Une optimisation de la valeur des derniers bits opérée par un algorithme génétique tel que Differential Evolution [14] peut aider à régler le problème. En fait, le choix de résolution entre 16 bits et 24 ou 32 bits serait lié à l'implémentation et à la performance du système, spécifique à l'architecture, plutôt qu'à la qualité du signal généré. Cependant les accumulateurs 16 bits entiers ne disposent pas d'une résolution suffisante pour garantir un SNR de 85 dB lors du filtrage.



Effets de la quantification en 16 bits des coefficients sur la réponse fréquentielle de l'interpolateur.



Interpolation. Le rectangle gris représente la mémoire du programme, contenant les impulsions $G_d[k]$ et leurs différences. La partie inférieure du diagramme montre l'interpolation linéaire du FIR, suivi par sa convolution avec l'échantillon. Les données sont groupées par paquets de quatre échantillons, pour montre la parallélisation possible des calculs à l'aide de jeu d'instruction SIMD. Notez qu'il est nécessaire de retourner les impulsions de façon à indexer leurs éléments correctement lors de la convolution (le $N/2-k$ de la formule).

4.1.5 Conception pour r < 1

Comme nous l'avons dit dans notre analyse, le cas $r<1$ a une solution assez efficace : utiliser un niveau supplémentaire de MIP map, sur-échantillonné à un taux de $4/3$. C'est probablement la solution idéale pour une système dans lequel l'utilisation de mémoire et la bande passante disponible ne sont pas des problèmes. Cependant ce n'est pas toujours le cas. Passer rapidement de $r<1$ à $r>1$ et vice-versa, par exemple à l'occasion d'un vibrato, ou

jouer un accord va consommer beaucoup de bande passante mémoire. Nous pouvons freiner ce coût en n'ajoutant pas de niveau MIP map. De plus, l'implémentation est un tantinet complexe. Principalement par simplicité, nous irons dans l'autre direction que nous avons dégagée précédemment : un filtre à pente plus raide.

Pour que l'équilibre de charge de calcul soit conservé entre les deux cas, nous choisirons un filtre de longueur double, mais sans sur-échantillonnage. Hélas, nous allons tout de même un peu tricher avec les caractéristiques originales que nous avons spécifiées pour l'interpolateur. En effet, la raideur atteinte n'est pas suffisante et nous ne pouvons pas garantir le SNR sur une bande passante de 90 % de la bande totale. Nous devons la réduire un peu, et rendre un peu plus progressive l'atténuation en bande coupée. L'algorithme de Parks-McClellan est chargé avec les contraintes suivantes pour $N'=2N=24$ et $M=64$:

Intervalle	Poids	Oscillations ou atténuation (dB)
$[0\ ;0.85F_C]$	1	0.1
$[1.14F_C;1.99F_C]$	64	-81
$[2F_C;3.99F_C]$	128	-87
$[4F_C;F_N]$	256	-93

4.1.6 Le filtre de sous-échantillonnage

Le filtre demi-bande IIR est conçu avec les spécifications suivantes :

- Atténuation en bande coupée : -90 dB
- Largeur de la bande passante : $9F_C/10$

Nous obtenons ces caractéristiques :

- 7 coefficients
- Largeur de la bande de transition (symétrique) : $F_N/10$
- Atténuation effective en bande coupée : -93.3 dB
- Oscillations en bande passante insignifiantes
- Retard de groupe après décimation :
  - 18 échantillons à $9F_{NI}/10$,
  - 2.3 échantillons à $F_{NI}/2$,
  - 1.7 échantillons près de 0 Hz.

Le retard de groupe est assez élevé sur la fin du spectre, mais s'atténue très vite quand la fréquence baisse. Les coefficients supportent très bien la quantification ; il est ainsi tout à fait acceptable de les stocker en 16 bits si nécessaire. De toutes façons il est nécessaire de garder une résolution supérieure pour les variables d'état lors du traitement.

Pour le cas $r<1$, nous n'avons pas besoin de sur-échantillonner, mais nous devons nous assurer que la continuité de la phase sera respectée lors des passages entre les deux cas. Pour arriver à un tel résultat, nous alimenterons le filtre avec le signal non-sur-échantillonné, auquel on aura ajouté un zéro entre chaque échantillon. Le surcoût en calcul est négligeable grâce à la structure du filtre et à son implémentation polyphase.

4.1.7 Transitions entre les niveaux de la MIP-map

Le changement de taux de ré-échantillonnage r nécessite parfois de changer le niveau de MIP-map utilisé. Théoriquement, le changement est presque transparent pour le signal final ; les deux contenus fréquentiels sont les mêmes quand on passe la « frontière ». Cependant, le changement fait apparaître ou disparaître instantanément des harmoniques dans la partie supérieure du spectre du signal sur-échantillonné, ce qui provoque des discontinuités. Celles-ci occupent alors toute la largeur du spectre, provoquant un clic audible dans le signal.

Nous avons deux solutions simples pour contourner le problème. La première consiste à ne plus changer de niveau de MIP map une fois que celui-ci a été sélectionné pour la reproduction d'un son. Nous devons alors supposer que les changements de taux sont faibles et que le repliement ou la perte de hautes fréquences générés par le dépassement de la limite normale du niveau vont être négligeables.

La seconde solution met en jeu un cross-fading des signaux produits par les deux niveaux de MIP-map. Afin de ne pas avoir de problème de phase, le taux r utilisé par les deux niveaux sera strictement identique. Cette méthode utilise donc plus ou moins la première solution (utilisation de niveau hors de sa plage habituelle) lors du fade-out du niveau précédent et du fade-in du niveau suivant.

Le temps inhérent au cross-fade nécessite soit :

- de limiter l'évolution du taux de rééchantillonnage, de façon à être sûr qu'un cross-fade est fini avant de commencer le suivant,
- de retarder le changement de MIP map en attendant la fin du cross-fade courant,
- d'adapter la durée des cross-fade à la transition en cours.

Le principal inconvénient de la seconde méthode est qu'elle nécessite un doublement des calculs pendant un cours laps de temps. Notez que la transition entre les taux $r<1$ et $r>1$ relève exactement de la même problématique.

4.1.8 Performances

Il est facile d'évaluer les performances par rapport au débit de sortie si l'on se focalise uniquement sur les calculs DSP effectués en temps réel. L'interpolateur FIR est constitué de 12 coefficients, nécessitant ainsi 12 MAC. Pour générer cet interpolateur, nous devons interpoler deux impulsions en implémentant la formule $y=x_1+k(x_2-x_1)$. Comme x_2-x_1 est pré-calculé, le calcul ne demande qu'un MAC par coefficient. Nous voici donc avec 12 MAC de plus, pour un total de 24 MAC. Comme le filtre d'interpolation est sur-échantillonné d'un facteur 2, nous devons multiplier ce chiffre par 2. Enfin, le filtre de décimation demande 1 MAC par coefficient et par échantillon de sortie, ce qui fait donc 7 MAC pour cet étage.

Le total donne 55 MAC par échantillon de sortie.

Évidemment, cela ne peut mesurer les performances réelles de l'algorithme. Nous devons prendre en compte la partie « logique » du programme : sélection du niveau de MIP map, l'évolution de la phase, le chargement, la préparation et le stockage des données, etc. Quoi qu'il en soit, ces parties devraient être relativement légère, comparées aux calculs. À cause de la structure vectorisée des données, les dépendances entre les résultats intermédiaires sont relativement faibles. Il est donc possible d'opérer la logique et les calculs plus ou moins en parallèle. De plus, c'est une opportunité parfaite pour utiliser les jeux d'instruction SIMD.

Si plusieurs voies jouent en même temps, il peut être plus efficace de mixer ces voies avant la décimation du sur-échantillonnage, bien que ce soit contre-intuitif.

En plus du chargement des MIP maps, l'occupation mémoire peut être grossièrement exprimée par la formule suivante :

$$S=2(N+N')\cdot M\cdot D \quad (4.1.8\text{-}1)$$

Où D est la taille prise par un coefficient, en octets ; par exemple 4 pour des données en virgule flottante 32 bits. Dans notre cas, les coefficients de l'interpolateur occupent 18 kB en mémoire.

Nous avons implémenté la méthode décrite et obtenu une vitesse moyenne de 178 cycles d'horloge par échantillon de sortie sur un ordinateur conventionnel. Cette valeur est largement suffisante pour la reproduction polyphonique d'échantillons en temps réel. Le test a été accompli dans des conditions de test réalistes, prenant en compte les problèmes de mémoire cache. Le programme a été écrit en C++ standard, avec un traitement complet en virgule flottante (32 bits pour le stockage), sans code assembleur explicite ou d'instructions SIMD. On peut ainsi encore largement optimiser le programme en utilisant ces techniques de programmation. Le code source de ce programme, ainsi que celui de la libraire « resampler » qui peut être utilisée pour des applications concrètes, est disponible.

4.2 Variations possibles

Dans la section précédente, nous avons décrit une implémentation particulière de notre méthode de ré-échantillonnage. Ce n'est bien sûr pas la seule possible. Nous pouvons ajuster plusieurs paramètres pour respecter des spécifications différentes. Nous allons lister dans cette section – de manière non-exhaustive – quelques variations possibles :

- **Nombre de phases (M).** L'augmenter rejette le repliement causé par les lobes latéraux dans la réponse spectrale de l'interpolateur, mais demande plus de mémoire pour le filtre.
- **Longueur de la phase (N).** La qualité générale du filtre varie dans la même direction que la longueur de phase. Cette variable permet d'établir le compromis entre les oscillations en bande passante, le niveau en bande coupée et la largeur de la bande de transition.
- **Interpolation de la phase.** Nous avons choisi une interpolation linéaire, mais il est possible d'utiliser des polynômes d'ordre supérieur afin de troquer dans une certaine limite de l'occupation de mémoire contre des calculs.
- **Ordre d'exécution des interpolations.** Selon le contexte, il peut être plus efficace d'interpoler d'abord deux échantillons de sortie contigus à l'aide du FIR, puis d'utiliser l'interpolateur polynomial pour extraire l'échantillon final.
- **Méthode de conception du filtre.** Elle dépend des outils dont vous disposez ! Si vous ne pouvez pas utiliser l'algorithme de Parks-McClellan, essayez un sinc fenêtré. Les fenêtres classiques comme celles de Hanning ou de Blackman-Harris ont généralement des lobes latéraux trop élevés. Préférez les fenêtres que vous pouvez paramétrer, comme celle de Dolph-Chebyshev ou de Kaiser.
- **Taux de sur-échantillonnage.** Il est possible de baisser le taux de sur-échantillonnage jusqu'à $3/2$ pour des MIP map espacées d'une octave. Cela peut réduire les calculs mais nécessite un filtre d'interpolation à pente plus raide. On peut aussi augmenter ce taux pour baisser les contraintes sur le filtre.
- **Espacement des niveaux de MIP map.** L'octave est facile à implémenter, mais il est possible d'utiliser d'autres espaces, comme l'octave double ou la demi-octave, changeant alors l'occupation mémoire. La largeur de la bande de transition du filtre peut être réduite, ainsi que le taux de ré-échantillonnage. Il est même possible de se débarrasser du MIP mapping si r reste bas.
- **Adaptation dynamique du FIR.** La table polyphase du FIR peut être changée en fonction du taux de ré-échantillonnage. Nous l'avons fait pour les cas $r<1$ et $r>1$; il est possible d'ajuster le FIR pour n'importe quelle plage de variation. Cela permettrait de se débarrasser des MIP maps, et même peut-être de l'étage de sur-échantillonnage, parfois au prix d'un peu plus de calculs. Cela peut être une alternative importante à considérer.

- **Stockage du filtre.** Comme l'impulsion est symétrique (phase linéaire), il est possible de stocker seulement la moitié des coefficients. Cependant ces données deviennent plus compliquées à récupérer au moment des calculs.
- **Filtre à phase minimale.** Afin de réduire au maximum le retard entre l'ordre de jeu d'un échantillon et sa sortie effective, on peut utiliser un filtre d'interpolation à phase minimale. Une transformation peut être appliquée au filtre linéaire pour obtenir un filtre à phase minimale.
- **Pré-interpolation de la phase.** Nous pouvons ne pas stocker les différences entre deux phases, ce qui libère de la mémoire mais augmente les calculs.
- **Filtre de décimation.** Il peut être modifié si nécessaire. Il peut être judicieux de l'implémenter sous forme de FIR, surtout si le taux de sur-échantillonnage n'est pas une puissance de deux.
- **Cas** $r<1$. Des méthodes alternatives ont été évoquées dans une section précédente.
- **Changement de niveaux de MIP map.** Même chose que précédemment.
- **Fréquence d'échantillonnage de sortie.** C'est un point assez important car les contraintes sur le filtres peuvent être influencées par la fréquence de sortie. En 44.1 kHz nous avons 2 kHz de marge, alors qu'en 48 kHz nous avons le double, sans parler des 96 kHz et au-delà.

5. Références

[1] Polynomial Interpolators for High-Quality Resampling of Oversampled Audio
Olli Niemitalo, août 2001
http://www.biochem.oulu.fi/~oniemita/dsp/deip.pdf

[2] Performance of Low-Order Polynomial Interpolators in the Presence of Oversampled Input
Duane K. Wise & Robert Bristow-Johnson
107ème convention de l'AES, septembre 1997

[3] The Effects Of Quantizing The Fractional Interval In Interpolation Filters
Jussi Vesma, Francisco Lopez, Tapio Saramäki & Markku Renfors [lien cassé]
http://www.es.isy.liu.se/norsig2000/publ/page215_id108.pdf

[4] Multirate Digital Signal Processing
Ronald E. Crochiere et Lawrence R. Rabiner
Éditions Prentice-Hall

[5] Digital Audio Resampling Home Page
Julius O. Smith III, janvier 2000
http://ccrma-www.stanford.edu/~jos/resample/resample.html

[6] Windowing Functions Improve FFT Results
Richard Lyons
Test & Measurement World, juin et septembre 1998, pp. 37-44
http://www.e-insite.net/index.asp?layout=article&articleId=CA187572
http://www.e-insite.net/index.asp?layout=article&articleId=CA187573

[7] The Dolph-Chebyshev Window — A Simple Optimal Filter
Peter Lynch, janvier 1996
http://www.maths.tcd.ie/~plynch/Publications/Dolph.pdf

[8] MIP-mapping
http://whatis.techtarget.com

[9] Polyphase Filter Designer in Java
Artur Krukowski
http://www.cmsa.wmin.ac.uk/~artur/Poly.html

[10] FIR Digital Filter Design Techniques Using Weighted Chebyshev Approximations
L. R. Rabiner, J. H. McClellan, and T. W. Parks
Proc. IEEE 63, 1975

[11] Computational Improvements To Linear Convolution With Multirate Filtering Methods
Jason R. VandeKieft, avril 1998
http://www.music.miami.edu/programs/mue/Research/jvandekieft

[12] The Mathematical Theory of Dithered Quantization
Robert A. Wannamaker, thèse de Ph.D., juillet 1997
Département de Mathématiques Appliquées, Université de Waterloo, Canada
http://audiolab.uwaterloo.ca/~rob/phd.html

[13] Whither Dither - Experience with High-Order Dithering Algorithms In The Studio
James A. Moorer et Julia C. Wen
95ème convention de l'AES, octobre 1993
http://www.jamminpower.com/main/articles.jsp

[14] Differential Evolution homepage
Kenneth Price et Rainer Storn
http://www.icsi.berkeley.edu/~storn/code.html

6. Appendices

6.1 Coefficients des filtres

```
double iir_polyphase_lpf = {
0.0457281, 0.168088, 0.332501, 0.504486,
0.663202, 0.803781, 0.933856 };

double fir_halfband_lpf [81] = {
0.000199161, 0.000652681, 0.000732184, -5.20771e-005,
-0.0007641618462, -5.638448426e-005, 0.001043007134, 0.0002618629784,
-0.001427331739, -0.0005903598292, 0.001886374043, 0.001076466716,
-0.00241732264, -0.001750193532, 0.003009892088, 0.002655276516,
-0.003657747155, -0.003838234309, 0.004347045423, 0.005358361364,
-0.005067457982, -0.007294029088, 0.00579846115, 0.009746961422,
-0.006524811739, -0.01286996135, 0.007226610031, 0.01690351872,
-0.007884756779, -0.02225965965, 0.008479591244, 0.02971806374,
-0.008993687303, -0.04095867041, 0.009411374397, 0.06037160312,
-0.009719550593, -0.1041018935, 0.009908930771, 0.3176382757,
0.4900273874, 0.3176382757, 0.009908930771, -0.1041018935,
-0.009719550593, 0.06037160312, 0.009411374397, -0.04095867041,
-0.008993687303, 0.02971806374, 0.008479591244, -0.02225965965,
-0.007884756779, 0.01690351872, 0.007226610031, -0.01286996135,
-0.006524811739, 0.009746961422, 0.00579846115, -0.007294029088,
-0.005067457982, 0.005358361364, 0.004347045423, -0.003838234309,
-0.003657747155, 0.002655276516, 0.003009892088, -0.001750193532,
-0.00241732264, 0.001076466716, 0.001886374043, -0.0005903598292,
-0.001427331739, 0.0002618629784, 0.001043007134, -5.638448426e-005,
-0.0007641618462, -5.20770713e-005, 0.0007321838627, 0.000652681315,
0.0001991612514
};

double fir_interpolator_1 [64 * 12] = {
0, 0.001742279734, 0.0003227324025, 0.0003504220639,
0.0003784201858, 0.0004060523799, 0.000433560667, 0.000460242331,
0.0004863371066, 0.0005111187936, 0.0005348296736, 0.0005567178399,
0.0005577003141, 0.0005594868077, 0.0006105017978, 0.0006229741377,
0.0006324574426, 0.0006637955229, 0.0006397031985, 0.0006367227753,
0.0006294231227, 0.0006168285476, 0.0005994721071, 0.000576164941,
0.0005473310058, 0.0005113181524, 0.0004686499247, 0.0004176613044,
0.000360179431, 0.000294828955, 0.0002239625091, 0.0001397615714,
4.917075029e-005, -4.906454506e-005, -0.0001586730304, -0.0002759525972,
-0.0004037611106, -0.0005403286683, -0.0006872050613, -0.0008432719479,
-0.001009559165, -0.001185183727, -0.00137082501, -0.00156565289,
-0.001770105534, -0.001983373564, -0.00220574127, -0.002436438721,
-0.002675609502, -0.002922412142, -0.003176764646, -0.003437642969,
-0.003704785626, -0.003977186249, -0.004254645554, -0.004536168589,
-0.004821306638, -0.0051085353, -0.005397156872, -0.005686020159,
-0.005975209561, -0.006262452042, -0.006546344761, -0.006827480141,
-0.007102792819, -0.007372232611, -0.007633943637, -0.00788688517,
-0.008129587034, -0.008360797821, -0.008579124816, -0.008783232512,
-0.008971721235, -0.00914319924, -0.009296252118, -0.009429483812,
-0.009541564083, -0.009631162555, -0.009697032537, -0.00973786063,
-0.009752387548, -0.009739324335, -0.009697580358, -0.009626088825,
-0.009523960257, -0.009390153133, -0.009223736719, -0.009023832733,
-0.008790026698, -0.008521722857, -0.008218254761, -0.007878933678,
-0.007504467131, -0.007093708517, -0.006647210362, -0.006165014615,
-0.005647212846, -0.005094386491, -0.004506902567, -0.003885599968,
-0.003231214068, -0.002544844667, -0.00182755147, -0.001080682372,
-0.0003055993791, 0.0004960207406, 0.001322467315, 0.002171720624,
0.003041773856, 0.003930379128, 0.004835267947, 0.005753857674,
0.006683530599, 0.007621410207, 0.008564704475, 0.009510380936,
0.01045533995, 0.01139613927, 0.0123295518, 0.01325220433,
0.01416068595, 0.01505101153, 0.01591999786, 0.01676380471,
0.01757851783, 0.01836076832, 0.01910650003, 0.01981223275,
0.02047409904, 0.02108852463, 0.02165179807, 0.02216040787,
0.02261085054, 0.02299979734, 0.02332394892, 0.0235801863,
0.0237654538, 0.02387693283, 0.02391195908, 0.02386807742,
0.0237429762, 0.02353457915, 0.02324103906, 0.0228608214,
0.02239262438, 0.02183534433, 0.02118814096, 0.02045058738,
0.01962257961, 0.01870419866, 0.01769581587, 0.01659837872,
0.01541303206, 0.01414098157, 0.01278441212, 0.0113452081,
0.009826081402, 0.008229855716, 0.006559784686, 0.004819464346,
0.003012819169, 0.001144170699, -0.0007818426856, -0.002760242962,
-0.004785736303, -0.006852724797, -0.008955278991, -0.01108715657,
-0.01324187199, -0.01541268359, -0.01759261257, -0.01977441632,
-0.02195062519, -0.02411358996, -0.02625557221, -0.0283865651,
-0.03044473373, -0.03247564125, -0.03445323438, -0.036369256,
-0.03821536833, -0.03998331928, -0.04166508141, -0.04325230996,
-0.04473714713, -0.04611169335, -0.04736816632, -0.04849918852,
-0.04949738927, -0.05035578772, -0.05106786765, -0.051627048,
-0.05202737026, -0.05226326572, -0.05232951832, -0.0522214062,
-0.05193460354, -0.05146539981, -0.05081054857, -0.04996742686,
-0.04893390401, -0.04770850047, -0.04629031953, -0.04467920731,
-0.04287556956, -0.04088051322, -0.03869581877, -0.03632406645,
-0.03376841422, -0.03103276272, -0.02812175712, -0.02504086042,
-0.02179601806, -0.01839411765, -0.01484270347, -0.01114988159,
-0.007324699964, -0.003376659821, 0.000683918981, 0.004846165968,
0.009098558205, 0.01342894118, 0.01782462363, 0.02227232605,
0.0267583219, 0.03126837881, 0.0357878146, 0.04030152292,
0.04479407263, 0.04924969032, 0.05365237905, 0.05798585405,
0.06223368171, 0.06637930212, 0.07040613857, 0.0742975011,
0.07803678014, 0.08160747482, 0.08499328207, 0.08817799363,
0.09114574369, 0.09388102591, 0.09636868824, 0.09859394674,
0.1005427553, 0.1022013392, 0.1035567907, 0.1045967585,
0.1053096883, 0.1056848338, 0.1057122442, 0.1053829182,
0.1046887854, 0.1036228114, 0.1021790258, 0.1003525021,
0.09813949524, 0.09553742354, 0.09254493804, 0.08916193185,
0.08538957045, 0.08123030902, 0.07668798651, 0.07176776518,
0.06647614661, 0.06082100909, 0.05481167793, 0.04845881893,
0.04177446192, 0.03477204034, 0.02746641332, 0.01987371878,
0.0120114516, 0.003898510203, -0.004445050075, -0.01299785294,
-0.02173724057, -0.03063949823, -0.03967960565, -0.04883163923,
-0.05806849217, -0.06736218009, -0.07668371714, -0.08600331368,
-0.09529038984, -0.104513663, -0.1136411693, -0.1226404465,
-0.1314784946, -0.1401219554, -0.1485372711, -0.1566904448,
-0.1645474937, -0.172074439, -0.1792372909, -0.1860022387,
-0.1923357307, -0.1982046325, -0.2035761834, -0.2084182126,
-0.2126992079, -0.2163884272, -0.2194558911, -0.221872701,
-0.223610932, -0.2246437585, -0.2249456736, -0.2244923915,
-0.2232611184, -0.2212305232, -0.2183808121, -0.2146293907,
-0.2101533743, -0.2047446886, -0.198455083, -0.1912737755,
-0.1831919406, -0.1742028128, -0.1643016924, -0.1534860427,
-0.1417554206, -0.1291116623, -0.1155587919, -0.1011030852,
-0.085753021, -0.06951943703, -0.05241538944, -0.03445621409,
-0.01565946649, 0.003954984002, 0.02436513506, 0.04554676387,
0.06747348062, 0.09011675956, 0.1134461258, 0.1374289443,
0.162030763, 0.1872151458, 0.212943884, 0.2391770576,
0.2658730424, 0.2929886858, 0.3204793157, 0.4619038651,
0.4906368532, 0.5193995898, 0.5481394221, 0.5768031931,
0.6053374807, 0.6336885973, 0.6618027691, 0.689626244,
0.7171055053, 0.7441872751, 0.7708187431, 0.7969476598,
0.8225225065, 0.847492502, 0.8718078895, 0.8954199882,
0.9182812855, 0.9403455873, 0.9615681762, 0.981905798,
1.001316998, 1.019761942, 1.037202776, 1.053603525,
1.068930331, 1.08315147, 1.096237409, 1.108160957,
1.118897291, 1.128423998, 1.136721226, 1.143771574,
1.149560292, 1.154075265, 1.157307047, 1.159248818,
1.159896529, 1.159248818, 1.157307047, 1.154075265,
1.149560292, 1.143771574, 1.136721226, 1.128423998,
1.118897291, 1.108160957, 1.096237409, 1.08315147,
1.068930331, 1.053603525, 1.037202776, 1.019761942,
1.001316998, 0.981905798, 0.9615681762, 0.9403455873,
0.9182812855, 0.8954199882, 0.8718078895, 0.847492502,
0.8225225065, 0.7969476598, 0.7708187431, 0.7441872751,
0.7171055053, 0.689626244, 0.6618027691, 0.6336885973,
0.6053374807, 0.5768031931, 0.5481394221, 0.5193995898,
0.4906368532, 0.4619038651, 0.4332526757, 0.4047346019,
0.3764001754, 0.3482989051, 0.3204793157, 0.2929886858,
0.2658730424, 0.2391770576, 0.212943884, 0.1872151458,
0.162030763, 0.1374289443, 0.1134461258, 0.09011675956,
0.06747348062, 0.04554676387, 0.02436513506, 0.003954984002,
-0.01565946649, -0.03445621409, -0.05241538944, -0.06951943703,
-0.085753021, -0.1011030852, -0.1155587919, -0.1291116623,
-0.1417554206, -0.1534860427, -0.1643016924, -0.1742028128,
-0.1831919406, -0.1912737755, -0.198455083, -0.2047446886,
-0.2101533743, -0.214693907, -0.2183808121, -0.2212305232,
-0.2232611184, -0.2244923915, -0.2249456736, -0.2246437585,
-0.223610932, -0.221872701, -0.2194558911, -0.2163884272,
-0.2126992079, -0.2084182126, -0.2035761834, -0.1982046325,
-0.1923357307, -0.1860022387, -0.1792372909, -0.172074439,
-0.1645474937, -0.1566904448, -0.1485372711, -0.1401219554,
-0.1314784946, -0.1226404465, -0.1136411693, -0.104513663,
-0.09529038984, -0.08600331368, -0.07668371714, -0.06736218009,
-0.05806849217, -0.04883163923, -0.03967960565, -0.03063949823,
-0.02173724057, -0.01299785294, -0.004445050075, 0.003898510203,
0.0120114516, 0.01987371878, 0.02746641332, 0.03477204034,
0.04177446192, 0.04845881893, 0.05481167793, 0.06082100909,
0.06647614661, 0.07176776518, 0.07668798651, 0.08123030902,
0.08538957045, 0.08916193185, 0.09254493804, 0.09553742354,
0.09813949524, 0.1003525021, 0.1021790258, 0.1036228114,
0.1046887854, 0.1053829182, 0.1057122442, 0.1056848338,
0.1053096883, 0.1045967585, 0.1035567907, 0.1022013392,
0.1005427553, 0.09859394674, 0.09636868824, 0.09388102591,
0.09114574369, 0.08817799363, 0.08499328207, 0.08160747482,
0.07803678014, 0.0742975011, 0.07040613857, 0.06637930212,
0.06223368171, 0.05798585405, 0.05365237905, 0.04924969032,
0.04479407263, 0.04030152292, 0.0357878146, 0.03126837881,
0.0267583219, 0.02227232605, 0.01782462363, 0.01342894118,
0.009098558205, 0.004846165968, 0.000683918981, -0.003376659821,
-0.007324699964, -0.01114988159, -0.01484270347, -0.01839411765,
-0.02179601806, -0.02504086042, -0.02812175712, -0.03103276272,
-0.03376841422, -0.03632406645, -0.03869581877, -0.04088051322,
-0.04287556956, -0.04467920731, -0.04629031953, -0.04770850047,
-0.04893390401, -0.04996742686, -0.05081054857, -0.05146539981,
-0.05193460354, -0.0522214062, -0.05232951832, -0.05226326572,
-0.05202737026, -0.051627048, -0.05106786765, -0.05035578772,
-0.04949738927, -0.04849918852, -0.04736816632, -0.04611169335,
-0.04473714713, -0.04325230996, -0.04166508141, -0.03998331928,
-0.03821536833, -0.036369256, -0.03445323438, -0.03247564125,
-0.03044473373, -0.02836865061, -0.02625557221, -0.02411358996,
-0.02195062519, -0.01977441632, -0.01759261257, -0.01541268359,
-0.01324187199, -0.01108715657, -0.008955278991, -0.006852724797,
-0.004785736303, -0.002760242962, -0.0007818426856, 0.001144170699,
0.003012819169, 0.004819464346, 0.006559784686, 0.008229855716,
0.009826081402, 0.0113452081, 0.01278441212, 0.01414098157,
0.01541303206, 0.01659837872, 0.01769581587, 0.01870419866,
0.01962257961, 0.02045058738, 0.02118814096, 0.02183534433,
0.02239262438, 0.0228608214, 0.02324103906, 0.02353457915,
0.0237429762, 0.02386807742, 0.02391195908, 0.02387693283,
0.0237654538, 0.0235801863, 0.02332394892, 0.02299979734,
0.02261085054, 0.02216040787, 0.02165179807, 0.02108852463,
0.02047409904, 0.01981223275, 0.01910650003, 0.01836076832,
0.01757851783, 0.01676380471, 0.01591999786, 0.01505101153,
0.01416068595, 0.01325220433, 0.0123295518, 0.01139613927,
0.01045533995, 0.009510380936, 0.008564704475, 0.007621410207,
0.006683530599, 0.005753857674, 0.004835267947, 0.003930379128,
0.003041773856, 0.002171720624, 0.001322467315, 0.0004960207406,
-0.0003055993791, -0.001080682372, -0.00182755147, -0.002544844667,
-0.003231214068, -0.003885599968, -0.004506902567, -0.005094386491,
-0.005647212846, -0.006165014615, -0.006647210362, -0.007093708517,
-0.007504467131, -0.007878933678, -0.008218254761, -0.008521722857,
-0.008790026698, -0.009023832733, -0.009223736719, -0.009390153133,
-0.009523960257, -0.009626088825, -0.009697580358, -0.009739324335,
-0.009752387548, -0.00973786063, -0.009697032537, -0.00963116255,
-0.009541564083, -0.009429483812, -0.009296252118, -0.00914319924,
-0.008971721235, -0.008783232512, -0.008579124816, -0.008360797821,
-0.008129587034, -0.00788688517, -0.007633943637, -0.007372232611,
-0.007102792819, -0.006827480141, -0.006546344761, -0.006262452042,
-0.005975209561, -0.005686020159, -0.005397156872, -0.0051085353,
-0.004821306638, -0.004536168589, -0.004254645554, -0.003977186249,
-0.003704785626, -0.003437642969, -0.003176764646, -0.002922412142,
-0.002675609502, -0.002436438721, -0.00220574127, -0.001983373564,
-0.001770105534, -0.00156565289, -0.00137082501, -0.001185183727,
-0.001009559165, -0.0008432719479, -0.0006872050613, -0.0005403286683,
-0.0004037611106, -0.0002759525972, -0.0001586730304, -4.906454506e-005,
4.917075029e-005, 0.0001397615714, 0.0002239625091, 0.000294828955,
0.000360179431, 0.0004176613044, 0.0004686499247, 0.0005113181524,
0.0005473310058, 0.000576164941, 0.0005994721071, 0.0006168285476,
0.0006294231227, 0.0006367227753, 0.0006397031985, 0.000637955229,
0.0006324574426, 0.0006229741377, 0.0006105017978, 0.000594868077,
0.000577003141, 0.0005567178399, 0.0005348296736, 0.0005111187936,
0.0004863371066, 0.000460242331, 0.000433560667, 0.0004060523799,
0.0003784201858, 0.0003504220639, 0.0003227324025, 0.001742279734 };
```

```
double fir_interpolator_2 [64 * 24] = {
0, 0.00084549, 0.00025079, 0.00028614,
0.00032345, 0.0003625, 0.00040326, 0.00044552,
0.00048927, 0.00053431, 0.00058068, 0.00062815,
0.00067673, 0.0007262, 0.00077656, 0.00082761,
0.00087945, 0.00093197, 0.00098539, 0.0010396,
0.0010947, 0.0011503, 0.0012067, 0.0012634,
0.0013208, 0.0013791, 0.0014384, 0.001498,
0.0015577, 0.0016174, 0.0016795, 0.0017398,
0.0018011, 0.0018625, 0.0019233, 0.0019841,
0.002044, 0.0021034, 0.0021616, 0.0022187,
0.0022743, 0.0023281, 0.0023799, 0.0024295,
0.0024766, 0.0025212, 0.0025628, 0.0026013,
0.0026363, 0.0026679, 0.0026957, 0.0027199,
0.0027399, 0.0027558, 0.0027673, 0.0027746,
0.0027774, 0.0027755, 0.0027687, 0.0027577,
0.0027414, 0.0027203, 0.0026943, 0.0026631,
0.0026269, 0.0025854, 0.0025387, 0.0024866,
0.0024291, 0.002366, 0.0022975, 0.0022233,
0.0021434, 0.0020578, 0.0019663, 0.0018691,
0.001766, 0.0016571, 0.0015423, 0.0014217,
0.0012953, 0.0011633, 0.0010257, 0.00088249,
0.00073402, 0.00058038, 0.00042169, 0.00025803,
8.9978e-005, -8.2619e-005, -0.00025933, -0.00043984,
-0.00062408, -0.00081155, -0.0010021, -0.0011953,
-0.001391, -0.0015887, -0.0017882, -0.0019891,
-0.0021911, -0.0023938, -0.0025969, -0.0028001,
-0.0030029, -0.0032051, -0.0034062, -0.0036058,
-0.0038038, -0.0039995, -0.0041926, -0.0043828,
-0.0045696, -0.0047527, -0.0049315, -0.0051058,
-0.0052751, -0.0054388, -0.0055967, -0.0057482,
-0.005893, -0.0060306, -0.0061605, -0.0062825,
-0.006396, -0.0065006, -0.006596, -0.0066818,
-0.0067576, -0.0068232, -0.0068781, -0.0069221,
-0.0069549, -0.0069763, -0.006986, -0.0069839,
-0.0069698, -0.0069435, -0.0069049, -0.0068539,
-0.0067904, -0.0067144, -0.0066258, -0.0065247,
-0.0064111, -0.006285, -0.0061465, -0.0059957,
-0.0058327, -0.0056577, -0.0054708, -0.0052722,
-0.0050622, -0.004841, -0.0046088, -0.004366,
-0.0041129, -0.0038499, -0.0035773, -0.0032955,
-0.0030051, -0.0027064, -0.0023999, -0.0020863,
-0.0017661, -0.0014398, -0.0011081, -0.00077158,
-0.00043093, -8.6808e-005, 0.00026005, 0.00060893,
0.00095913, 0.0013099, 0.0016604, 0.0020099,
0.0023577, 0.0027029, 0.0030449, 0.0033826,
0.0037155, 0.0040427, 0.0043634, 0.0046769,
0.0049823, 0.005279, 0.0055661, 0.005843,
0.0061088, 0.006363, 0.0066047, 0.0068334,
0.0070483, 0.0072487, 0.0074342, 0.007604,
0.0077575, 0.0078942, 0.0080136, 0.0081152,
0.0081984, 0.0082628, 0.0083081, 0.0083337,
0.0083395, 0.008325, 0.0082901, 0.0082345,
0.008158, 0.0080605, 0.0079419, 0.0078022,
0.0076415, 0.0074597, 0.007257, 0.0070336,
0.0067897, 0.0065255, 0.0062415, 0.0059379,
0.0056152, 0.0052738, 0.0049144, 0.0045374,
0.0041434, 0.0037332, 0.0033074, 0.0028667,
0.002412, 0.0019441, 0.0014638, 0.00097207,
0.0004698, -4.2025e-005, -0.00056238, -0.0010903,
-0.0016245, -0.0021642, -0.002708, -0.0032548,
-0.0038035, -0.0043529, -0.0049017, -0.0054486,
-0.0059925, -0.006532, -0.0070659, -0.0075928,
-0.0081116, -0.0086207, -0.0091191, -0.0096053,
-0.010078, -0.010536, -0.010979, -0.011404,
-0.01181, -0.012198, -0.012564, -0.012909,
-0.01323, -0.013528, -0.013801, -0.014048,
-0.014268, -0.01446, -0.014624, -0.014759,
-0.014864, -0.014938, -0.014981, -0.014993,
-0.014972, -0.014919, -0.014833, -0.014714,
-0.014561, -0.014376, -0.014157, -0.013904,
-0.013619, -0.0133, -0.012948, -0.012564,
-0.012148, -0.011701, -0.011222, -0.010714,
-0.010175, -0.0096082, -0.0090132, -0.0083913,
-0.0077436, -0.0070711, -0.006375, -0.0056567,
-0.0049174, -0.0041585, -0.0033816, -0.002588,
-0.0017795, -0.00095756, -0.00012395, 0.00071962,
0.0015714, 0.0024295, 0.0032921, 0.0041574,
0.0050234, 0.0058881, 0.0067497, 0.0076062,
0.0084555, 0.0092957, 0.010125, 0.010941,
0.011742, 0.012526, 0.01329, 0.014034,
0.014755, 0.015451, 0.01612, 0.016761,
0.017372, 0.01795, 0.018495, 0.019004,
0.019476, 0.01991, 0.020304, 0.020656,
0.020966, 0.021231, 0.021452, 0.021626,
0.021754, 0.021833, 0.021864, 0.021845,
0.021776, 0.021657, 0.021486, 0.021265,
0.020993, 0.02067, 0.020295, 0.01987,
0.019395, 0.018869, 0.018294, 0.01767,
0.016999, 0.016281, 0.015516, 0.014707,
0.013855, 0.012961, 0.012026, 0.011052,
0.010042, 0.0089957, 0.0079163, 0.0068055,
0.0056656, 0.0044987, 0.0033071, 0.0020932,
0.00085947, -0.00039149, -0.0016571, -0.0029347,
-0.0042215, -0.0055146, -0.0068114, -0.0081088,
-0.0094039, -0.010694, -0.011975, -0.013246,
-0.014502, -0.015741, -0.016959, -0.018154,
-0.019323, -0.020463, -0.02157, -0.022642,
-0.023677, -0.02467, -0.02562, -0.026524,
-0.027379, -0.028183, -0.028933, -0.029627,
-0.030263, -0.030839, -0.031352, -0.0318,
-0.032183, -0.032498, -0.032744, -0.032919,
-0.033023, -0.033053, -0.03301, -0.032892,
-0.032699, -0.03243, -0.032086, -0.031665,
-0.031168, -0.030595, -0.029947, -0.029224,
-0.028427, -0.027556, -0.026614, -0.025601,
-0.024518, -0.023368, -0.022152, -0.020872,
-0.019531, -0.01813, -0.016672, -0.015159,
-0.013596, -0.011984, -0.010326, -0.008626,
-0.0068872, -0.0051131, -0.0033072, -0.0014732,
0.00038505, 0.0022636, 0.0041584, 0.0060655,
0.0079806, 0.0098996, 0.011818, 0.013732,
0.015636, 0.017527, 0.0194, 0.02125,
0.023074, 0.024866, 0.026622, 0.028338,
0.03001, 0.031633, 0.033203, 0.034716,
0.036167, 0.037553, 0.03887, 0.040114,
0.041281, 0.042369, 0.043372, 0.044289,
0.045116, 0.04585, 0.046488, 0.047028,
0.047468, 0.047805, 0.048037, 0.048163,
0.048182, 0.04809, 0.047889, 0.047577,
0.047153, 0.046616, 0.045968, 0.045208,
0.044336, 0.043353, 0.042261, 0.041059,
0.039751, 0.038336, 0.036818, 0.035199,
0.03348, 0.031665, 0.029757, 0.027758,
0.025673, 0.023505, 0.021258, 0.018935,
0.016543, 0.014084, 0.011563, 0.0089869,
0.0063592, 0.0036857, 0.00097182, -0.0017767,
-0.0045541, -0.0073545, -0.010172, -0.013,
-0.015832, -0.018663, -0.021485, -0.024292,
-0.027078, -0.029837, -0.03256, -0.035243,
-0.037879, -0.04046, -0.04298, -0.045434,
-0.047814, -0.050115, -0.05233, -0.054453,
-0.056479, -0.058401, -0.060214, -0.061913,
-0.063491, -0.064946, -0.06627, -0.067461,
-0.068513, -0.069423, -0.070186, -0.070799,
-0.071259, -0.071563, -0.071709, -0.071693,
-0.071515, -0.071171, -0.070662, -0.069986,
-0.069142, -0.068131, -0.066952, -0.065607,
-0.064095, -0.062418, -0.060579, -0.058578,
-0.056419, -0.054104, -0.051636, -0.04902,
-0.046258, -0.043356, -0.040318, -0.037149,
-0.033855, -0.030441, -0.026913, -0.023278,
-0.019543, -0.015715, -0.011801, -0.0078082,
-0.0037453, 0.00037992, 0.0045589, 0.008783,
0.013043, 0.017331, 0.021637, 0.025951,
0.030264, 0.034567, 0.038849, 0.043101,
0.047313, 0.051475, 0.055577, 0.059608,
0.063559, 0.067421, 0.071182, 0.074834,
0.078366, 0.081769, 0.085034, 0.088151,
0.091111, 0.093906, 0.096527, 0.098965,
0.10121, 0.10326, 0.10511, 0.10674,
0.10815, 0.10933, 0.11029, 0.111,
0.11148, 0.1117, 0.11168, 0.1114,
0.11086, 0.11006, 0.109, 0.10767,
0.10608, 0.10422, 0.1021, 0.099704,
0.097048, 0.094129, 0.090949, 0.087511,
0.083819, 0.079877, 0.075689, 0.071261,
0.066598, 0.061708, 0.056597, 0.051272,
0.045742, 0.040016, 0.034103, 0.028013,
0.021756, 0.015343, 0.0087852, 0.0020949,
-0.0047158, -0.011634, -0.018646, -0.02574,
-0.032899, -0.040111, -0.047361, -0.054632,
-0.061911, -0.069181, -0.076426, -0.08363,
-0.090777, -0.097851, -0.10483, -0.11171,
-0.11846, -0.12507, -0.13153, -0.1378,
-0.14389, -0.14977, -0.15542, -0.16083,
-0.16599, -0.17086, -0.17545, -0.17973,
-0.18369, -0.18731, -0.19058, -0.19349,
-0.19601, -0.19814, -0.19987, -0.20117,
-0.20204, -0.20247, -0.20245, -0.20196,
-0.201, -0.19955, -0.19761, -0.19518,
-0.19223, -0.18878, -0.18481, -0.18031,
-0.17529, -0.16973, -0.16365, -0.15703,
-0.14988, -0.14219, -0.13397, -0.12522,
-0.11594, -0.10613, -0.09581, -0.08497,
-0.07362, -0.061769, -0.049424, -0.036594,
-0.023289, -0.0095179, 0.0047068, 0.019373,
0.034469, 0.049979, 0.06589, 0.082187,
0.098854, 0.11587, 0.13323, 0.1509,
0.16887, 0.18713, 0.20564, 0.22439,
0.24336, 0.26253, 0.28187, 0.30136,
0.32099, 0.34072, 0.36054, 0.38042,
0.40033, 0.42025, 0.44016, 0.46004,
0.47985, 0.49958, 0.51919, 0.53867,
0.55798, 0.57711, 0.59603, 0.61472,
0.63314, 0.65129, 0.66912, 0.68663,
0.70378, 0.72056, 0.73694, 0.7529,
0.76842, 0.78347, 0.79805, 0.81212,
0.82567, 0.83869, 0.85114, 0.86303,
0.87432, 0.88501, 0.89508, 0.90452,
0.91331, 0.92144, 0.9289, 0.93569,
0.94178, 0.94718, 0.95187, 0.95585,
0.95912, 0.96166, 0.96348, 0.96458,
0.96494, 0.96458, 0.96348, 0.96166,
0.95912, 0.95585, 0.95187, 0.94718,
0.94178, 0.93569, 0.9289, 0.92144,
0.91331, 0.90452, 0.89508, 0.88501,
0.87432, 0.86303, 0.85114, 0.83869,
0.82567, 0.81212, 0.79805, 0.78347,
0.76842, 0.7529, 0.73694, 0.72056,
0.70378, 0.68663, 0.66912, 0.65129,
0.63314, 0.61472, 0.59603, 0.57711,
0.55798, 0.53867, 0.51919, 0.49958,
0.47985, 0.46004, 0.44016, 0.42025,
0.40033, 0.38042, 0.36054, 0.34072,
0.32099, 0.30136, 0.28187, 0.26253,
0.24336, 0.22439, 0.20564, 0.18713,
0.16887, 0.1509, 0.13323, 0.11587,
0.098854, 0.082187, 0.06589, 0.049979,
0.034469, 0.019373, 0.0047068, -0.0095179,
-0.023289, -0.036594, -0.049424, -0.061769,
-0.07362, -0.08497, -0.09581, -0.10613,
-0.11594, -0.12522, -0.13397, -0.14219,
-0.14988, -0.15703, -0.16365, -0.16973,
-0.17529, -0.18031, -0.18481, -0.18878,
-0.19223, -0.19518, -0.19761, -0.19955,
-0.201, -0.20196, -0.20245, -0.20247,
-0.20204, -0.20117, -0.19987, -0.19814,
-0.19601, -0.19349, -0.19058, -0.18731,
-0.18369, -0.17973, -0.17545, -0.17086,
-0.16599, -0.16083, -0.15542, -0.14977,
-0.14389, -0.1378, -0.13153, -0.12507,
-0.11846, -0.11171, -0.10483, -0.097851,
-0.090777, -0.08363, -0.076426, -0.069181,
-0.061911, -0.054632, -0.047361, -0.040111,
-0.032899, -0.02574, -0.018646, -0.011634,
-0.0047158, 0.0020949, 0.0087852, 0.015343,
0.021756, 0.028013, 0.034103, 0.040016,
0.045742, 0.051272, 0.056597, 0.061708,
0.066598, 0.071261, 0.075689, 0.079877,
0.083819, 0.087511, 0.090949, 0.094129,
0.097048, 0.099704, 0.1021, 0.10422,
0.10608, 0.10767, 0.109, 0.11006,
0.11086, 0.1114, 0.11168, 0.1117,
0.11148, 0.111, 0.11029, 0.10933,
0.10815, 0.10674, 0.10511, 0.10326,
0.10121, 0.098965, 0.096527, 0.093906,
0.091111, 0.088151, 0.085034, 0.081769,
0.078366, 0.074834, 0.071182, 0.067421,
0.063559, 0.059608, 0.055577, 0.051475,
0.047313, 0.043101, 0.038849, 0.034567,
0.030264, 0.025951, 0.021637, 0.017331,
0.013043, 0.008783, 0.0045589, 0.00037992,
-0.0037453, -0.0078082, -0.011801, -0.015715,
-0.019543, -0.023278, -0.026913, -0.030441,
-0.033855, -0.037149, -0.040318, -0.043356,
-0.046258, -0.04902, -0.051636, -0.054104,
-0.056419, -0.058578, -0.060579, -0.062418,
-0.064095, -0.065607, -0.066952, -0.068131,
-0.069142, -0.069986, -0.070662, -0.071171,
-0.071515, -0.071693, -0.071709, -0.071563,
-0.071259, -0.070799, -0.070186, -0.069423,
-0.068513, -0.067461, -0.06627, -0.064946,
-0.063491, -0.061913, -0.060214, -0.058401,
-0.056479, -0.054453, -0.05233, -0.050115,
-0.047814, -0.045434, -0.04298, -0.04046,
-0.037879, -0.035243, -0.03256, -0.029837,
-0.027078, -0.024292, -0.021485, -0.018663,
-0.015832, -0.013, -0.010172, -0.0073545,
-0.0045541, -0.0017767, 0.00097182, 0.0036857,
0.0063592, 0.0089869, 0.011563, 0.014084,
0.016543, 0.018935, 0.021258, 0.023505,
0.025673, 0.027758, 0.029757, 0.031665,
0.03348, 0.035199, 0.036818, 0.038336,
0.039751, 0.041059, 0.042261, 0.043353,
0.044336, 0.045208, 0.045968, 0.046616,
0.047153, 0.047577, 0.047889, 0.04809,
0.048182, 0.048163, 0.048037, 0.047805,
0.047468, 0.047028, 0.046488, 0.04585,
0.045116, 0.044289, 0.043372, 0.042369,
0.041281, 0.040114, 0.03887, 0.037553,
0.036167, 0.034716, 0.033203, 0.031633,
0.03001, 0.028338, 0.026622, 0.024866,
0.023074, 0.02125, 0.0194, 0.017527,
0.015636, 0.013732, 0.011818, 0.0098996,
0.0079806, 0.0060655, 0.0041584, 0.0022636,
0.00038505, -0.0014732, -0.0033072, -0.0051131,
-0.0068872, -0.008626, -0.010326, -0.011984,
-0.013596, -0.015159, -0.016672, -0.01813,
-0.019531, -0.020872, -0.022152, -0.023368,
-0.024518, -0.025601, -0.026614, -0.027556,
-0.028427, -0.029224, -0.029947, -0.030595,
-0.031168, -0.031665, -0.032086, -0.03243,
-0.032699, -0.032892, -0.03301, -0.033053,
-0.033023, -0.032919, -0.032744, -0.032498,
-0.032183, -0.0318, -0.031352, -0.030839,
-0.030263, -0.029627, -0.028933, -0.028183,
-0.027379, -0.026524, -0.02562, -0.02467,
-0.023677, -0.022642, -0.02157, -0.020663,
-0.019323, -0.018154, -0.016959, -0.015741,
-0.014502, -0.013246, -0.011975, -0.010694,
-0.0094039, -0.0081088, -0.0068114, -0.0055146,
-0.0042215, -0.0029347, -0.0016571, -0.00039149,
0.00085947, 0.0020932, 0.0033071, 0.0044987,
0.0056656, 0.0068055, 0.0079163, 0.0089957,
0.010042, 0.011052, 0.012026, 0.012961,
0.013855, 0.014707, 0.015516, 0.016281,
0.016999, 0.01767, 0.018294, 0.018869,
0.019395, 0.01987, 0.020295, 0.02067,
0.020993, 0.021265, 0.021486, 0.021657,
0.021776, 0.021845, 0.021864, 0.021833,
0.021754, 0.021626, 0.021452, 0.021231,
0.020966, 0.020656, 0.020304, 0.01991,
0.019476, 0.019004, 0.018495, 0.01795,
0.017372, 0.016761, 0.01612, 0.015451,
0.014755, 0.014034, 0.01329, 0.012526,
0.011742, 0.010941, 0.010125, 0.0092957,
0.0084555, 0.0076062, 0.0067497, 0.0058881,
0.0050234, 0.0041574, 0.0032921, 0.0024295,
0.0015714, 0.00071962, -0.00012395, -0.00095756,
-0.0017795, -0.002588, -0.0033816, -0.0041585,
-0.0049174, -0.0056567, -0.006375, -0.0070711,
-0.0077436, -0.0083913, -0.0090132, -0.0096082,
-0.010175, -0.010714, -0.011222, -0.011701,
-0.012148, -0.012564, -0.012948, -0.0133,
-0.013619, -0.013904, -0.014157, -0.014376,
-0.014561, -0.014714, -0.014833, -0.014919,
-0.014972, -0.014993, -0.014981, -0.014938,
-0.014864, -0.014759, -0.014624, -0.01446,
-0.014268, -0.014048, -0.013801, -0.013528,
-0.01323, -0.012909, -0.012564, -0.012198,
-0.01181, -0.011404, -0.010979, -0.010536,
-0.010078, -0.0096053, -0.0091191, -0.0086207,
-0.0081116, -0.0075928, -0.0070659, -0.006532,
-0.0059925, -0.0054486, -0.0049017, -0.0043529,
-0.0038035, -0.0032548, -0.002708, -0.0021642,
-0.0016245, -0.0010903, -0.00056238, -4.2025e-005,
0.0004698, 0.00097207, 0.0014638, 0.0019441,
0.002412, 0.0028667, 0.0033074, 0.0037332,
0.0041434, 0.0045374, 0.0049144, 0.0052738,
0.0056152, 0.0059379, 0.0062415, 0.0065255,
0.0067897, 0.0070336, 0.007257, 0.0074597,
0.0076415, 0.0078022, 0.0079419, 0.0080605,
0.008158, 0.0082345, 0.0082901, 0.008325,
0.0083395, 0.0083337, 0.0083081, 0.0082628,
0.0081984, 0.0081152, 0.0080136, 0.0078942,
0.0077575, 0.007604, 0.0074342, 0.0072487,
0.0070483, 0.0068334, 0.0066047, 0.006363,
0.0061088, 0.005843, 0.0055661, 0.005279,
0.0049823, 0.0046769, 0.0043634, 0.0040427,
0.0037155, 0.0033826, 0.0030449, 0.0027029,
0.0023577, 0.0020099, 0.0016604, 0.0013099,
0.00095913, 0.00060893, 0.00026005, -8.6808e-005,
-0.00043093, -0.00077158, -0.0011081, -0.0014398,
-0.0017661, -0.0020863, -0.0023999, -0.0027064,
-0.0030051, -0.0032955, -0.0035773, -0.0038499,
-0.0041129, -0.004366, -0.0046088, -0.004841,
-0.0050622, -0.0052722, -0.0054708, -0.0056577,
-0.0058327, -0.0059957, -0.0061465, -0.006285,
-0.0064111, -0.0065247, -0.0066258, -0.0067144,
-0.0067904, -0.0068539, -0.0069049, -0.0069435,
-0.0069698, -0.0069839, -0.006986, -0.0069763,
-0.0069549, -0.0069221, -0.0068781, -0.0068232,
-0.0067576, -0.0066818, -0.006596, -0.0065006,
-0.006396, -0.0062825, -0.0061605, -0.0060306,
-0.005893, -0.0057482, -0.0055967, -0.0054388,
-0.0052751, -0.0051058, -0.0049315, -0.0047527,
-0.0045696, -0.0043828, -0.0041926, -0.0039995,
-0.0038038, -0.0036058, -0.0034062, -0.0032051,
-0.0030029, -0.0028001, -0.0025969, -0.0023938,
-0.0021911, -0.0019891, -0.0017882, -0.0015887,
-0.001391, -0.0011953, -0.0010021, -0.00081155,
-0.00062408, -0.00043984, -0.00025933, -8.2619e-005,
8.9978e-005, 0.00025803, 0.00042169, 0.00058038,
0.00073402, 0.00088249, 0.0010257, 0.0011633,
0.0012953, 0.0014217, 0.0015423, 0.0016571,
0.001766, 0.0018691, 0.0019663, 0.0020578,
0.0021434, 0.0022233, 0.0022975, 0.002366,
0.0024291, 0.0024866, 0.0025387, 0.0025854,
0.0026269, 0.0026631, 0.0026943, 0.0027203,
0.0027414, 0.0027577, 0.0027687, 0.0027755,
0.0027774, 0.0027746, 0.0027673, 0.0027558,
0.0027399, 0.0027199, 0.0026957, 0.0026679,
0.0026363, 0.0026013, 0.0025628, 0.0025212,
0.0024766, 0.0024295, 0.0023799, 0.0023281,
0.0022743, 0.0022187, 0.0021616, 0.0021034,
0.002044, 0.0019841, 0.0019233, 0.0018625,
0.0018011, 0.0017398, 0.0016795, 0.0016174,
0.0015577, 0.001498, 0.0014384, 0.0013791,
0.0013208, 0.0012634, 0.0012067, 0.0011503,
0.0010947, 0.0010396, 0.00098539, 0.00093197,
0.00087945, 0.00082761, 0.00077656, 0.0007262,
0.00067673, 0.00062815, 0.00058068, 0.00053431,
0.00048927, 0.00044552, 0.00040326, 0.0003625,
0.00032345, 0.00028614, 0.00025079, 0.00084549 };
```