Math behind cl-audio-downsample library.

shamaz.mazum

March 15, 2018

1 Fourier transform

The Fourier transform of function $f(x)$ is a function $F(\xi)$ defined by

$$F(\xi) = \int f(x)e^{ix\xi}dx \tag{1}$$

and its inverse transform is defined by

$$f(x) = \frac{1}{2\pi}\int F(\xi)e^{-ix\xi}d\xi \tag{2}$$

Where direct and inverse transforms exist is a really hard question to be discussed here, so I will only say that if the inverse transform exists for a particular $F(\xi)$ then $F(\xi)$ is decaying to zero when $|\xi|$ moves toward infinity, in other words, $\lim_{|\xi|\to\infty} F(\xi) \to 0$. I also will use Fourier transform operator $Ff(x)$ which means "apply Fourier transform to a function $f(x)$". Fourier transform has some useful properties, some of them I describe below:

Linear property $F[af(x)+bg(x)] = aFf(x)+bFg(x)$. This is because of linear property of integral.

Shift property $Ff(x-a) = e^{ia\xi}Ff(x)$. You can replace $x' = x-a$ in the integral to prove it.

Image multiplication property $F[f*g(x)] = Ff(x)Fg(x)$.

$f*g$ above is called convolution and is defined as the following:

$$f*g(x) = \int_{-\infty}^{\infty} f(\tau)g(x-\tau)d\tau \tag{3}$$

or if your functions equal to zero when $x<0$

$$f*g(x) = \int_{0}^{x} f(\tau)g(x-\tau)d\tau \tag{4}$$

The Fourier transform is of great importance in digital signal processing because you can think of $exp(ix\xi)$ as a harmonic oscillator with frequency ξ. You can write $exp(iw) = cos(w) + isin(w)$.

2 Bandlimiting a signal

Imagine now an oscillator which oscillates uniformly at all frequencies $|\xi| < \omega_0$. Its Fourier transform will be

$$F(\xi) = \begin{cases} 1 & \text{if } |\xi| < \omega_0, \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

The inverse transform is $f(x) = \frac{sin(x)}{\pi x}$. There is a Shannon sampling theorem which says that every bandlimited signal (i.e. it has no frequencies above some $f_0 = \frac{\omega_0}{2\pi}$) can be represented as shifted versions of functions $\frac{sin(\omega_0 x)}{\pi x}$ multiplied by samples of that signal taken at frequency $2f_0$ Hz. That means if you want to store a signal which contains, say, sine waves at frequency at most 1kHz, you need to take samples at frequency 2Hz (or higher). If you choose frequency less than $2f_0$ then some of high frequency data will be ”misinterpreted” as low frequency data.

3 Signal filtering

So if you want to downsample your audio data taken with sampling frequency, say, 96kHz to sampling frequency 48kHz, taking every second sample is not enough! If you do so, all inaudible data in the range between 24 and 48 Hz will be misinterpreted and become audible. This thing is called aliasing. So before dropping any data you need to filter it to cut all oscillations above $newsamplerate/2$. To do so we will use the image multiplication property of the Fourier transform and convolve our signal with $\frac{sin(\omega_0 x)}{\pi x}$ using (4) (provided that our signal begins from time $x = 0$). There is a problem with that, although. When convolving numerically, you need to use **all** samples in you signal to get a new value for **one** sample. That’s because support of $\frac{sin(\omega_0 x)}{\pi x}$ (i.e. set of values where the function does not equal zero, with possible exception of a set with measure zero) is the whole set of real numbers, $\mathbb{R}$. What we need is a filtering function with **compact support** or in the case of functions defined of $\mathbb{R}$, **bounded support**. Unfortunately, these functions cannot act as an ideal low-pass filter, but they can be close to it. Usually people use the same $\frac{sin(x)}{\pi x}$ function multiplied by some

window function to make its support compact, but I propose a different approach. I do not know, if it is known to anybody (most likely to), in this case I just ”reinvented” it on my own.

4 Inverse Fourier transform of some ”non-invertable” functions

Can you invert a Fourier image, say, $F(\xi) = 1$? No, you cannot. That's because it does not die out when $|\xi|$ moves toward infinity. But you still can ”nominate” some ”function” to be its inverse transformed version! To understand it you will need a concept of weak convergence. For a functions $f \in H$, where H is some Hilbert space, a sequence $f_n \in H$ is said to be convergent to f if $\lim_{n\to\infty} \langle f_n, g\rangle = \langle f, g\rangle$ for all $g \in H$. It is denoted as $\{f_n\} \to^w f$.

$\langle \cdot, \cdot \rangle$ here is a **scalar** or **dot product** on that space H. If you do not know that it means, remember orthogonal vectors on $\mathbb{R}^2$, i.e. vectors for which the angle between them is 90 degrees. Scalar product of these vectors is zero. It is some measure of ”independence” in the sense that if angle between two lines is changed from zero to 90 degrees, the scalar product will change from some value to zero. Scalar product of vectors ϕ and ψ on $\mathbb{R}^N$ is $\langle \phi, \psi \rangle = \sum_{i=1}^{N} \phi_i \psi_i$. $\mathbb{R}^N$ is an example of a Hilbert space. It turns out, that scalar product can be defined not only on $\mathbb{R}^N$ but on spaces of functions. A most important example is $L^2(\mathbb{C})$ space, i.e. space of functions, for which exists a Lebesgue integral $\int_{-\infty}^{\infty} |f(x)|^2 dx$. For these functions there always exists an integral $\langle f, g\rangle = \int_{-\infty}^{\infty} f(x)\overline{g(x)}dx$, $f, g \in L^2(\mathbb{C})$. This is how scalar product can be defined for functions. There are also orthogonal functions on L^2. If functions in some set are orthogonal to each other and $\langle f, f\rangle = 1$ this set is called a set of **orthonormal** functions. L^2 is an another example of a Hilbert space. There are some sets of orthonormal functions on a Hilbert space called **basis**. You can represent any function on that space as a sum of basis functions: $f(x) = \sum_i \langle f, \phi_i\rangle \phi_i$. Another important concept is **norm** denoted as $\|\cdot\|$. In $\mathbb{R}^N$ it tells how long a vector is. For L^2 a norm is defined as $\|f(x)\| = \sqrt{\langle f, f\rangle}$.

Back to weak convergence. Take functions $f_n(x) = \frac{sin(nx)}{\pi x}$. They all belong to L^2 with norm equals to $\sqrt{n/\pi}$. As you can see, if n rises, norm of $f_n(x)$ also will rise, so $f_n(x)$ is divergent (there cannot be an element with infinite norm). But what happens to a sequence $\langle f_n, \phi\rangle$ when $n \to \infty$ where $\phi(x)$ is any function from

$L^2(\mathbb{R})$? Follow this trick:

$$\begin{aligned} \lim_{n\to\infty}\int_{-\infty}^{\infty} f_n(x)\phi(x)dx = \\ \lim_{n\to\infty}\int_{-\infty}^{\infty} \frac{sin(nx)}{\pi x}\phi(x)dx = \\ \phi(0)\lim_{n\to\infty}\int_{-\infty}^{\infty}\frac{sin(nx)}{\pi x}dx + \lim_{n\to\infty}\int_{-\infty}^{\infty} sin(nx)\frac{\phi(x)-\phi(0)}{\pi x} = \\ \phi(0)\lim_{n\to\infty}\int_{-\infty}^{\infty}\frac{sin(nx)}{\pi x}dx + \int_{-\infty}^{\infty}\lim_{n\to\infty} sin(nx)\frac{\phi(x)-\phi(0)}{\pi x} \end{aligned} \tag{6}$$

Integration and limit switched places because integrated function does not have any irregularities at $x = 0$ now. It can be proven that for a "good enough" function $f(x)$ an integral $\int_{-\infty}^{\infty} sin(nx)f(x)dx$ converges to zero as $n \to \infty$. On the other hand $\int_{\infty}^{\infty}\frac{sin(nx)}{\pi x}$ is always 1, no matter what value n takes, so

$$\lim_{n\to\infty}\int_{-\infty}^{\infty} f_n(x)\phi(x)dx = \phi(0) = \int_{\infty}^{\infty}\delta(x)\phi(x)dx \tag{7}$$

$\delta(x)$ is called a **delta-function** and is defined simply as

$$\int f(x)\delta(x)dx = f(0) \tag{8}$$

Surely, it's no a "real" function, but rather a functional on Hilbert space $L^2(\mathbb{R})$, i.e. $\langle\delta,\cdot\rangle$ takes a function and returns its value at point $x = 0$. Also you can define $\langle\delta(x-a), f(x)\rangle = f(a)$. You can see that by substitution $x' = x + a$.

Another very important property on delta-function is that $f*\delta = f$, i.e. convolution of any function with delta is that function. For some functions you can define so-called **convolution algebra** where $\delta(x)$ is like 1 in "ordinary" algebra and convolution of functions is like multiplication of numbers.

Speaking of Fourier transform, $F[\frac{sin(\omega_0 x)}{\pi x}](\xi)$ is (5). If $n \to \infty$, the Fourier image becomes simply $F(\xi) = 1$. Inverse transform of that $F(\xi)$ becomes $\delta(x)$. If you want to pass your signal through such a filter (it's like identity filter, which does not change your signal), you must convolve it with $\delta(x)$: $f * \delta = f(x)$. So as you can see, your signal is not changed indeed.

In case of discrete signals continuous convolution is replaced by discrete convolution. $f = \phi * \psi$ becomes

$$f_n = \sum_{i=-\infty}^{\infty}\phi_i\psi_{n-i} \tag{9}$$

Surely, in practice infinity is replaced by some finite l, known as, if I am right, a **filter order**.

5 My filter

In the (9) let $\phi_i = \phi_{-i}$. Let's now imagine, that our filter is itself a function, sampled at infinitely high sampling rate, while your signal is sampled at frequency $f_{ref} = \frac{\omega_{ref}}{2\pi}$. Let a filter function be defined as:

$$\phi(x) = \sum_{n=-N}^{N} \phi_n \delta(x + n\frac{2\pi}{\omega_{ref}}) \tag{10}$$

where ϕ_n are some coefficients. Its Fourier transform, $\Phi(\xi)$ is

$$\Phi(\xi) = \sum_{n=-N}^{N} \phi_n e^{in\frac{2\pi}{\omega_{ref}}\xi} \tag{11}$$

I used the linear and shift properties of Fourier transform and the fact that $F[\delta(x)] = 1$.

The equation above can be rewritten if members like $exp(ia\xi)$ and $exp(-ia\xi)$ are combined together. Remember, that $\phi_n = \phi_{-n}$.

$$\begin{gathered} \sum_{n=-N}^{N} \phi_n e^{in\frac{2\pi}{\omega_{ref}}\xi} = \\ \phi_0 + \sum_{n=1}^{N} \phi_n (e^{in\frac{2\pi}{\omega_{ref}}\xi} + e^{-in\frac{2\pi}{\omega_{ref}}\xi}) = \\ \phi_0 + \sum_{n=1}^{N} \frac{\phi_n}{2} cos(n\frac{2\pi}{\omega_{ref}}\xi) \end{gathered} \tag{12}$$

Good, functions like 1 and $cos(n\frac{2\pi}{\omega_{ref}}\xi)$ with different n are orthonormal to each other on a segment $[-\omega_{ref}/2, \omega_{ref}/2]$. They constitute a basis on that segment for functions called **even functions**, that is function $f(x)$ is even if $f(x) = f(-x)$. Our ideal filter (5) is an even function. You can find ϕ_n calculating a scalar product of $F(\xi)$ in (5) and functions from our basis set and normalising. Introduce a variable $a \leq 1$, so $\omega_0 = a\omega_{ref}/2$. We get the following:

$$\begin{gathered} \phi_0 = \frac{2}{\omega_{ref}} \int_0^{a\omega_{ref}/2} d\xi = a \\ \phi_n = \frac{4}{\omega_{ref}} \int_0^{a\omega_{ref}/2} cos(n\frac{2\pi}{\omega_{ref}}\xi) d\xi = \frac{2}{n\pi} sin(a\pi n) \end{gathered} \tag{13}$$

A sequence $\{\phi_0, \phi_1/2, \phi_2/2, \cdots\}$ will constitute our filter. Note, that these coefficients do not depend on ω_{ref}. If you convolve this sequence with your signal by (9), you will get a new signal, in which

all frequencies above $a\omega_{ref}/2$ are cut (the original signal has frequencies up to $\omega_{ref}/2$). The process of finding $\{\phi_0, \phi_1, \phi_2, \cdots\}$ is called **Fourier series** decomposition, not to be confused with Fourier transform. Note, that Fourier transform of our filter is real-valued and you will not get any phase shift in frequencies below cutpoint, and sometimes phase shift of π above it, but who cares?

Good enough? No. An ideal filter has a discontinuity at $|\xi| = \omega_0 = a\omega_{ref}/2$. Fourier series decomposition behaves very badly in the area around points of discontinuity. Even if you choose N to be large enough, that "area of bad behaviour" will be smaller and smaller, but bad behaviour will not vanish at all. It's known as **Gibbs phenomenon**. Bad behaviour expresses itself as big oscillations around points of discontinuity. We can eliminate that behaviour by introducing a small transition region between preserved frequencies and cut frequencies. To do this, let's find a function which is:

- even;
- has a differential in all points belonging to $[0, 1]$
- equals to 1 at 0 and 0 at 1
- its differential equals to 0 at 0 and 1

or, speaking math, we need to find such $f(x)$, so

- $f(x) = f(-x)$;
- $f(x) \in C[0, 1]$
- $f(0) = 1$ and $f(1) = 0$
- $f'(0) = 0$ and $f'(1) = 0$

One function which satisfies our demands is $f(x) = x^4 - 2x^2 + 1$. Then you can substitute $x = \frac{\xi - (a\omega_{ref}/2 - \epsilon)}{2\epsilon}$ and get a new filter:

$$F(\xi) = \begin{cases} 1 & \text{if } |\xi| < a\omega_{ref}/2 - \epsilon, \\ x^4 - 2x^2 + 1 & \text{if } a\omega_{ref}/2 - \epsilon \leq |\xi| \leq a\omega_{ref}/2 + \epsilon, \\ 0 & \text{otherwise.} \end{cases} \tag{14}$$

It has a transition region which width is 2ϵ, but it has differential in all its point. Fourier series for it will converge much faster and without Gibbs phenomenon.

I use $f(x)$ with some additional requirements:

- $f''(0) = 0$ and $f''(1) = 0$

- $f'''(0) = 0$ and $f'''(1) = 0$

This is my $f(x)$: $f(x) = 4x^{10} - 15x^8 + 20x^6 - 10x^4 + 1$. Fourier series decomposition will converge even faster to this function when $n \to \infty$. Although decomposition coefficients can be calculated explicitly, I prefer to compute them numerically by the **Simpson's formula** which is a method of precise numerical integration.

6 Error estimation

It also would be great to know how close Fourier series decomposition is to (14) or to another filter with another polynomial. Remember, that decomposed signal is in the following form:

$$F(\xi) = \phi_0 + \sum_{n=1}^{N} \frac{\phi_n}{2} cos(\frac{2\pi}{\omega_{ref}} n\xi) \tag{15}$$

A global minimum of that is somewhere "to the right" of the frequency $\xi = \omega_0 + \epsilon$ which is $\xi = a\omega_{ref}/2 + \epsilon$. I suggest the value $|F(\xi_{min})|$ as an error measure. To find a minimum of $F(\xi)$ you need to find its differential and equate it to zero. You will get an equation (a constant multiplier, $\frac{\pi}{\omega_{ref}}$, is dropped):

$$F'(\xi) = \sum_{n=1}^{N} \phi_n n sin(\frac{2\pi}{\omega_{ref}} n\xi) = 0 \tag{16}$$

As I said, we will search for a solution on a segment $\xi \in [\omega_0 + \epsilon, \omega_{ref}/2]$. We need only one solution, which corresponds to the global minimum. $F(\xi)$ is decreasing in the range $\xi \in [\omega_0 - \epsilon, \omega_0 + \epsilon]$ and continues to decrease for some $\xi \geq \omega_0 + \epsilon$ "by inertia". So, $F'(\xi)$ is negative in this range and equals to zero in the point where $F(\xi)$ is minimal. $F'(\xi)$ behaves in that region almost linearly, so you can find a tangent to it in the point $\xi = \omega_0 + \epsilon$ and find out where it becomes a zero. This is a first step of so called **Newton's method** of numerically solving equations like $f(x) = 0$ for monotonous $f(x)$. I will restrict myself to that only step, because, as I said, $F'(x)$ behaves almost linearly in the region of interest.

Define b, so that $\epsilon = ab\omega_{ref}/2$. Then our tangent is defined as follows:

$$T(\xi) = A(\xi - \omega_0 - \epsilon) + B \tag{17}$$

with $\xi_{min} - \omega_0 - \epsilon = -B/A$. I calculated B/A to be

$$B/A = \frac{\sum_{n=1}^{N} \phi_n n sin(\pi na(1+b))}{\sum_{n=1}^{N} \phi_n n^2 cos(\pi na(1+b))} \tag{18}$$

An estimated minimum value of $F(\xi_{min})$ is

$$F(\xi_{min}) = \phi_0 + \sum_{n=1}^{N} \phi_n cos(\pi n(a + b - \pi B/A) \tag{19}$$

On (1) you can see a graph which represents an error estimation for two types of filters, one of them being (14) (solid line) and another is the tenth order polynomial which I use (dash line).

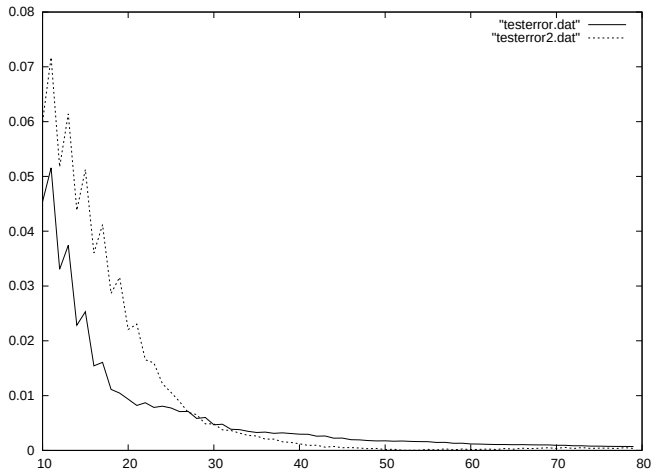


Figure 1: The error $|F(\xi_{min})|\,(N)$ of Fourier series decomposition.

Below on (2) I also include a Fourier transform and a partial sum of the first 30 members of its Fourier series decomposition for my filter with $a = 1/2, b = 1/7$. Practically, it's better to use $N > 50$ to get more or less acceptable quality.

7 Downsampling in cl-audio-downsample

So, after your signal is filtered to cut all frequencies which cannot be saved with new sampling rate (suppose, you choose a filter with parameter a), you can just pick every a^{-1}-th sample, where a^{-1} is an integral ratio of an old sampling rate to a new sampling rate, $a^{-1} = f_{ref_{old}}/f_{ref_{new}}, a^{-1} \in \mathbb{N}$. Also you can resample to any sampling rate with the ratio $N/M = f_{ref_{old}}/f_{ref_{new}}$ first inserting $N - 1$ zeros between your original samples, when filtering that signal with $a = 1/M$, when taking every M-th sample from the result. But there is more efficient methods to do such a resampling, so my library supports only downsampling to an integral ratio of sampling

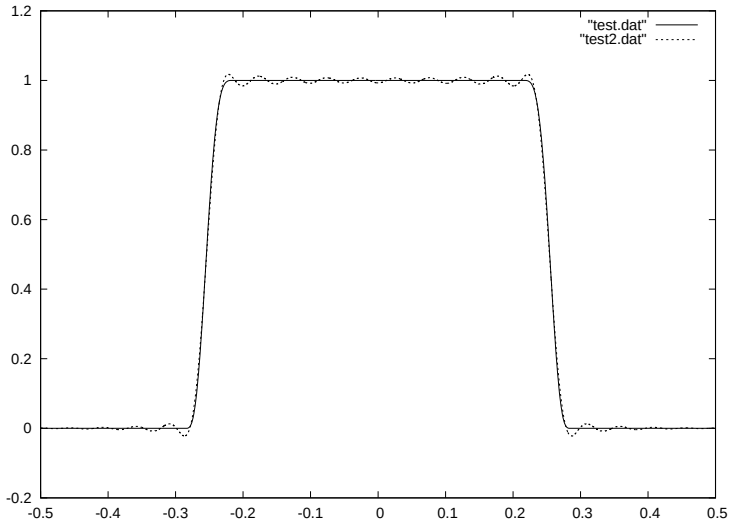


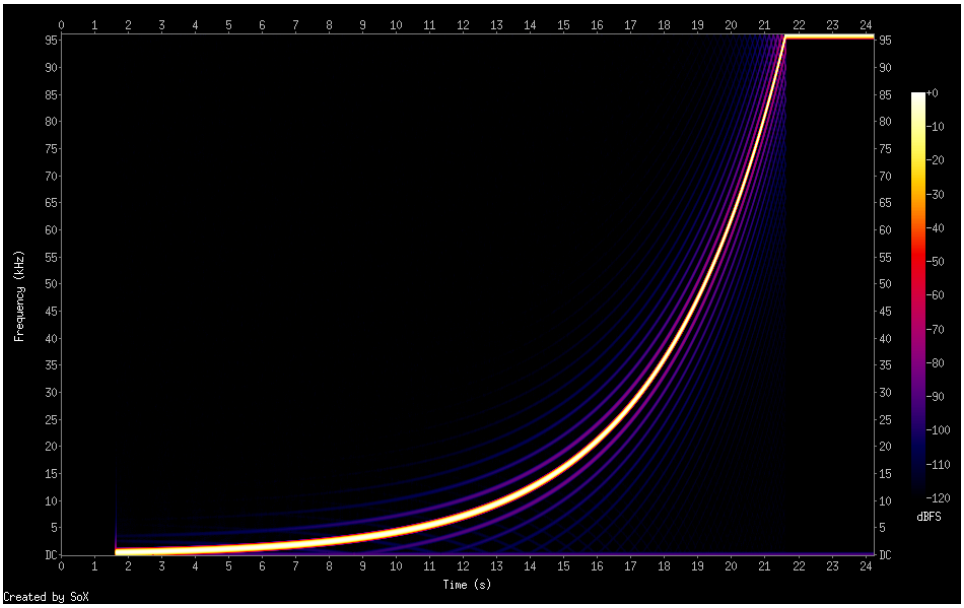
Figure 2: The Fourier transform of my filter (solid) and its Fourier series decomposition (the first 30 members, dashed line).

frequencies (e.g. from 192kHz to 48kHz, or from 96KHz to 48kHz). Surely, it can be extended to support any ratio, but why bother?

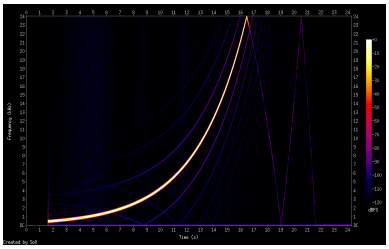
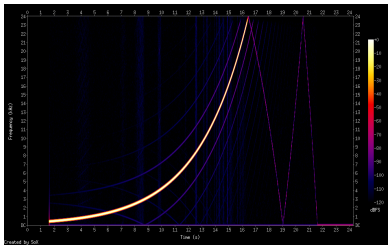
8 Quality

In this section I will provide some comparison data as some measure of quality. I compared my resampler with swr, a standard resampler for FFmpeg. I used a sine wave with a frequency starting at 440Hz and increasing to 96kHz at exponential rate as an input signal. The input signal is taken with sampling rate 192kHz and its downsampled version has a sampling rate 48kHz. The signal was generated with SuperCollider, and spectrograms below was generated by SoX. My library easy-audio was used to read from and write to wav audio files.

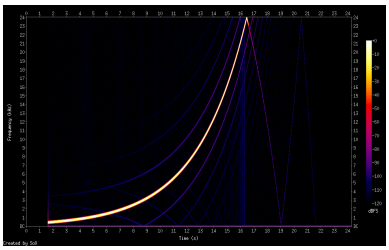
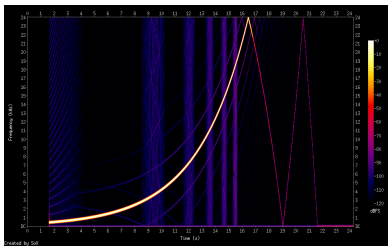
(3) is a spectrogram of the original sine wave (up) followed by spectograms of resampled versions (bottom). I used two settings for my resampler: with a transition range $b = 1/7$ and a fixed filter order $N = 50$ and with a transition range $b = 1/20$ and an error 0.0001. Settings which use fixed N can be considered ”fast” settings, and settings with ”floating” N and a particular error can be considered the ”best”.



(a) A spectrogram of the original wave.

(b) Some spectrograms of the signal resampled with FFmpeg's swr

(c) The signal resampled with cl-audio-downsample with the parameters $b = 1/20, error = 0.0001$ (left) and $b = 1/7, N = 50$ (right).

Figure 3: Spectrograms of the original signal and its downsampled versions.

9 Conclusion

Although cl-audio-downsample shows test results comparable to FFmpeg's swr, it's no much for soxr resampler. It also has im-

plementation limitations, e.g. it cannot downsample from 96kHz to 44.1kHZ (only to 48KHz). What's more important, is that I observed that aliasing does not vanish much when you choose $b : 0 < b < 1/20$ with some constant error. I cannot explain it now, maybe it's just a computational error. It's also not very fast, but this is most certainly a flaw of the implementation.