Custom Graphic Equalizer Filter

Carlos Rafael Gimenes das Neves

26.27.28 SEPT 2013
Cictem 2013

BandBBBTec

Allow me to start by telling a short story…

In the beginning

1998!

there was Winamp!

Cool! What kind of plugin can I create for it?!

Hey, its equalizer doesn’t follow ISO standards!

Back in 1998!

60 170 310 600 1k 3k 6k 14k 16k

That’s it! I will create a new equalizer for Winamp!

However… Hold on a second…

Winamp already has plenty of good equalizer plugins!

And most important…

What are equalizers made of?!?!

Considering those points…
Why sticking to the idea…?

Considering those points…
Why sticking to the idea…?

Any reason is a good reason to study and listen to some music!

So, let's get going!
Straight to the studies!

What could be so difficult, anyway?

Better think again!

Band-pass filter

Q Factor

Frequency domain

Blackman Window

Hanning Window

Low-pass filter

Overlap-add

FIR Filters

Time domain

Convolution

Step Response

Impulse Response

IIR Filters

Fourier Transform

A “while” later, I was at least able to understand what all that DSP jargon meant

A "while" later, I was at least able to understand what all that DSP jargon meant

Still far from an equalizer, though...

Much later, an equalizer came out…

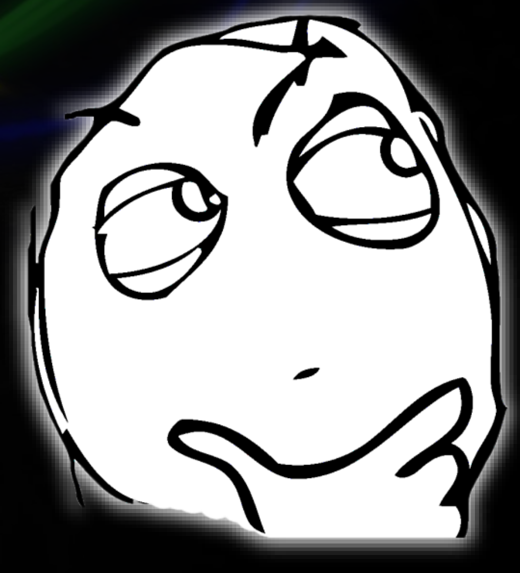
…well, a sum of 10 isolated band-pass IIR filters, actually

Going deeper into DSP, I was introduced to FIR filters, and the whole concept of convolution

Which, in turn, led me to Fourier Transform

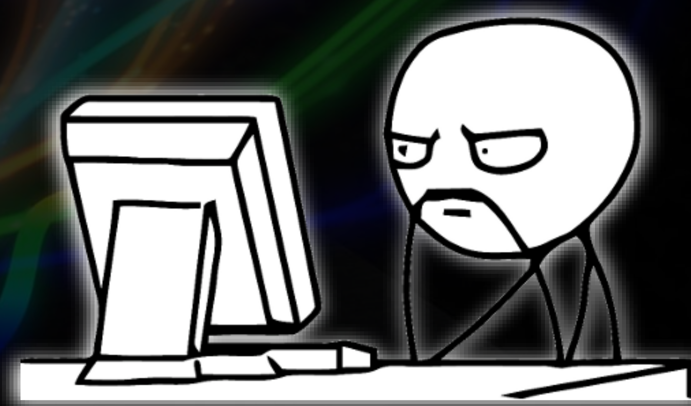
That’s when an idea struck me

If I’m already going to transform the audio into frequency domain, why not allowing the user to adjust more than 10 bands?



That way the users would be given more power to fine tune their equalizer!

It took a lot of time, and also a lot of studying, but the result finally came out a few years ago!



Creating the filter

Obtain an array of amplitudes from the user

→ Stipulate L, the filter's length (L must be a power of 2 for the FFT to work)

→ Map those amplitudes into M + 1 complex numbers (M = L / 2)

→ Apply the inverse FFT on the complex numbers to obtain the filter's response in time domain

→ Apply the desired window function on the first M + 1 samples, zeroing out the rest (to smooth the filter, improving the overlap-add method)

→ Apply the FFT on the samples to obtain the filter's response in frequency domain again!

Creating the filter

1. Obtain an array of amplitudes from the user
2. Stipulate L, the filter's length (L must be a power of 2 for the FFT to work)
3. Map those amplitudes into M + 1 complex numbers (M = L / 2)
4. Apply the inverse FFT on the complex numbers to obtain the filter's response in time domain
5. Apply the desired window function on the first M + 1 samples, zeroing out the rest (to smooth the filter, improving the overlap-add method)
6. Apply the FFT on the samples to obtain the filter's response in frequency domain again!

We need the filter to be in frequency domain to speed up the convolution process!

Applying the filter

Fill an array called DATA with M audio samples + M zeroes

Apply the FFT on DATA, which results in M + 1 complex numbers

Multiply the filter by the M + 1 complex numbers

Apply the inverse FFT on the complex numbers, obtaining a filtered version of DATA

Add the M samples from a secondary array, called LAST, to the first M samples of DATA

Copy the last M samples from DATA into LAST, and repeat the whole process as necessary!

Applying the filter

Fill an array called DATA with M audio samples + M zeroes

Apply the FFT on DATA, which results in M + 1 complex numbers

Multiply the filter by the M + 1 complex numbers

Apply the inverse FFT on the complex numbers, obtaining a filtered version of DATA

Add the M samples from a secondary array called LAST to the first M samples of DATA

Copy the last M samples from DATA into LAST, and repeat the whole process as necessary!

LAST is used to perform the overlap-add technique, which allows us to smoothly apply a filter to a much longer signal

Enough talk!

Let’s cut to the demonstration!

The source code is available at:

https://github.com/carlosrafaelgn/GraphicalFilterEditor

The project can be tested at:

carlosrafaelgn.com.br/GraphicalFilterEditor

Thank you!!!

Questions?!
Suggestions?!

Leather texture: fantasystock.deviantart.com/art/Cracked-Leather-Texture-1-66541079
Light waves: csys-279.deviantart.com/art/Light-Wave-Wallpaper-193489523