The Frequency Domain And Diamond Cut

When we did this article last month on Understanding the Time and Frequency domain in Diamond Cut, we didn't know if our readers would care about this more technical type of information. In fact, we received more emails and calls about this article than any other one we've done in our newsletter. We love getting these notes of encouragement and thank you for writing. It seems that Diamond Cut users want to know a bit more about the technology "under the hood" so here's the second installment on this subject.

If you remember, the time domain display is simply the waveform graphic that you see whenever you open a file or record audio in DIAMOND CUT or Live. The time domain display really only shows two items of information - the amplitude of the signal at a point (this is the vertical axis) and the time of a point which is the horizontal axis. This is the display you use all the time and it should be well understood by just about all users.

However, the time domain display shows nothing about frequency information for complex signals. What is a complex signal, you say? It's any signal that contains multiple frequencies such as speech, music, noise, etc. Go ahead and do this - open up your favorite Monkees song and zoom in on the audio. You'll note that you can see the waveform quite well, but there's no way you can see the singing, drums or noise - it's all mixed together into a single waveform that you can see. The computer has exactly the same problem in that it cannot "see" all the frequency information in the time domain either. Remember this point - a complex signal such as music contains many frequencies and a sine wave contains only one frequency.

Now lets turn to the shadowy, surreal, smoky world of the frequency domain. We have created a sample file for this exercise that you can download and use. Click here to download the 500K Wave file and go ahead and open it up with Diamond Cut.

http://www.tracertek.com/waves/continuousnoisesample.wav

Listen to the file. Notice it has pretty pronounced 60hz hum in it. However, that hum is all mixed up with the rest of the audio. If you zoom in using the time domain, you will not be able to see the hum. To attack this hum, we'll use our Continuous Noise Filter which is a frequency domain filter. Bring up this filter and select a sample of the audio at the beginning where the hum exists by itself. Click on the Sample Noise button. The red line that is drawn lets you look the frequency domain right in the eye. The horizontal axis is frequency and the vertical axis is amplitude.

Surprise! We fooled you. The hum is not at 60hz, but is instead at 120hz. Make sure you can read this on the display. The Sample Noise function has taken the complex signal and has broken it down into its individual frequencies. It should be obvious to you that the hum is at 120hz or so. You can also see all the other frequencies that make up the complex signal you selected. You are now peering into the dark depths of the frequency domain.

If you remember our last newsletter, we touched on the fact that we could convert a time domain signal into a frequency domain signal. This is just what has happened here. This bit of magic is done by a highly mathematical algorithm called an FFT - which stands for Fast Fourier Transform. We don't need to understand the math that's behind this, we just need to know what it does.

And what is does is to analyze a complex signal and break it down into its component frequencies and their corresponding amplitudes (actually, we also store a third component of information for each frequency which is its phase, which could be another series of articles). Once we have done this, we are no longer dealing with a complex signal, but a bunch of sine waves. We can do lots of things with sine waves, including subtract them. The red line you see here is just the Reader's Digest version, however. It shows you in a single display the distribution of the major frequencies throughout the audio slice you have sampled.

Mr. FFT, however, has the full and unabridged version of the FFT results in its memory. It knows exactly what frequencies exist in our sample and at exactly what amplitudes. It knows, for example, that exactly 432.6hz exists at exactly 22db. You can't see this kind of detail on the display. And you don't need to.

If noise reduction were easy to do, we'd click on Preview or Run Filter and the Continuous Noise filter would subtract exactly the specific frequencies and amplitudes that make up the noise sample and we'd have a great and pretty noise-free file. This is the method used by some low level noise reduction programs and it results in various problems including the famous calliope sounding digital artifact. Can you think of reasons why this might be? Some folks might like this sound, but we don't, so the Diamond Cut and Live Continuous Noise filter contains far more sophistication to reduce the chance of this.

Notice the FFT dropdown box in the Continuous Noise filter. You'll find selections from 256 to 8192 in this box. This relates to a new concept called Frequency Bins or Frequency Buckets - just different common names for the same thing. Basically, we can't use our FFT to simply break down the entire audio spectrum each minute component part. Your computer wouldn't be fast enough for this.

Instead we look at frequency bins. Each bin contains a small portion of the audio spectrum. You can figure out the size of a Bin by taking the FFT number and first dividing it by 2. The reason for this division is beyond the scope of this article, but trust us, this is what you do. Let's take the 2048 setting as an example.

If we divide it by 2, we end up with 1024 frequency bins. These bins make up our entire audio spectrum of 20hz to 20khz. This works out to roughly 20,000 integer frequencies (20,21,22, . . . 19998, 19999, 20000). So if we divide our 1024 bins into 20,000, we get that each bin handles about 19hz. Now look at the horizontal axes in the continuous noise filter. Imagine that it’s not a smooth progression from left to right of 20hz to 20khz, but instead is made up of 1024 little frequency windows each being about 19hz wide.

By using these frequency Bins, we are able to use our FFT in a much more efficient way. We can analyze and store only a single result for each bin and therefore simplify the calculations we do - which is still quite a lot and would not have been possible on many computers even a few years ago.

Notice as well that the higher the FFT number, the smaller the frequency range for each bin. At a setting of 8096 we end up with only about 5hz in each bin and we have 4096 bins. At a setting of 128 we have only 64 bins and each bin covers 312hz. Since only an aggregate result is stored in each bin, this wide bin won't have as much resolution.

It seems that just about everything in the universe is a tradeoff. With our hard working Continuous Noise filter, we can choose more bins and get better resolution at the expense of doing LOTS more calculations. OR, we can choose lower resolution and do fewer real time calculations.

In summary, we've looked at how the Sample Noise button in the Continuous Noise filter performs FFT calculations to convert a time domain noise signal into a windowed frequency domain signal. Next time, we'll look at how these bins or buckets are actually used.