Notes on Ogg Vorbis and the MDCT

Keith Wright *

February 20, 2007

Abstract

Assuming only basic trigonometry, we define the Modifed Discrete Cosine Transform (MDCT), and prove its basic properties. The dread words "it can be shown" do not occur.

This document is *not* complete, but I am releasing the draft because there is enough here that it may be useful to someone. Comments and suggestions for improvement are welcome, especially if you know the answers to any of the questions at the end.

Preface

Both the Vorbis audio format[20], the better known MP3 format, as well as other audio and video compressed file formats use something called the Modified Discrete Cosine Transform (MDCT).

I was interested in how this worked, but did not know where to start. Is this the name of a particular algorithm, or does it just mean something like a cosine transform, but not quite right? So I asked the Vorbis mailing list. Nobody answered.

It was not a high priority, so I did other things for about a year. I got interested again, so I did a Google search for MDCT. I got just a few hits, one of which was my own mailing list question from a year previous, and none of which answered my questions. So I searched the local library and the internet, came up with a few definitions, removed the errors and differing sign conventions and worked out the details. That resulted in a document, Wright [19], that was released to the Web in May 2003.

The main changes in this second version are:

- Addition of a section on Change of Window size.
- Substitution of $2K$ for K in the definition of MDCT. This simplifies several formulae.
- This preface.
- Since I haven't thought about this since 2005, I changed the Licence from "Draft, please don't copy" to Creative Commons. After 15 years it's not a draft.

This doesn't count as research; it is, of course "well known" in the mathematician's sense. Nevertheless, my experience leads me to believe that it is not perfectly trivial to actually become one of those who know.

1 Trigonometry

Prologue
The Exponential Function
This is the most important function in mathematics.
—Walter Rudin [13]

The exponential function is defined by an infinite series, $\exp(z) = \sum_{n=0}^{\infty} z^n/n!$. The name comes from the fact that using this function the usual definition of integer exponentiation by repeated multiplication can be extended to all complex numbers, and when this is done it is found that complex exponentiation still follows the usual algebraic rules and that by defining the number $e = \exp(1)$ we have $e^z = \exp(z)$.

We use the mathematician's notation so that $i = \sqrt{-1}$ is the imaginary unit . We are interested in the values of the exponential function for pure imaginary numbers. It turns out that for pure imaginary arguments, the values of the exponential function lie on the unit circle, that is, $|\exp(ix)| = 1$ for all real x.

The functions $\cos$ and $\sin$ are defined as

$$\begin{aligned}\cos(z) &= \frac{\exp(iz) + \exp(-iz)}{2} \\ \sin(z) &= \frac{\exp(iz) - \exp(-iz))}{2i}\end{aligned}$$

For a real argument, x, this amounts to the real and imaginary parts, respectively, of the imaginary exponential

$$\exp(ix) = \cos(x) + i\sin(x)$$

It often happens that the easiest way to derive (or remember) a complicated-looking formula involving real trigonometric functions is to write down a simple property of complex exponentials and then take the real or imaginary part. Two complicated equations can be traded for one simple one!

For example, every child knows that

$$\cos(x+y) = \cos x \cos y - \sin x \sin y$$

This is often taught as a long geometrical proof leading to a formula that must simply be memorized, but it can be quickly derived from the simple identity $e^{i(x+y)} = e^{ix}e^{iy}$ by applying the definition of complex multiplication, *to wit*: $(x_1 + ix_2)(y_1 + iy_2) = (x_1y_1 - x_2y_2) + i(x_1y_2 + x_2y_1)$, and taking the real part. Taking the imaginary part leads to the corresponding formula for the $\sin$ function.

Substituting $-y$ for y,

$$\cos(x-y) = \cos x \cos y + \sin x \sin y$$

adding the two previous equations

$$\cos(x-y) + \cos(x+y) = 2\cos x \cos y \tag{1}$$

For a second example, we can prove the equation called Lagrange's trigonometric identity.

Claim 1 (Lagrange)

$$\sum_{n=0}^{N-1} \cos(nx) = \begin{cases} \frac{1}{2} + \frac{\sin((N-\frac{1}{2})x)}{2\sin(x/2)} & \textit{if } x \bmod 2\pi \neq 0, \\ N & \textit{if } x \bmod 2\pi = 0 \end{cases}$$

Despite the cases, this is a continuous function of x.

Proof: If $z \neq 1$, then the sum of a geometric series is given by $\sum_{n=0}^{N-1} z^n = (1-z^N)/(1-z)$, and so

$$\sum_{n=0}^{N-1} \exp(inx) = \sum_{n=0}^{N-1} (e^{ix})^n = \frac{1-(e^{ix})^N}{1-e^{ix}}$$

assuming that $e^{ix} \neq 1$, that is, that $x \bmod 2\pi \neq 0$. Multiplying above and below by $\exp(-\frac{1}{2}ix)$

$$\begin{aligned} &= \frac{\exp(-\frac{1}{2}ix) - \exp((N-\frac{1}{2})ix)}{\exp(-\frac{1}{2}ix) - \exp(\frac{1}{2}ix)} \\ &= \frac{\exp(-\frac{1}{2}ix) - \exp((N-\frac{1}{2})ix)}{-2i\sin(x/2)} \\ &= \frac{i(\exp(-\frac{1}{2}ix) - \exp((N-\frac{1}{2})ix))}{2\sin(x/2)} \end{aligned}$$

Now, taking the real part of this equation

$$\begin{aligned}\sum_{n=0}^{N-1} \cos(nx) &= \frac{-\sin(-x/2) + \sin((N-\frac{1}{2})x)}{2\sin(x/2)} \\ &= \frac{1}{2} + \frac{\sin((N-\frac{1}{2})x)}{2\sin(x/2)}\end{aligned}$$

The formula for the sum of a geometric series is no good if $z = 1$, so the above computation does not get off the ground when $x \bmod 2\pi = 0$. Fortunately, in this case $\cos(x) = 1$ so the sum is easily seen to be N. Since the cosine is a continuous function, the sum of (finitely many) cosines is continuous, so if the sum is correctly done it must be continuous. If you are skeptical, use L'Hôpital's rule to show that

$$\lim_{x \to 0} \frac{\sin((N-\frac{1}{2})x)}{2\sin(x/2)} = N - \frac{1}{2}$$

$\dashv$

Let's go through that again, with an offset!

Claim 2 (offset Lagrange)

$$\sum_{n=0}^{N-1} \cos((n+\frac{1}{2})x) = \begin{cases} \dfrac{\sin(Nx)}{2\sin(x/2)} & \textit{if } x \bmod 2\pi \neq 0, \\ N & \textit{if } x/2\pi \textit{ is even} \\ -N & \textit{if } x/2\pi \textit{ is odd} \end{cases}$$

Proof:

$$\begin{aligned} \sum_{n=0}^{N-1} \exp(i(n+\frac{1}{2})x) &= \exp(\frac{1}{2}ix)\frac{(1-\exp(iNx))}{(1-\exp(ix))} \\ &= \frac{(1-\exp(iNx))}{(\exp(-\frac{1}{2}ix)-\exp(\frac{1}{2}ix))} \\ &= \frac{i(1-\exp(iNx))}{2\sin(x/2)} \end{aligned}$$

Taking the real part yields the first case of the result. If $x/2\pi$ is even then the argument of the cosine is a multiple of 2π and so the sum has N terms, all equal to one. If $x/2\pi$ is odd then the argument of the cosine is π plus a multiple of 2π and so the sum has N terms, all equal to minus one. $\dashv$

2 Notation

Since the cases in the preceding claim simply provide the unique extension of a formula that has isolated undefined points to a continuous function defined on real numbers, we will sometimes use a formula like $\sin(Nx)/(2\sin(x/2))$ to denote the continuous function given in Claim 2 by filling in the isolated undefined points with N and $-N$.

We follow Graham, Knuth, and Patashnik[9, page 25], who follow Iverson[8, page 11] in using square brackets around a formula (e.g. an equation) as an arithmetic expression which has the value one if the formula is true, and zero if the formula is false. Since we also use square brackets to enclose array indices, we will use double square brackets for this conversion of boolean to integer values.

$$[\![x=y]\!] = \begin{cases} 1 & \text{if } x=y, \\ 0 & \text{if } x \neq y \end{cases} \tag{2}$$

3 Modified Discrete Cosine Transforms

Vorbis[20] uses the Modified Discrete Cosine Transform (MDCT), also called a Modulated Lapped Transform (MLT) or Time Domain Alias Cancelation (TDAC).

The MDCT maps an array of $2K$ real numbers into an array of K real numbers. To save a little space, let $d = 1+K$. Of course, d is an integer.

Let x be an array of $2K$ real numbers, indexed from zero. We denote the MDCT of x by $(\overrightarrow{\mathcal{F}}_K\, x)$. It is, of course, also an array indexed from zero, and is defined for all m such that $0 \leq m < K$ by the formula

$$(\overrightarrow{\mathcal{F}}_K\, x)[m] = \sum_{k=0}^{2K-1} x[k]\cos\left(\frac{\pi}{K}(k+\frac{d}{2})(m+\frac{1}{2})\right)$$

If X is an array of K elements, then the reverse MDCT is an array of $2K$ elements defined for all j such that $0 \leq j < 2K$ by

$$(\overleftarrow{\mathcal{F}}_K\, X)[j] = \frac{2}{K}\sum_{m=0}^{K-1} X[m]\cos\left(\frac{\pi}{K}(j+\frac{d}{2})(m+\frac{1}{2})\right)$$

We have called this a *reverse*, rather than an *inverse*, MDCT because the reverse transform is not a full inverse of the forward transform. It takes us part of the way back, but not all the way

We want to compute the result of applying the forward MDCT followed by the reverse MDCT. The following claim is the heart of the matter. (Recall the notation of equation 2.)

Claim 3 *If* $0 \leq j,k < 2K$ *then*

$$\begin{aligned} &\sum_{m=0}^{K-1} \cos\left(\frac{\pi}{K}(k+\frac{d}{2})(m+\frac{1}{2})\right) \\ &\qquad\qquad \cos\left(\frac{\pi}{K}(j+\frac{d}{2})(m+\frac{1}{2})\right) \\ &= \frac{K}{2}\big([\![k=j]\!] - [\![k=2K-d-j]\!] \\ &\qquad\qquad + [\![k=4K-d-j]\!]\big) \end{aligned}$$

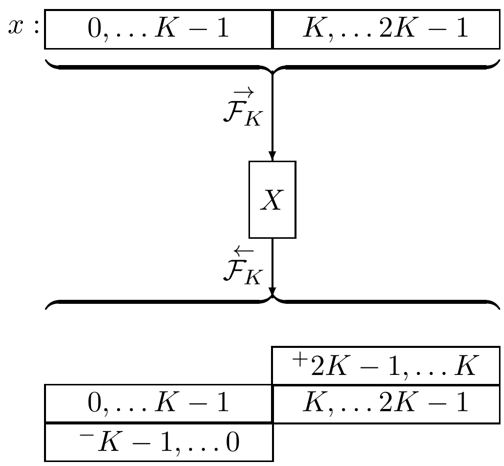

Figure 1: Forward followed by reverse MDCT

Proof: First note that by equation 1

$$\cos\left(\frac{\pi}{K}(k+\frac{d}{2})(m+\frac{1}{2})\right)\cos\left(\frac{\pi}{K}(j+\frac{d}{2})(m+\frac{1}{2})\right)$$
$$=\frac{1}{2}\cos\left(\frac{\pi}{K}(k-j)(m+\frac{1}{2})\right)$$
$$+\frac{1}{2}\cos\left(\frac{\pi}{K}(k+j+d)(m+\frac{1}{2})\right)$$

This reduces the sum of products in the claim to a sum of sums, which, by the associative law, can be added separately.

Furthermore by Claim 2 the sum of the first set of terms is

$$\sum_{m=0}^{K-1}\cos\left(\frac{\pi}{K}(k-j)(m+\frac{1}{2})\right)=\frac{\sin(\pi(k-j))}{2\sin(\pi(k-j)/(2K))}$$

but, because j and k are integers, the numerator is zero and so the fraction is zero unless $\frac{\pi}{K}(k-j) \bmod 2\pi = 0$ that is, unless $(k-j)$ is a multiple of $2K$. Because of the bounds on j and k, this can happen only when $j-k=0$, in which case the sum is K. This accounts for the first term in the right side of the claim.

Similarly, the sum of the second set of terms is

$$\sum_{m=0}^{K-1}\cos\left(\frac{\pi}{K}(k+j+d)(m+\frac{1}{2})\right)$$
$$=\frac{\sin(\pi(k+j+d))}{2\sin(\pi(k+j+d)/(2K))}$$

which is zero unless $(k+j+d)$ is a multiple of $2K$. Because of the bounds on j and k, this can happen when $k+j+d=2K$ or when $k+j+d=4K$. In the first case the sum is $-K$, in the second it is K. This accounts for the last two terms.

Of course, at most one of the three terms is non-zero for any given j and k. $\dashv$

With this formula in mind, we can compute the result of applying the forward and reverse MDCT in succession (see Figure 1).

Claim 4 *If x is an array of $2K$ numbers and j is such that $0 \le j < 2K$*

$$(\overleftarrow{\mathcal{F}_K}\,\overrightarrow{\mathcal{F}_K}\,x)[j]=\begin{cases}x[j]-x[K-j-1], & j<K\\ x[j]+x[3K-j-1], & K\le j\end{cases}$$

Proof: Just to shorten the equations a bit, let

$$s(k,j,m)=\cos\left(\frac{\pi}{K}(k+\frac{d}{2})(m+\frac{1}{2})\right)$$
$$\cos\left(\frac{\pi}{K}(j+\frac{d}{2})(m+\frac{1}{2})\right)$$

So that

$$(\overleftarrow{\mathcal{F}_K}\,\overrightarrow{\mathcal{F}_K}\,x)[j]=\frac{2}{K}\sum_{m=0}^{K-1}\sum_{k=0}^{2K-1}x[k]s(k,j,m)$$
$$=\frac{2}{K}\sum_{k=0}^{2K-1}x[k]\sum_{m=0}^{K-1}s(k,j,m)$$
$$=\sum_{k=0}^{2K-1}x[k]\big([\![k=j]\!]-[\![k=2K-d-j]\!]$$
$$+[\![k=4K-d-j]\!]\big)$$

$$=x[j]-x[2K-d-j]+x[4K-d-j]$$
$$=x[j]-x[2K-K-1-j]$$
$$+x[4K-K-1-j]$$
$$=x[j]-x[K-j-1]+x[3K-j-1]$$

Now note that if $j<K$ then $3K-j-1>2K-1$ and so $x[3K-j-1]=0$ but if $K\le j$ then $K-j-1<0$ and so $x[K-j-1]=0$. $\dashv$

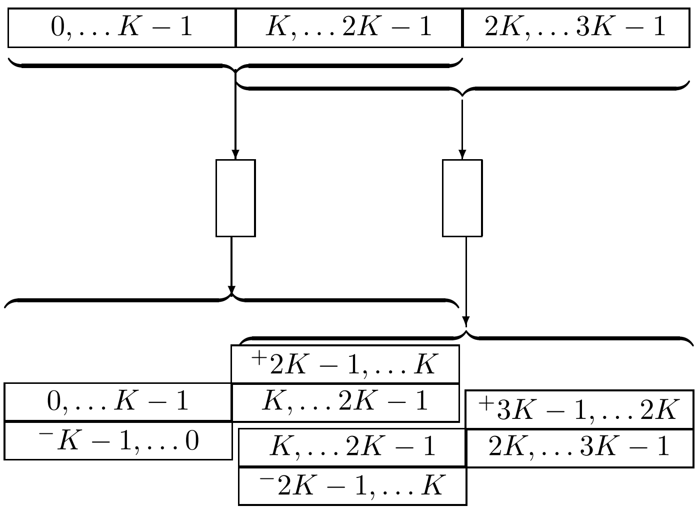

Figure 2: Reconstruction by Overlap

Thus we see that the Modified Discrete Cosine Transform maps a sequence of $2K$ real numbers into a sequence of K real numbers in such a way that the original sequence can be recovered, only slightly scrambled. (Of course it is not surprising that the sequence gets a bit scrambled, since the transform has only half as much information.) The scrambling is called Time Domain Aliasing, because the sequences are often samples of a waveform which is a function of time, and the scrambling consists of adding in parts of the waveform that should occur at some other time.

The trick to unscramble the sequence is called Time Domain Alias Cancelation. It is illustrated in Figure 2. To reconstruct any of the original sequence, we need two of the half-size blocks of transformed data. By transforming two overlapping blocks, first in the forward direction, then in the reverse, we find that by combining the results we can perfectly reconstruct the original signal, while totally canceling the aliased part. Since this makes the transformed data the same length as the original data, it is not absurd on its face to try to get the original data back, but there it is still not obvious that it can be done. In fact, although Discrete Fourier Transforms have been known since stone age times[1], it was not until 1986 that Princen and Bradley[12] showed that alias cancelation would work.

To form the transform of a long sequence, instead of breaking it into disjoint blocks, we break it into blocks of length $2K$ that start at multiples of K, so that the blocks overlap by half their length. We then take the MDCT of each block. Since the MDCT of a block of length $2K$ has length only K, the result of transforming a long sequence is approximately the same length as the original sequence. (There is half a block of overhead at each end.)

Claim 5 *Suppose x is a long sequence of samples, and let $x_0[j] = x[j]$ and $x_1[j] = x[j+K]$ for $0 \le j < 2K$.*

For any j such that $0 \le j < K$,

$$(\overleftarrow{\mathcal{F}_K}\,\overrightarrow{\mathcal{F}_K}\,x_0)[j+K] + (\overleftarrow{\mathcal{F}_K}\,\overrightarrow{\mathcal{F}_K}\,x_1)[j] = 2x[j+K]$$

Proof: For j such that $0 \le j < K$ we have, by Claim 4,

$$\begin{aligned}
&(\overleftarrow{\mathcal{F}_K}\,\overrightarrow{\mathcal{F}_K}\,x_0)[j+K] + (\overleftarrow{\mathcal{F}_K}\,\overrightarrow{\mathcal{F}_K}\,x_1)[j] \\
&\quad = x_0[j+K] + x_0[3K-(j+K)-1] \\
&\qquad + x_1[j] - x_1[K-j-1] \\
&\quad = x[j+K] + x[2K-j-1] \\
&\qquad + x[j+K] - x[K-j-1+K] \\
&\quad = 2x[j+K]
\end{aligned}$$

⊣

Thus, by overlapping and adding two adjacent blocks, we cancel out the aliased parts and recover the original data in the overlapped part.

4 Windowing

At this point we have shown how to transform a signal into a frequency domain representation with (approximately) the same number of bits, and then reverse the transformation to get the original back. Well, that is nice, but but we could have done that by taking ordinary Fourier or Cosine transforms of non-overlapping blocks. So what has been gained?

The whole point is to compress the signal by using some coding tricks to represent the transformed signal in fewer bits. This causes a loss of information, and therefore, after coding and decoding, the transformed signal is changed slightly and

[1] Well, almost.

we can no longer expect perfect reconstruction. We hope to do it in such a way that the changes to the signal due to coding and decoding, are not audible. The problem is that when non-overlapping blocks are processed independantly, any changes will make the signal "discontinuous" [2] at the block boundaries. This discontinuity will be heard as a buzz or hum with pitch proportional to the block rate.

To eliminate this so-called "blocking artifact", we multiply each block by a "window" so that the blocks end smoothly, rather than just cut off suddenly. Because the blocks overlap, we can arrange that the signal fades out smoothly at the end of one block and fades in at the beginning of the overlapping block in such a way that the overall gain remains constant during the block transition. To do this while keeping the perfect reconstruction property, we must apply a window to both the input and the output of the transform procedure. This gives us enough degrees of freedom to adjust so that alias cancelation can be achieved.

Let h_0 and h_1 be the window functions for the input of the first and second blocks, respectively, and let f_0 and f_1 be the window functions for the output of the two blocks. Then the input blocks to the transform procedure are $x_0[j] = h_0[j]x[j]$ and $x_1[j] = h_1[j]x[j+K]$ for $j = 0 \ldots 2K$. The following claim gives the conditions on the windows under which perfect time domain alias cancelation will occur when windowed segments are overlapped and added.

Refer to figure 3, where the labels along the bottom show indices into x_0 and the labels along the top show indices into x_1.

Claim 6 *If for j such that* $0 \le j < K$

$$f_0[j+K]h_0[j+K] + f_1[j]h_1[j] = 1$$

and

$$f_0[j+K]h_0[2K-j-1] - f_1[j]h_1[K-j-1] = 0$$

then for j such that $0 \le j < K$

$$f_0[j+K](\overleftarrow{\mathcal{F}}_K \overrightarrow{\mathcal{F}}_K x_0)[j+K] + f_1[j](\overleftarrow{\mathcal{F}}_K \overrightarrow{\mathcal{F}}_K x_1)[j]$$
$$= 2x[j+K]$$

[2] The word 'discontinuous' is in quotes, because the concept does not really apply to a discrete signal.

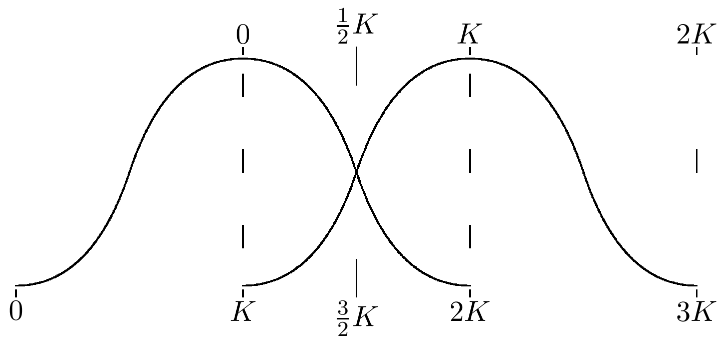

Figure 3: Two Windows of the Same Size

Proof: We have, by Claim 4,

$$
\begin{aligned}
&f_0[j+K](\overleftarrow{\mathcal{F}}_K \overrightarrow{\mathcal{F}}_K x_0)[j+K] + f_1[j](\overleftarrow{\mathcal{F}}_K \overrightarrow{\mathcal{F}}_K x_1)[j] \\
&\quad = f_0[j+K]x_0[j+K] \\
&\qquad + f_0[j+K]x_0[3K-(j+K)-1] \\
&\qquad + f_1[j]x_1[j] - f_1[j]x_1[K-j-1] \\
&\quad = f_0[j+K]h_0[j+K]x[j+K] \\
&\qquad + f_0[j+K]h_0[2K-j-1]x[2K-j-1] \\
&\qquad + f_1[j]h_1[j]x[j+K] \\
&\qquad - f_1[j]h_1[K-j-1]x[2K-j-1]
\end{aligned}
$$

In order to recover the original function we need the coefficient of $x[j+K]$ to be one and the coefficient of $x[2K-j-1]$ to be zero. In other words,

$$f_0[j+K]h_0[j+K] + f_1[j]h_1[j] = 1$$

and

$$f_0[j+K]h_0[2K-j-1] - f_1[j]h_1[K-j-1] = 0$$

$\dashv$

When we use the same window for all blocks, we can drop the subscripts. (This does not hold if blocks of different sizes are mixed.)

Vorbis uses the window

$$w_K(x) = \sin\left(\frac{\pi}{2}\sin^2\left(\frac{\pi}{2K}x\right)\right)$$

offset by half a step, for both input and output.

This means that

$$f[j] = h[j] = w_K(j + \frac{1}{2}) = \sin\left(\frac{\pi}{2}\sin^2\left(\frac{\pi}{2K}(j+\frac{1}{2})\right)\right)$$

Note that $w_K(0) = w_K(2K) = 0$ and $w_K(K) = 1$, and that the window (which extends from 0 to $2K$) is symmetric about K. The half-step offset makes the integerized window symmetric, in the sense that $f[0] = f[2K-1]$, $f[K-1] = f[K]$, and in general $f[j] = f[2K-1-j]$.

The following claim shows that the Vorbis window satisfies the conditions for perfect reconstruction.

Claim 7 *If* $f[j] = h[j] = \sin\left(\frac{\pi}{2}\sin^2\left(\frac{\pi}{2K}(j+\frac{1}{2})\right)\right)$ *then*

$$f[j+K]h[j+K] + f[j]h[j] = 1$$
$$f[j+K]h[2K-j-1] - f[j]h[K-j-1] = 0$$

Proof: Since, $\sin(\frac{\pi}{2}+x) = \sin(\frac{\pi}{2}-x) = \cos(x)$ and $\cos^2(x)+\sin^2(x) = 1$, for any real x, we have

$$\sin^2\left(\pi\frac{x+K}{2K}\right) = \sin^2\left(\frac{\pi x}{2K}+\frac{\pi}{2}\right) = \cos^2\left(\frac{\pi x}{2K}\right) = 1 - \sin^2\left(\frac{\pi x}{2K}\right)$$

Taking $x = j + 1/2$, we have

$$f[j+K] = h[j+K] = \sin\left(\frac{\pi}{2}\sin^2\left(\pi\frac{(j+K)+\frac{1}{2}}{2K}\right)\right) = \sin\left(\frac{\pi}{2} - \frac{\pi}{2}\sin^2\left(\pi\frac{j+\frac{1}{2}}{2K}\right)\right) = \cos\left(\frac{\pi}{2}\sin^2\left(\frac{\pi(j+\frac{1}{2})}{2K}\right)\right)$$

Let $v(j) = \frac{\pi}{2}\sin^2\left(\frac{\pi(j+1/2)}{2K}\right)$. Then

$$f[j]{=}h[j] = \sin\left(\frac{\pi}{2}\sin^2\left(\frac{\pi(j+\frac{1}{2})}{2K}\right)\right) = \sin(v(j))$$

and

$$f[j+K]{=}h[j+K] = \cos(v(j))$$

so the first condition becomes

$$f[j+K]h[j+K] + f[j]h[j] = \cos^2(v(j)) + \sin^2(v(j)) = 1$$

At the same time, letting $x = (j+1/2)$, we have

$$h[2K-j-1] = \sin\left(\frac{\pi}{2}\sin^2\left(\pi\frac{(2K-j-1)+\frac{1}{2}}{2K}\right)\right) = \sin\left(\frac{\pi}{2}\sin^2\left(\pi\frac{2K-(j+\frac{1}{2})}{2K}\right)\right) = \sin\left(\frac{\pi}{2}\sin^2\left(\pi - \pi\frac{j+\frac{1}{2}}{2K}\right)\right) = \sin\left(\frac{\pi}{2}\sin^2\left(\pi\frac{j+\frac{1}{2}}{2K}\right)\right) = \sin(v(j))$$

while taking $x = -(j+1/2)$, we have

$$h[K-j-1] = \sin\left(\frac{\pi}{2}\sin^2\left(\pi\frac{K-(j+\frac{1}{2})}{2K}\right)\right) = \sin\left(\frac{\pi}{2} - \frac{\pi}{2}\sin^2\left(\pi\frac{-(j+\frac{1}{2})}{2K}\right)\right) = \cos(v(j))$$

and so the second condition is

$$f[j+K]h[2K-j-1] - f[j]h[K-j-1] = \cos(v(j))\sin(v(j)) - \sin(v(j))\cos(v(j)) = 0$$

$\dashv$

5 Change of window size

Vorbis can change the size of the window, using a long window when encoding relatively "smooth" parts of the waveform and a shorter window when encoding rapidly changing

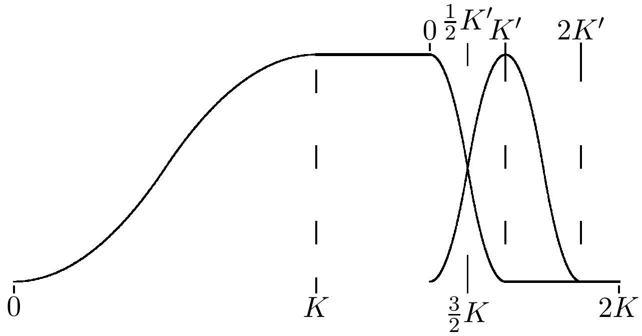

Figure 4: Two Windows of Different Sizes

parts. The basic reference for this seems to be a paper in German by Edler[2]. I have been unable to obtain a copy of this paper; perhaps someone can help me with this (I can read German, with time and motivation). The reference to Edlers paper is taken from Sporer et. al.[14] The following is reconstructed from Vorbis documentation and source code.

In the previous section we have shown how to construct a window of length $2K$, where K is a parameter.

Assume that $K \geq K'$.

$$w_{KK'}(x) \begin{cases} w_K(x), & x \leq K \\ 1, & K \leq x \leq K+L \\ w_{K'}(x'), & K+L \leq x \leq K+L+K' \\ 0, & K+L+K' \leq x < 2K \end{cases}$$

where $x' = x - \frac{3}{2}K + \frac{3}{2}K' = x - (K+L) + K'$.

$f_0[j] = h_0[j] = w_{KK'}(j + \frac{1}{2})$

Note that if $K = K'$ then this formula reduces to the previous one, $f_0[j] = h_0[j] = w_K(j)$ for $0 \leq j < 2K$. Also note that the case which is fixed at 1 and the case which is fixed at 0 are the same length, $\frac{1}{2}K - \frac{1}{2}K'$. Let's define $L = \frac{1}{2}K - \frac{1}{2}K'$ so that L is this common length.

Let the input blocks to the transform procedure be $x_0[j] = h_0[j]x[j]$ for $j = 0 \ldots 2K$, and $x_1[j] = h_1[j]x[j+K+L]$ for $j = 0 \ldots 2K'$,

Refer to figure 4, where the labels along the bottom show indices into x_0 and the labels along the top show indices into x_1.

Claim 8 *If for j such that* $0 \leq j < L + K'$

$$f_0[j+K]h_0[j+K] + f_1[j]h_1[j] = 1$$

and

$$f_0[j+K]h_0[2K-j-1] - f_1[j]h_1[K-j-1] = 0$$

then for j such that $0 \leq j < L + K'$

$$f_0[j+K](\overleftarrow{\mathcal{F}_K}\,\overrightarrow{\mathcal{F}_K}\,x_0)[j+K] + f_1[j](\overleftarrow{\mathcal{F}_{K'}}\,\overrightarrow{\mathcal{F}_{K'}}\,x_1)[j]$$
$$= 2x[j+K]$$

6 Questions

The above is a fairly complete and detailed account of the basics of the MDCT, but there are a many related issues that are just not mentioned at all.

Algorithms The original objective was to understand the algorithm used in `mdct.c`. Knowing what it computes is a step forward, but does not suffice to expain the code.

Context and Overview How does the MDCT relate to the ordinary Cosine Transform and to the Fourier transform? The TEX source of this document has some fragmentary notes on this which have not yet been pieced together.

References

[**] This is a list of publications that seemed like they might be relevant, I have not even seen copies of all of them.

[1] Ronald E. Crochiere & Lawrence R. Rabiner, *Multirate Digital Signal Processing*, Prentice-Hall signal processing series, Englewood Cliffs, New Jersey (1983) TK5102.5 C76, ISBN 0-13-605162-6

[2] B. Edler, *Coding of Audio Signals with Overlapping Block Transform and Adaptive Window Functions* (in German), Frequenz, vol. 43(1989), pp. 252–256

[3] M. Temerinac, B. Edler *A Unified Approach to Lapped Orthogonal Transforms* IEEE Trans. on Image Processing, Vol. 1, No. 1, pp. 111-116, Jan. 1992

[4] M. Temerinac, B. Edler *Overlapping Block Transform: Window Design, Fast Algorithm and an Image Coding Experiment* IEEE Trans. on Communications, Vol. 43, No. 9, Sep. 1995

[5] M. Temerinac, B. Edler *LINC: A Common Theory of Transform and Subband Coding* IEEE Trans. on Communications, Vol. 41, No. 2, pp. 266-274, Feb. 1993

[6] B. Edler *Aliasing Reduction in Subbands of Cascaded Filter Banks with Decimation* Electronics Letters, Vol. 28, No. 12, pp. 1104-1105, June 1992.

[7] Douglas F. Elliott, *Handbook of Digital Signal Processing Engineering Applications* Academic Press (1987) TK5102.5 H32, ISBN 0-12-237075-9

[8] Kenneth E. Iverson, *A Programming Language*, Wiley (1962)

[9] Ronald L. Graham, Donald E. Knuth & Oren Patashnik *Concrete Mathematics* Addison-Wesley (1994)

[10] N. S. Jayant & P. Null *Digital Coding of Waveforms* Prentice-Hall (1984)

[11] Henrique S. Malvar, *Signal Processing with Lapped Transforms*, Artech House, Norwood MA (1992)

[12] J. Princen & A. Bradley. *Analysis/Synthesis Filter Bank Design Based on Time Domain Aliasing Cancellation*, IEEE Transaction, ASSP-34, No. 5, Oct. 1986, pp. 1153-1161

[13] Walter Rudin, *Real & Complex Analysis, third edition*, McGraw-Hill (1987)

[14] Th. Sporer & Kh. Brandenburg & B. Edler, *The use of multirate filter banks for coding of high quality digital audio*, 6th European Signal Processing Conference (EUSIPCO), Amsterdam, June 1992, Vol.1 pp. 211-214.

[15] Pat Yip & K. Ramamohan Rao, *Fast DIT Algorithms for a Family of Discrete Cosine and Sine Transforms*, Circuits, 3, 387-408(1984)

[16] K. R. Rao & P. Yip, *Discrete Cosine Transform: Algorithms, Advantages and Applications*, Academic Press, ISBN: 012580203X (1990)

[17] Pat Yip & K. Ramamohan Rao, *Fast Discrete Transforms* Chapter 6 in [7] pp 481-525

[18] P. Yip & K. R. Rao, *The decimation-in-frequency algorithms for a family of discrete sine and cosine transforms* Circuits, Systems, and Signal Processing, pages 4–19, 1988.

[19] Keith Wright, *Post to vorbis@xiph.org, archived at* `http://lists.xiph.org/pipermail/vorbis/2003-May/010560.html`

[20] Xiphophorus,
`http://www.vorbis.com`
`http://www.xiph.org`