Lecture 35 COMP 423 April 7, 2006

Last class, I introduced you to the idea of subband coding for a sequence $\vec{X}$. The idea was to decompose $\vec{X}$ into a set of sequences, such that each of these sequences was composed of a band of frequencies. I gave an example of using two bands which were defined by a local difference $\vec{d}$ and a local sum $\vec{s}$. I argued that the $\vec{s}$ and the $\vec{d}$ sequences consist mostly[1] of frequencies with large and small values of k, respectively. I also showed we could subsample these two sequences and reconstruct the original sequence $\vec{X}$ from the subsamples only.

To show how to decompose $\vec{X}$ into many (32) subbands and show how to reconstruct by subsampling and taking appropriate combinations to obtain the original sequence again would require several more lectures and would take us far away from the topic of Data Compression and into rather technical issues in Signal Processing and, in particular, Subband Coding. Instead, today I will give an example of subband coding that generalizes the idea from last class, and in which the subsampling and reconstruction are particularly simple. This will at least give you a flavor of how subband coding works in a more complicated situation, without going into the gory details of subband coding in general. It will also allow me to sketch out a coding method that is similar to MP3.[2]

The two band method from last class can be interpreted as transform coding of $\vec{X}$ using block size of $m = 2$. We normalize the local sums and differences as

$$s_i = \frac{1}{\sqrt{2}}(X_i + X_{i-1})$$

$$d_i = \frac{1}{\sqrt{2}}(X_i - X_{i-1})$$

Then s_{2j} and d_{2j} are the $Y_j(0)$ and $Y_j(1)$ DCT coefficients of block j, where the blocks are of size $m = 2$.

[NOTE (April 7): I changed the DCT definition in lecture 30, so that the columns of $\mathbf{C}$ are indeed of unit length. Please make sure that you print out the most recent version of the notes.]

Let's consider two spectrograms, defined by blocks of size $m = 32$ and blocks of size $m = 512$. I will refer to the blocks of size $m = 32$ as *subblocks*. These will generalize the blocks of size $m = 2$ discussed last lecture. The blocks of size $m = 512$ are the same as discussed earlier. These blocks will be used to estimate the masking effects. I will refer to these as *superblocks*.

The spectrogram defined by the subblocks of size $m = 32$ has $k = 0, 1, \ldots, 31$ and is sampled every $\frac{32}{44{,}000}$ seconds, which is about $\frac{3}{4}$ ms. That is, we have 32 numbers (the $Y_j(k)$ coefficients) every $\frac{3}{4}$ ms.

The k values represent temporal frequencies $k/2$ cycles per $\frac{32}{44{,}000}$ seconds, that is, $k = 1$ is about 700 cycles per second, $k = 2$ is about 1400 cycles per second, etc. These k values only crudely represent the various sound frequencies. We also say that these k values provide very poor *resolution* of the frequencies present in the sound.

We need good resolution of different frequencies in order to estimate masking effects in the sound, since masking occurs when multiple nearby sound frequencies are present. In order to know that *multiple* nearby frequencies are present, we need to be able to distinguish different nearby

[1] There is no hard cutoff where we say that the spectrograms of the $\vec{s}$ have non-zero Y values in the lower band of the k's and the spectrogram of the $\vec{d}$ sequence have non-zero Y values only in the upper band.

[2] MP3 is the "third layer" of the audio portion of the MPEG standard. It was developed in the early 1990s.

frequencies, and this means we need a large range of k values. Since the number of frequencies represented in the spectrogram is m, which is the block size, it follows that to have good frequency resolution (for analysing and predicting the masking properties sound), we therefore need large block sizes.

Encoder

The encoder computes two spectrograms, the subblock and superblock which are of sizes $m = 32$ and $m = 512$, respectively. The superblock spectrogram is used to estimate the amount of masking that would occur in the original signal $\vec{X}$, in particular, it is used to compute Δ values (see below). The subblock spectrogram defines the data that is encoded. For subblock j within the superblock, and for each $k = 1, \ldots, 32$, there are 32 $Y_j(k)$ coefficients.

The quantization widths Δ are defined for each *super*block and for each 32 values of k in each *sub*block of that superblock. Thus, there are 32 quantizers $\Delta(k)$ for each superblock.

For a given superblock, let $Y_j(k)$ denote the j^{th} sample of subband k in this superblock, where $k = 1, ..., 32$ and $j = 1, ..., 16$.

You might think that the quantization levels are chosen in the usual way, namely using a mid-tread quantizer, and

$$l_j(k) = round(\frac{Y_j(k)}{\Delta(k)}$$

MP3 doesn't do this, however. Instead a midrise quantizer is used.

The midrise quantizer has no overload errors. This is achieved by choosing the number of quantization levels to be large enough, and encoding this number. For each band k and each superblock, let $N(k)$ be the smallest integer such that, for all subblocks $j \in 1, \ldots, 16$,

$$Y_j(k) \in [-N(k)\Delta(k),\ N(k)\Delta(k)]$$

That is, $2N(k)$ levels can capture all 16 $Y_j(k)$ values for each k.

Recall that the $\Delta(k)$ values are chosen, based on the superblock spectrogram which has high frequency resolution. Since the superblock has a time (sampling) resolution of 512 samples, only one $\Delta(k)$ values is chosen for each superblock and each $k = 1, \ldots, 15$.

Thus, for each superblock, the encoder sends the 32 $N(k)$ values, the 32 $\Delta(k)$ values, and the $512 = 32 \times 16$ levels,

$$l_j(k) = \lceil(\frac{Y_j(k)}{\Delta(k)}\rceil$$

for each of the 32 subbands and each of the 16 subblocks.

Decoder

The decoder (the MP3 player) is much simpler than the encoder. The decoder does not have to do any masking calculations. For each superblock, the decoder is given the $N(k)$ values and the $\Delta(k)$ values, From these, it defines $N(k)$ reconstruction values, which are the midpoints of the intervals of size $\Delta(k)$.

For each $Y_j(k)$ value, the decoder is given a number from $-N(k), \ldots, -1$ or $1, 2, \ldots, N(k)$. It estimates the $Y_j(k)$ value as the midpoint of the corresponding interval. This is just standard midrise quantization with no overload error.

Finally, the decoder reconstructs an estimate of the original signal $\vec{x}$ by applying the inverse DCT ($m = 32$) to the quantized $Y_j(k)$ values of each subblock.