Lossy differential coding

Recall from lecture 22 that with *lossless* differential coding, the encoder sends the first value and then sends the difference between each value and the previous one. That is, the encoder sends $x_1, x_2 - x_1, x_3 - x_2, \ldots$. The decoder then reconstructs the original sequence by $x_{j+1} := x_j + (x_{j+1} - x_j)$. Lossy differential coding is more subtle. With *lossy differential coding*, the encoder quantizes the differences $x_{j+1} - x_j$. Quantization introduces error, and so the encoder need to be careful with what errors it introduces.

Quantization Method 1 (Bad)

The most obvious way to encode the differences (which turns out to be a bad method) is to quantize as follows:

$$Q^*(x_1), \quad Q(x_2 - x_1), \quad Q(x_3 - x_2), \quad Q(x_4 - x_3), \quad \ldots$$

[Note that a different quantizer $Q^*()$ is used for x_1 than for the differences $x_{j+1} - x_j$ since we expect very different range of values for x_1 than for the differences $x_{j+1} - x_j$.]

Unfortunately this turns out to be a bad method. Why? The decoder would estimate the original sequence of samples as follows. Let $\hat{x}_j$ be the decoder's estimate of x_j, and define the quantization errors:

$$\epsilon_i \equiv Q(x_{i+1} - x_i) - (x_{i+1} - x_i), \qquad \epsilon^* \equiv Q^*(x_1) - x_1$$

Then,

$$\hat{x}_1 = Q^*(x_1)$$

$$\begin{aligned} \hat{x}_{j+1} &= \hat{x}_1 + \sum_{i=1}^{j} Q(x_{i+1} - x_i) \\ &= x_1 + \epsilon^* + \sum_{i=1}^{j} (x_{i+1} - x_i) + \sum_{i=1}^{j} \epsilon_i, \\ &= x_{j+1} + \epsilon^* + \sum_{i=1}^{j} \epsilon_i \end{aligned}$$

You might think that the quantization errors ϵ_i's would cancel out. But this is not always the case. For example, take the example that $x_j = cj$ where c is a constant. Then, $x_{j+1} - x_j = c$ and so $\epsilon_j = Q(c) - j$ for all j. Thus, for this example,

$$\hat{x}_{j+1} = x_{j+1} + \epsilon^* + j(Q(c) - c)$$

and so the error grows without bound when $Q(c) \neq c$. Clearly this is not what we want.

Quantization Method 2 (Good)

There is an easy way to fix the problem. Instead of encoding the quantized difference between x_{j+1} and x_j, the encoder sends the quantized difference between x_{j+1} and the decoder's estimate of x_j.

That is, the encoder quantizes the difference $x_{j+1} - \hat{x}_j$. So the encoder sends $Q^*(x_1)$, $Q(x_2 - \hat{x}_1)$, $Q(x_3 - \hat{x}_2)$, $\ldots$.

The decoder's estimate of x_{j+1} is:

$$\begin{aligned}
\hat{x}_{j+1} &= \hat{x}_j + Q(x_{j+1} - \hat{x}_j) \\
&= \hat{x}_j + x_{j+1} - \hat{x}_j + \epsilon_j, \qquad \text{that is, } \epsilon_j \equiv Q(x_{j+1} - \hat{x}_j) - (x_{j+1} - \hat{x}_j) \\
&= x_{j+1} + \epsilon_j
\end{aligned}$$

Now we see that the decoder's error does not accumulate!

The general principle here is that the encoder should encode differences with respect to the decoder's estimate, not differences with respect to the true value. The principle will be used several times in the remaining lectures.

Linear predictive coding

Let's generalize the method of differential coding, by making the estimate for $\hat{x}_{j+1}$ depend on the estimates for the previous k samples, rather than only on the estimate of the previous sample. That is, rather than using $\hat{x}_j$ as a predictor for x_{j+1} and encoding the difference, the prediction for x_{j+1} is a weighted sum of $\hat{x}_j$, $\hat{x}_{j-1}$, $\ldots, \hat{x}_{j-k+1}$, namely

$$\sum_{m=0}^{k-1} a_m \; \hat{x}_{j-m}$$

where the a_m's are suitably chosen constants. Note that if $a_0 = 1$ and $a_m = 0$ for $m > 0$, then we just have differential coding as discussed last class.

To get process started, the encoder needs to transmit the first k samples, either the original values $x_1, \ldots, x_k$, or quantized original values $\hat{x}_1, \ldots, \hat{x}_k$. The encoder also needs to encode the k constants $a_0, \ldots, a_{k-1}$. I'll explain next lecture how these constants are obtained.

To encode the next value of the sequence, i.e. x_{j+1}, the encoder encodes the quantized difference between x_{j+1} and the predicted value of x_{j+1}, that is,

$$Q(x_{j+1} - \sum_{m=0}^{k-1} a_m \; \hat{x}_{j-m}) \; .$$

Then, the decoder's estimate of x_{j+1} is

$$\hat{x}_{j+1} = \sum_{m=0}^{k-1} a_m \; \hat{x}_{j-m} \; + Q(x_{j+1} - \sum_{m=0}^{k-1} a_m \; \hat{x}_{j-m}) \; , \tag{1}$$

that is, the predicted value plus the encoded difference. If ϵ_{j+1} is the quantization error in the difference between the predicted value and the true value of x_{j+1}, i.e.

$$\epsilon_{j+1} \equiv Q(x_{j+1} - \sum_{m=0}^{k-1} a_m \; \hat{x}_{j-m}) - (x_{j+1} - \sum_{m=0}^{k-1} a_m \; \hat{x}_{j-m})$$

then,

$$\hat{x}_{j+1} = x_{j+1} - \epsilon_{j+1}$$

and so there is no accumulation of error.