EE5585 Data Compression March 8, 2013

Lecture 13

Instructor: Arya Mazumdar *Scribe: Artem Mosesov*

Scalar Quantization

Basics

Being a subset of vector quantization, scalar quantization deals with quantizing a string of symbols (random variables) by addressing one symbol at a time (as opposed to the entire string of symbols.) Although, as one would expect, this is *not* ideal and will not approach any theoretical limits; scalar quantization is a rather simple technique that can be easily implemented in hardware. The simplest form of scalar quantization is uniform quantization.

Given a string, x_1,x_2,...x_n, we first pick a symbol to quantize and disregard the rest. We then quantize this continuous variable to a uniform set of points, as follows:

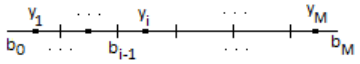

Figure 1: Uniform quantization

So we have M+1 boundaries b_i, and M quantization levels y_i (which fall in the middle of the boundary points). So a continuous number that falls between the boundaries b_{i-1} and b_i gets assigned a quantized value of y_i. Naturally, this introduces signal distortion - an error. The error measure typically used for this is mean squared error (Euclidean distance, as opposed to Hamming distance that's used for binary strings). We call this the quantization error, and recognize that it takes $\log_2(M)$ bits to store the symbol.

Optimization

We note that uniform quantization is only optimal (in the minimum MSE sense) for a uniform distribution. Given an arbitrary PDF (not necessarily uniform), we would like to find an optimal quantization.Let us consider a random variable X with a PDF $f_X(x)$.

The MSE is,

$$\int_{-\infty}^{\infty} (x - Q(x))^2 f_X(x)\, dx$$

where *Q(x)* is the quantized output of X, that is

$$Q(x) = y_i \quad if \quad b_{i-1} \leq x \leq b_i$$

Simplifying the expressions for the error, we have

$$\sigma_q^2 \equiv \text{MSE} = \sum_{i=1}^{M} \int_{b_{i-1}}^{b_i} (x - y_i)^2 f_X(x) dx$$

This, then, becomes an optimization problem - given a maximum distortion rate, we would like to find the optimal location of the quantization points (y_i's and b_i's). Of course, we can always have a very large number of quantization points to keep the distortion low; however, we would like to keep this number low, as to save memory space when storing these values.

Referring back to a uniform distribution, we note that (for a non-uniform pdf), the probability of different y_i's is not the same. That is, at the quantizer output we may see a lot more of a certain quantization point than another. This makes the points a candidate for Huffman coding, as seen earlier in the course. The probability of a particular quantization point is

$$P(Q(x) = y_i) = \int_{b_{i-1}}^{b_i} f_X(x)dx$$

Now we can begin to optimize the average length of the code for the quantization points, which is

$$\sum_{i=1}^{M} l_i \int_{b_{i-1}}^{b_i} f_X(x)dx \quad ,$$

where l_i is the length of the code for y_i. This optimization must occur subject to the following two constraints:

Constraint 1: l_i's satisfy Kraft's inequality.

Constraint 2: $\sigma_q^2 \equiv \text{MSE} = \sum_{i=1}^{M} \int_{b_{i-1}}^{b_i} (x - y_i)^2 f_X(x)dx \leq D$

To see how to simplify this problems, we look again at a uniform quantizer. Lets assume that X (the symbol we want to quantize) is a uniform $\sim U[-L, L]$ variable. The quantization 'lengths' are then $\Delta = \frac{2L}{M}$, as shown in figure 2.

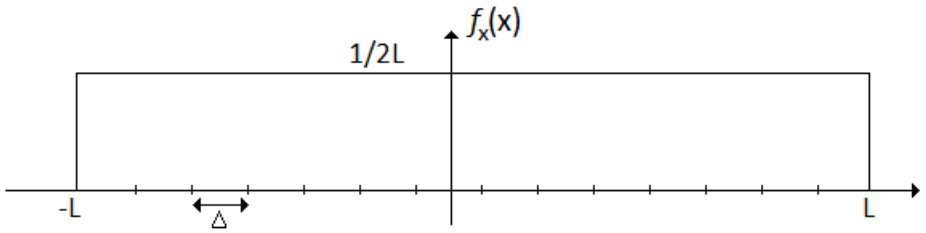

Figure 2: Uniform quantization for uniform random variable

The quantization error then becomes,

$$\sigma_q^2 = \sum_{i=1}^{M} \int_{-L+(i-1)\Delta}^{-L+i\Delta} (x - y_i)\frac{1}{2L}dx$$

The optimal y_i is then $\frac{b_{i-1}+b_i}{2}$. Of course, this is only for a uniform random variable, as initially assumed. We may also notice that the quantization error plot is merely a sawtooth wave with wavelength Δ and amplitude $\frac{\Delta}{2}$. The integral of this is then, $\sigma_q^2 = \frac{\Delta^2}{12}$.

We may think of the quantization error produced by the system as an additive noise - the 'quantization noise'. The power of this noise is then σ_q^2. The idea is shown in Figure 3, below.

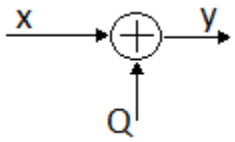

Figure 3: Uniform quantization for uniform random variable

From the figure, we note that the power of the input signal is,

$$\sigma_x^2 = \int_{-L}^{L} x^2 f_X(x) dx = \frac{L^2}{3}$$

Hence, we have, SNR $= 10 \log_{10} (\frac{\sigma_x^2}{\sigma_q^2}) = 20 \log_{10} M$, where M is, as before, the number of quantization levels. Since this is a uniform distribution, Huffman coding will not get us anywhere, and we have the maximum entropy of $20 \log_{10} M$. For an n-bit quantizer then, we get $20 \log_{10} 2^n = 20n \log_{10} 2 \approx 6n$ dB. So the SNR is directly proportional to the number of bits used for quantization - with an increase of one bit correspond to about a 6dB increase of SNR.

Now we take a look at optimum quantization for non-uniform distributions. Similarly, we have

$$\sigma_q^2 = \sum_{i=1}^{M} \int_{b_{i-1}}^{b_i} (x - y_i)^2 f_x(x) dx$$

which we would like to minimize. Often, however, we don't know the exact PDF of the symbols, nor do we know the variance. To overcome this, we use *adaptive quantization*. As we've seen before, one way to do this is to estimate the PDF by observing a string of symbols. This is known as *forward* adaptive quantization.

Going back to minimizing σ_q^2, we want

$$\frac{\delta \sigma_q^2}{\delta y_i} = \frac{\delta}{\delta y_i} \int_{b_{i-1}}^{b_i} (x - y_i)^2 f_x(x) dx =$$

$$= \frac{\delta}{\delta y_i} [\int_{b_{i-1}}^{b_i} x^2 f_x(x) dx - 2y_i \int_{b_{i-1}}^{b_i} x f_x(x) dx + y_i^2 \int_{b_{i-1}}^{b_i} f_x(x) dx] =$$

$$= -2 \int_{b_{i-1}}^{b_i} x f_x(x) dx + 2y_i \int_{b_{i-1}}^{b_i} f_x(x) dx = 0$$

And then we have,

$$y_i = \frac{\int_{b_{i-1}}^{b_i} x f_x(x) dx}{\int_{b_{i-1}}^{b_i} f_x(x) dx} \tag{1}$$

So this is the optimal location of the reconstruction points, given the quantization points. Now we have to find the quantization points. We do this similarly,

$$\frac{\delta \sigma_q^2}{\delta b_i} = 0$$

which gives us the optimal points

$$b_{i-1} = \frac{y_{i-1} + y_i}{2} \tag{2}$$

So what we can do with this is an iterative procedure, where we first initialize the variables, then go back and forth optimizing each one, and (ideally) arriving very close to an optimality point.

Lloyd-Max Algorithm

The *Lloyd-Max* algorithm is an iterative method that does just that. The crude steps (of one of the versions of this algorithm) are as follows:

1. Knowing b_0, assume y_1.
2. Using (1), find b_1.
3. Using (2), find y_2.

and so on...

We also note that since we know the (approximate) signal statistics, we know b_M. Then we have an idea of how much of the error the algorithm made by seeing how close it is to the known value of b_M after the last iteration. If it is too far off, we reinitialize and try again until we are within the accepted tolerance.

Later, we will see a more complex, but better performing method of vector quantization.