Speech Compression Using Vector Quantization: The LBG Algorithm

Adam Hess
Binghamton University
EECE 523 Data Compression

Abstract - The purpose of this paper is to give an in-depth description of vector quantization applied to speech signals. We will explore the famed LBG algorithm to numerically calculate appropriate reconstruction levels for the vector quantizer and show why vector quantization is an excellent form of compression for highly correlated signals like speech.

Introduction:

While the size of hard drives are on a constant rise, they are never large enough to satiate our thirst for storage. Our desire for high quality images, videos and sounds stored on our computers or transferred over the web is met with limited bandwidth and hard drive space. Ideally, we would like to be able to make the data stream as small as possible. Data compression attempts to solve this problem either through lossy or lossless compression.

Lossless compression schemes attempt to take advantage of the statistical properties of a signal. Their goal is to make values that occur frequently be represented by as few a number of bits as possible and less frequent values take up more bits. The end result is the average number of bits of the data stream is smaller; therefore, we have effectively compressed the data.

The downside of lossless compression schemes, such as Huffman and Arithmetic coding, is that they are relatively poor at compressing data. They have a compression ratio that ranges from 1.16 to 2.03 [1] which varies largely depending on the type of signal being used, i.e., text, audio, video.

Lossy compression, on the other hand, attempts to eliminate "redundancies" in the data by simply discarding them. A lossy compression scheme cannot perfectly reconstruct the original signal, but its goal is to create an approximation of the original signal. This is often based on human auditory or visual perception. Our eyes and ears are insensitive to noticing the errors that this approximation creates.

The fundamental element in a lossy compression scheme is quantization. Analog signals span an infinite range space; therefore, all values cannot be represented by a finite number of bits. A quantizer takes this infinite range and fixes it into quantized intervals that are multiples of a step size. If we use a step size of 1 unit, values that fall in between these units will get fixed to the closest step level. For example, 0.4 will get quantized to 0 and 3.8 will get quantized to 4. The image below is a representation of a mid-tread quantizer. X-axis is the range of input values that get mapped to a fixed Y value.

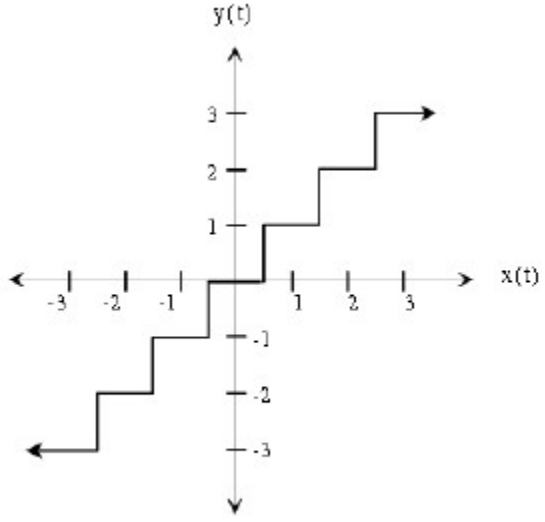

Figure 1 Mid-tread quantizer

A simple way to compress the data is to reduce the number of bits used to quantize the input signal; this can be done two ways, each having similar effects. The first would be to simply increase the step size of the scalar quantizer so that it is less sensitive to variation in the signal. This reduces the number of levels needed to represent the data. The second way is to remove the Least Significant Bit (LSB) of each of the quantized samples; fewer bits are needed to represent the data. Chopping off an LSB is effectively increasing the step size of the data by a factor of two.

Vector Quantization:

Consider the case where we are try to quantize two different, mutually exclusive, events such as the number of cars manufactured and the temperature of your car driving on a breezy day. Two, completely independent events and each would require their own vector quantizer.

The total number of levels required for this would be the number of levels to quantize the number of cars manufactured and one to quantize the temperature in your car. So if we used 8 bits for each quantizer we would need 16 bits to represent both pieces of data.

Let's consider the alternate case: What if we are trying to compress two highly dependent features, such as the temperature of a bar of metal and its length. There is an obvious relationship between the length of the bar and the temperature. As the temperature rises, the bar gets linearly longer, as can be seen in Figure 2.

If we assumed that the length and temperature of the metal are independent from one another and use two quantizers to represent each, we can see that we would be wasting a significant portion of the values that we can represent. An ideal quantizer would quantize only along the line that represents the temperature-length relationship.

Vector quantizers take the temperature and length values and represent them as a single vector. By using fewer levels and bits represent the data we can effectively compress this two dimensional problem into one dimension.

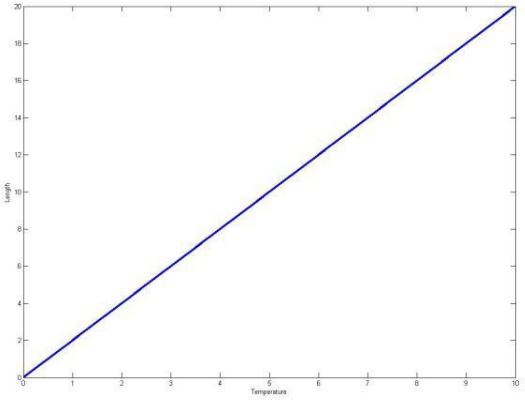

Figure 2 Temperature vs. Length of a metal bar

VQ and Speech:

Sucessive speech samples are like the relationship between Temperature and Length, they are highly correlated. Figure 3 shows a scatter plot of a large set of speech samples plotted against their amplitudes. The-X axis represents one sample, the Y-axis represents the sample that immediately succeeds that sample.

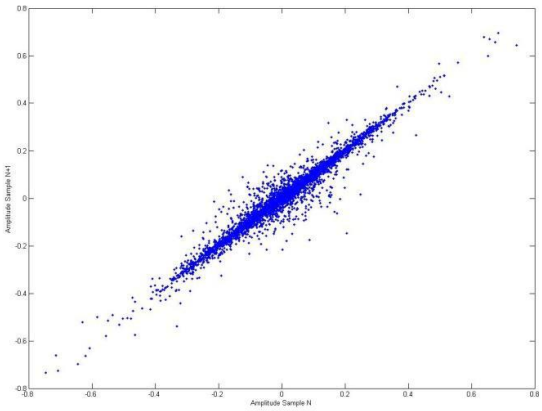

Figure 3 Relationship Between Successive Speech Samples

From this plot we can see that there is a clear relationship between the two samples because the plot slants. This means that there is some redundancy in the signal that we can take

advantage of and compress. Effectively, we can compress the speech sample directly by writing sample S[N+1] =S[N] and transmitting half the number of symbols.

While two and three dimension relationships can be visibly observable, , as we get to higher and higher order models we struggle to imagine or understand the relationship between 8, 16 or even 32 samples. This does not mean that there is not a relationship between them.

We can use the Linde-Buzzo-Gray LGB algorithm, also known as K-means clustering algorithm to simplify the process of generating reconstruction levels. Its goal is to reduce the distortion of the signal by reallocating bit locations to the most important parts of this signal.

Linde-Buzo-Gray (K-Means Clustering) Algorithm:

As described before, the goal of the LBG algorithm is to create a set of reconstruction levels that will better conform to the relationship between multiple dimensions, or in the case of speech, samples; secondly, by doing so we will reduce the distortion by placing more points along the trend line between the dimensions instead of wasting them in dead space. Thirdly, by representing multiple dimensions by a single index we can effectively compress the data by increasing the number of successive samples represented in the vector quantizer.

The LBG algorithm stems from pattern recognition, i.e., it is a learning algorithm. It updates the reconstruction levels based on how the old one performs, i.e., learning how the data is shaped and select a point closer to the center of the clusters.

The algorithm for the discrete case is given from [2] as:

1) Define an initial set of reconstruction levels

2) For each training vector compute the Euclidean distance between it and every reconstruction level.

3) Define a group for each reconstruction level that consists of the training vectors with the smallest Euclidean distance.

4) Calculate the mean squared error associated with each reconstruction level currently being used.

5) If the change in MSE or the MSE has met the desired condition Stop otherwise:

6) Find the centroid, i.e., the mean, of the set of vectors in each group of training vectors.

7) Define the centroid as the new reconstruction levels

8) If a reconstruction level does not have any training vectors associated with it, move it to the reconstruction level with the most number of elements in the cell and add a small perturbation.

9) Go to step 2 until the desired number of iterations have been complete

While the LBG will reduce the distortion in the system it is not guaranteed to converge to the optimal solution. Each set of initial conditions will result in a different solution.

One way to come up with reasonable reconstruction levels is to visually inspect the training data (if possible) and select points that lie in the center of clusters or along the trend line of the data. Another method is to use random vectors from the training data and uses those as the initial conditions.

The original paper on the LBG algorithm, mentioned by [2], anticipated using the *splitting technique.* A single element code book is used which is at the centroid of the entire training set. The single point is then split into two points, slight perturbation, and the LBG algorithm is applied. The two points are split into four points and the LBG algorithm is applied again. This process is repeated until the desired number of points is obtained.

Encoding/Decoding VQ:

Because we are no longer quantizing two independent events, a different method for encoding the data is used. First each reconstruction value is assigned an index. Secondly, the Euclidean distance between the input sample and all the reconstruction levels are calculated. Thirdly, the reconstruction level with the smallest Euclidean distance is selected as the assigned reconstruction level. Lastly, the encoded data is the assigned reconstruction level's index so that the data stream is simple the reconstruction level indexes.

Decoding the signal is a much easier task than encoding. The index is matched in the index in the codebook. Once the index is found, the values in the codebook are assigned as the reconstruction level.

The computation complexity of the encoding process is much more than then the complexity of the decoding processes. The number of compares needed for the encoding process is O(NK) where as the decoding processes can be simplified to as much as Q(log2(N)), or O(1) when using hash tables, where N is the number of samples being processes and K is the number of reconstruction levels.

Results:

Five speech files were used to generate the reconstruction levels. They consist of a woman saying "Hello sailor," a speech by Pres. John F. Kennedy, a male German speaker, an English female speaker and a segment of Pres. George W. Bush talking. The LBG algorithm was initialized by selecting random vectors from the concatenated speech file, but not from the training vectors. The algorithm completed one hundred iterations before being complete.

A test file was used that was not included in the initial training vector files. It is a different snippet of Pres. George W. Bush talking. The test file was encoded then decoded by a second algorithm that relied on the reconstruction levels of the LBG algorithm. The mean squared error was calculated for various bit rates and can be seen in Figure 4.

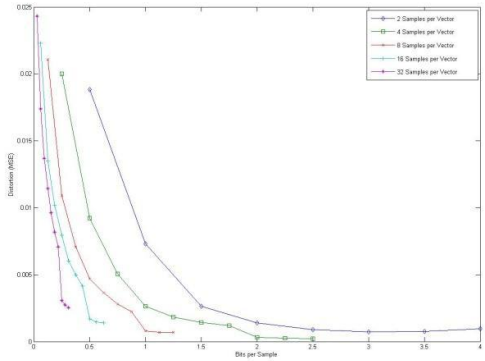

Figure 4 Rate Distortion Curve for Vector Quantizer

Each plot represents a different number of samples used per each vector. As the dimension of the vector increases we can see that both the distortion and the number of bits per a sample decrease dramatically. The curves are plotted with power of 2 increasing number of reconstruction levels and vector representations. The number of points represented on the higher order curves was limited by the computational time required to obtain those operating points. (Processing time would have taken several hours for a single point.)

We can immediately see the advantage of using higher order vector quantizer. As the number of successive samples is increased, higher dimensionality, the bit rate is lowered. As for the distortion, if we pick a point on the lowest curve and move to the right, we can see that in order to obtain the same distortion criteria in a lower dimensional vector quantizer we need drastically more bits per sample. For example, look at the curve for 32 bits/vector at 0.21 bits/sample, and then compare it to the curve for 2 samples/vector at the same distortion. The latter rate-distortion is requires nearly five times as many bits per a sample, in this case 1 bit/sample, than the former case.

The trend seen in Figure 4 indicates that as we increase the number of samples per a vector, but leave the rate constant, the distortion of the signal decreases; this trend does not continue indefinitely. Eventually the performance of the

quantizer approaches an asymptotical barrier that prevents any significant improvement in the rate distortion curve. An example of this can be seen in Figure 5.

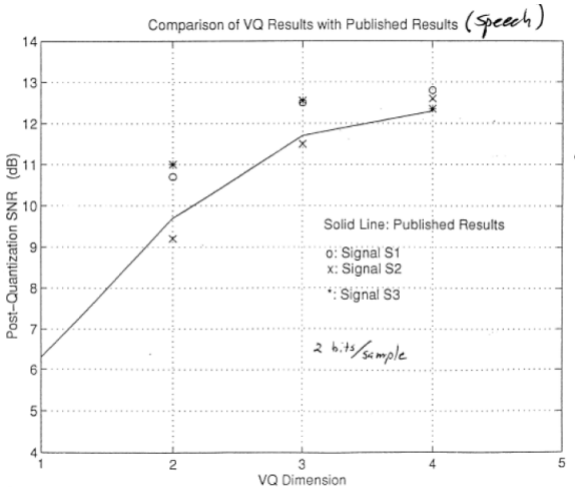

Figure 5 SNR for fixed bitrates and increasing dimension of vector quantizer.[1]

While unrelated to Rate-Distortion Curves, there is another beneficial feature to high dimensional vectors. For a fixed file size, increasing the dimension reduces the computation time required to process the same fixed length file. If a file is **M**-points before encoding, and the encoding vector is **L** dimension, we would require **N = M/L** number of points to encode the data. This ultimately leads to **MK/L** number of compares to encode the data. So as the dimension of the data is increased, the time required to encode the file is decreased.

Conclusion: We have explored vector quantization and shown that it is very useful for compressing multiple points of data that show a strong correlation between them. Vector quantizers produce better rate distortion curves and immense compression ratio until we reach the asymptotic limit. Overall, vector quantization is an effective means of lossy compression

References:

[1] M. Fowler EECE 523. Class Lecture, Topic: "Ch. 3 Huffman Coding." SUNY Binghamton, Binghamton, NY. Spring 2011.

[2] Kahlid Sayood. "Vector Quantizer" in *Introduction to Data Compression,* Third Edition. New York: Morgan Kaufmann, 2006, pg 273-324