A Comparison of Quantization Methods

Kyle Kastner and Johanna Hansen

***Abstract*—Quantization is an important technique in modern communication and data systems. Linear, non-linear, and vector quantization implementations have different characteristics and implementation concerns. Quantifying the differences in performance, and discussing the implementation issues associated with each technique, we show some of the trade offs made when choosing one of these quantization methods.**

***Index Terms*—Quantization, linear, vector, μ-law, k-means.**

I. Introduction

QUANTIZATION methods are crucial to modern technology. Quantization techniques have fueled advancements in communications, computing, and data processing. In this paper, we will compare linear and non-linear quantization methods at various bit levels, and discuss the complexities associated with each technique.

A. Background

Conveying information with minimal data has been and continues to be an important part of technology and research. Passwords, military codes, cellular communications, and compression are either in whole or in part based on this concept. One of the simplest ways to achieve data reduction is to eliminate data that does not transmit useful information. This elimination, done correctly, can greatly reduce the amount of data required to transmit information, such as words in a conversation.

B. Overview

Several different methods of quantization will be shown and compared in the following pages. Implementations of linear, μ-law, and vector quantizers were run against different audio files, generating results for mean squared error and signal to noise ratio. We also take special care to discuss the implementation details of each quantization type.

C. Goal

Our goal in this paper is to identify different ways to perform quantization, compare the performance of each method discussed, show implementation examples for each, and discuss the engineering trade-offs associated with choosing a particular quantization scheme.

II. Error

Quantization is considered a "lossy" scheme, which means that during the quantization operation, data is thrown out and cannot be recovered. This data may or may not contain useful information about the signal being quantized. When data is lost, error is introduced, known as quantization error. Given a sampled input x_n, quantization error can be shown as [1]

$$x_q = x_n + e_n \tag{1}$$

where e_n is the quantization error. Quantizer performance will be measured by comparing the error introduced by different methods while holding system inputs and parameters fixed. Two basic measurements convey the amount of error introduced, Normalized Root Mean Squared Error (NRMSE) and SNR (Signal To Noise Ratio).

III. Linear Quantization

A. Formulation

Linear quantization is the simplest method for applying quantization to a signal. The linear quantizer can be implemented as a sample and hold operation in hardware, or by the integer division method available in most computer languages. For an input sample x_n, the linear quantizer can be formulated as follows [1]

$$x_q = int(\frac{x_n * 2^B}{x_{max} - x_{min}}) \tag{2}$$

where B is the number of quantization bits.

B. Performance

The linear quantizer has near-zero computational overhead, and with a relatively large number of quantization bits or a narrow dynamic range, quantization error can be fairly low. However, the linear quantizer is a poor choice for many systems, as there are alternatives which introduce less quantization error for the same system parameters x_{max}, x_{min}, and B.

C. Visualization

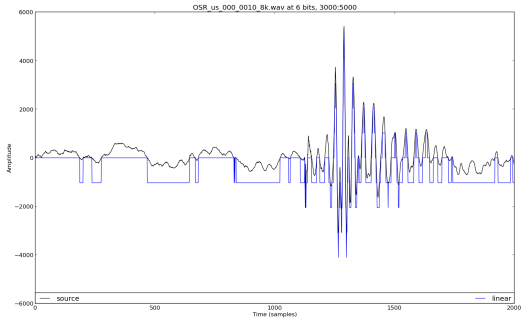

Fig. (1): Linear quantization, $B = 6$

The image in Figure 1 shows an example of linear quantization, applied to an audio file containing speech.

IV. Non-linear Quantization

Non-linear quantization covers a wide variety of quantization techniques. One of the most common non-linear quantizers is based on the μ-law algorithm, which uses logarithmic formulas to map input to a compressed domain (companding), apply a linear quantizer, then invert the companding stage (expanding) in order to perform non-linear quantization. There are other methods of non-linear quantization that use different algorithms, such as the A-law formula or specialized formulas for specific data types.

A. Formulation

The companding stage of the μ-law quantizer uses the following formula [2]:

$$x_q = \frac{ln(1+\mu|x_n|)}{ln(1+\mu)} sign(x_n), -1 \leq x_n \leq 1 \tag{3}$$

Linear expansion as in Equation 2 is applied, then an inversion formula is applied, where

$$y_n = \frac{1}{\mu}((1-\mu)^{|y_q|} - 1), -1 \leq y_q \leq 1 \tag{4}$$

shows the inversion formula.

B. Performance

The μ-law quantizer adds a minimal amount computational overhead, in exchange for superior error rates for commonly quantized data types (music, speech, etc.). The μ-law quantizer can be extended to an adaptive μ-law quantizer, or combined with the A-law quantizer to form a switching adaptive scheme.

C. Visualization

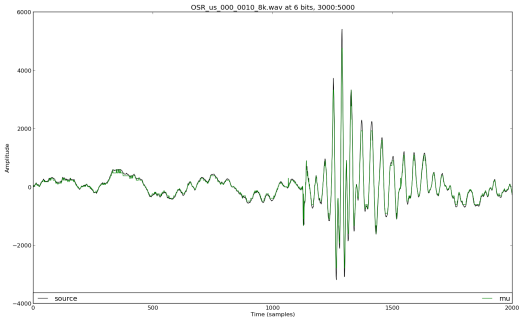

Fig. (2): μ-law quantization, $B = 6$, $\mu = 255$

V. Vector Quantization

Vector quantization can be seen as a further extension of non-linear quantization. Rather than assigning boundaries based on a set formula, the vector quantizer uses k-means clustering of an example data set to intelligently set bit boundaries. There are many issues to be dealt with when implementing the k-means clustering section of the vector quantizer, which will be discussed in the sections below.

A. K-means Initialization

K-means clustering is a technique for associating a set of values with k cluster centroids, where k is a predefined integer. This works very well for quantization, as the value for B is pre-determined, which means $k = 2^B$. To perform clustering, there are two primary stages: initialization, and association. K-means clustering is extremely sensitive to initialization, and there are several different ways to perform the initialization stage.

1) Value Initialization: Value initialization is the simplest way to initialize k-means centroids.

$$c_n = V, n = 1...k \tag{5}$$

Examples for V include 0, the mean of the data set, the median of the data set, or the mode of the data set.

B. Linear Initialization

Linear initialization uses a similar approach to the linear quantizer, seen in Equation 2, to set centroid cluster values.

$$c_n = x_{min} + \frac{n(x_{max} - x_{min})}{2^B}, n = 1...k \tag{6}$$

C. Probabilistic Methods

1) Forgy Method: The Forgy method for initialization takes k samples at random from the data set, and uses those values to initialize the k-means centroids.

$$c_n = X, X \in \{x_1...x_n\}, n = 1...k \tag{7}$$

2) Unknown Distribution Sampling: The set of data to be quantized can be thought of as data with an unknown distribution. There are several techniques for sampling from unknown distributions, including the rejection sampling and Metropolis-Hastings sampling techniques, which are the two techniques used for comparison in this paper. The implementation of these two algorithms is outside the scope of this paper, but more information can be found in [3].

D. Sculptor's Method

The Sculptor's method is a new method developed during work on this paper. While testing the different ways of initializing the k-means centroids, there seemed to be large performance differences between the repeatable methods (Value, Linear) and the probabilistic techniques (Forgy, Rejection, Metropolis-Hastings). Unfortunately, probabilistic methods are difficult to test and quantify, due to the sensitivity of the k-means algorithm to initialization. A repeatable method for initialization which provides performance approaching that of the probabilistic methods is desired. Method based on a center(s) of mass calculation were considered, but further exploration led to the creation of the Sculptor's method, formulated below.

$$c_n = x_{min} + \frac{n(x_{max} - x_{min})}{2^B} \frac{\sum_{i=0}^{n} count(x_{min} + i)}{n}, n = 1...k \tag{8}$$

If one imagines the linear initialization method as a flat piece of clay, with lines evenly spaced across the length, then the

placement of centroids for the Sculptor's method can be found by compressing or expanding different sections of the clay. Compressing sections will decrease the distances between lines, while stretching the clay will increase distance between lines. By thinking of the histogram of data as a "map" of pressure applied to the clay, the distance between lines is inversely proportional to the summed count of values per bin (histogram area).

E. Formulation

Once initialization is completed through some method, the k-means algorithm iterates through each point in the data set, assigning each data point to a centroid.

$$x_c = min((x_q - c_n)^2), n = 1...k \tag{9}$$

After a data point is assigned to a given centroid, the centroid value is updated.

$$\forall x \in x_c, c_n = c_n(1-\alpha) + x\alpha, n = 1..k, 0 < \alpha < 1 \tag{10}$$

Here x_c represents the set of values associated with each cluster by the previous assignment stage, and $\alpha = \frac{1}{v}$, where v is roughly the amount of data to "remember" during the update stage. A value of $\alpha = 0.01$ was used for this testing. Once all data has been processed, each value associated with a cluster centroid is set to the value of that centroid. [4]

F. Performance

Vector quantization has much more overhead than the previously discussed linear and non-linear techniques. Implementation is more complex, and there is a wide performance difference which depends on the initialization of k-means centroids. However, basing quantization levels on the content of the data instead of an what the content is expected to be has some performance benefits, at the cost of introducing a large amount of delay in order to keep the system causal.

G. Visualization

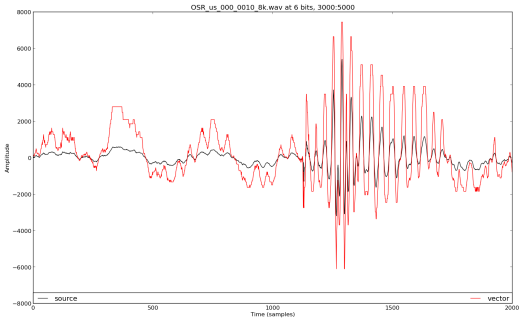

Fig. (3): Vector quantization, $B = 6$, $init = Sculptor's$

VI. Comparison

The two measurements used to compare the various quantization schemes discussed above were Normalized Root Mean Squared Error (NRMSE), shown in Equation 11, and Signal to Noise Ratio (SNR), shown in Equation 12. [1]

$$NRMSE = \frac{\sum_{i=0}^{n}(x_n - q_n)^2}{n}, n = length(x) \tag{11}$$

$$SNR = 10log(\frac{\sigma_x}{\sigma_q}) \tag{12}$$

The files used for this testing, as well as generation of Figures 1,2,3, were found at [5]. These particular tests were run against the "OSR_us_000_0010_8k.wav" file, though other speech files from this site show similar results. Applying the measurements to each of the quantization schemes, the following figures were generated.

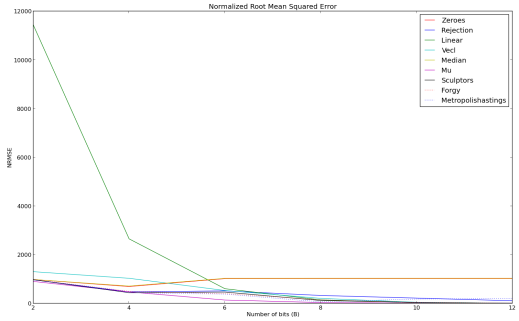

Fig. (4): Normalized Root Mean Squared Error

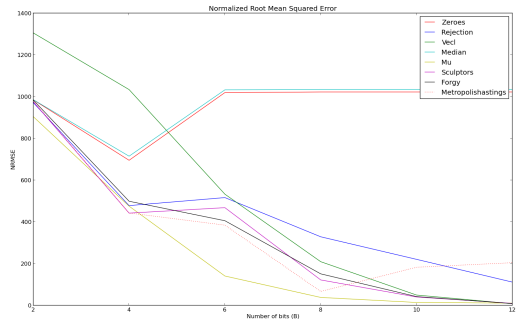

Fig. (5): Normalized Root Mean Squared Error, excluding linear quantization

VII. Future Work

More exploration and testing of the Sculptor's method is necessary. Testing against data sets besides speech data, as well as a larger sample set for quantifying the performance of the probabilistic methods, will go a long way toward validating the results of this initial exploration. There is also room to test the Sculptor's method, without follow on k-means clustering, as a different method for non-linear quantization. Implementations of the A-law quantizer, and adaptive

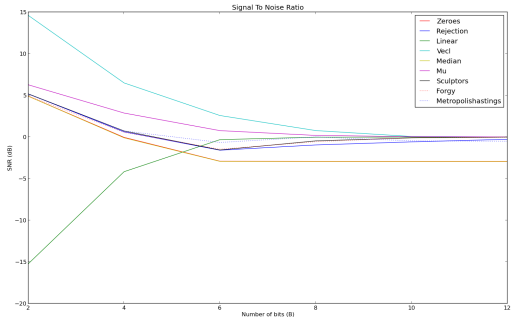

Fig. (6): Signal To Noise Ratio

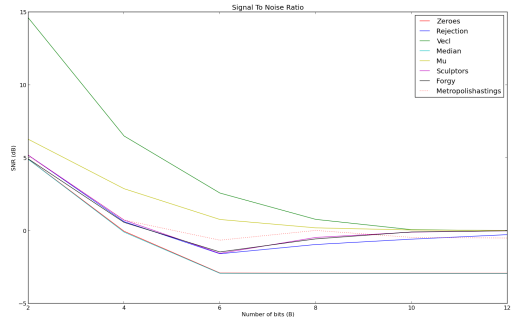

Fig. (7): Signal To Noise Ratio, excluding linear quantization

quantization schemes would add breadth to the test coverage, and allow for more accurate comparisons between the various methods.

VIII. Conclusion

This paper explored various algorithms for quantization. Using Normalized Root Mean Squared Error and Signal to Noise ratio measurements, linear, non-linear, and vector quantizers were compared for English speech data. A new type of initialization for vector quantizers, named the Sculptor's method, was introduced. This method offers repeatable output, simplicity, and performance, and seems to compare favourably with existing techniques for vector quantization initialization. Readers of this paper should find the information here helpful in determining which quantization methods are most appropriate for a given system.

References

[1] A. Oppenheim, R. Schafer, and J. Buck, *Discrete Time Signal Processing*, 2nd Edition. Prentice-Hall, 1999.

[2] Cisco, *Waveform Coding Techniques*. Retrieved November, 2012. http://www.cisco.com/en/US/tech/tk1077/technologies_tech_note09186a00801149b3.shtr

[3] A. Gelman, J. Carlin, H. Stern, D. Rubin, *Bayesian Data Analysis*, 2nd Edition. CRC Press, 2004.

[4] S. Lloyd, "Least squares quantization in PCM", *IEEE Transactions on Information Theory*. Volume 28, Issue 2, 1982.

[5] "OSR_us_000_0010_8k.wav", *Open Speech Repository*, American English. Telchemy Incorporated, Retrieved November, 2012. http://www.voiptroubleshooter.com/open_speech/american.html