Sonar Data Compression using Non-Uniform Quantization and Noise Shaping

Lok S. Wong, Gregory E. Allen, and Brian L. Evans
Applied Research Laboratories and
Dept. of Electrical and Computer Engineering
The University of Texas at Austin
Austin, Texas 78713
lswong@arlut.utexas.edu, gallen@arlut.utexas.edu, bevans@ece.utexas.edu

***Abstract*—Sonar sensor arrays potentially produce huge amounts of data to be recorded or transmitted over a telemetry system. Compression can reduce the required storage or transmission bandwidth, or allow a larger or higher fidelity array. We use a dataset for a sonar array receiving acoustic communication signals from a transmitter in a lake test. We compress the received signals to evaluate the effect of compression on performance. Based on analysis of the dataset, we use non-uniform quantization with a Laplace distribution along with noise-shaped feedback coding. We demonstrate that this sonar data can be compressed from 16-bit to five-bit values with little or no change in performance using our technique.**

I. INTRODUCTION

A sonar array system typically produces a very high rate of data output, especially when used for imaging or navigation. Typical sonar systems can have data rates as high as 1 GB/s [1]. Compressing data reduces the amount of resources required for storage and transmission.

One way to compress sonar data is by passing the discretized voltage values produced by hydrophones through a non-uniform quantizer whose distribution closely matches to that of the quantizer input. Those quantization values are then used by the receiver to process and decode the input signal. Meanwhile, the quantization error is fed back to the quantizer through a low-pass filter so that the quantization noise is shaped into higher, out-of-band or otherwise less important frequencies.

In November 2009, engineers from Applied Research Laboratories (ARL) conducted field measurements on Lake Travis in Austin, TX, using a single transmitter to transmit packets to a five-element sonar array [2, 3]. Fig. 1 shows bathymetric data of the testing site. The receiver was on the test station while the transmitter was on top of the boat [3]. These measurements were initially conducted to evaluate various Doppler correction algorithms. We used the dataset made publicly available in [4] to observe how much compression was possible. This dataset contains 360 packets of acoustic communication signals modulated by binary phase shift keying (BPSK) or quadrature phase shift keying (QPSK), and that each has about 0.5 s of samples. 29 QPSK-modulated packets were used as a representative subset of the dataset for analysis in [3].

This research was supported by the Independent Research and Development Program at Applied Research Laboratories: The University of Texas at Austin.

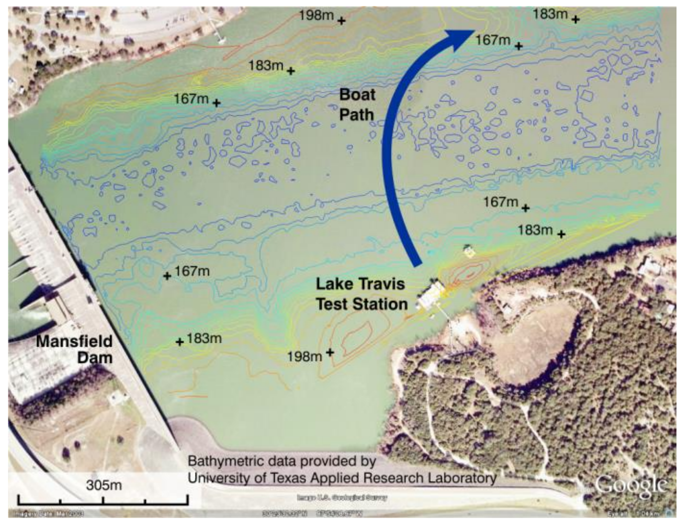

Fig. 1. Aerial view of the Lake Travis testing site with lake bed elevations in meters above sea level. The water level is at 198 m. Figure reprinted from [3].

In this paper, we describe the proposed compression system and how the noise shaper was designed. We compare the bit error rates (BER) of the packets using the Doppler correction algorithm concluded to be the best of the ones tested in [3] without this compressor and the BER of the packets using the same Doppler correction algorithm with the compressor. We demonstrate that this sonar data can be compressed from 16-bit to five-bit values with little or no change in performance.

II. BACKGROUND

Due to the high volume of data produced by sonar systems that often must be stored or transmitted in low-bandwidth acoustic channels [5], compression is very useful. One way is to use discrete wavelet transforms to compress the data [5]. However, using wavelet transforms cannot be based on input signal changes [6] and they smooth out sharp edges in sonar images [7], both of which are undesirable in applications requiring detection of sudden input changes.

To compress the sonar data, we propose using data conversion to decrease the number of the bits that represents each sample. We can accomplish this by passing the analog-to-digital converter (ADC) output through a quantizer with 2^B

levels, where B is the number of bits we want to compress each sample to. While this creates greater quantization error, if we pass the quantization error signal through a low-pass filter (LPF) and feed it back to the system, we can "shape" the quantization noise power towards frequencies that will subsequently be filtered out [8]. Fig. 2 shows a block diagram of the compressor and decompressor. In Fig. 2, x is the input signal, Q is the quantizer, Q^{-1} is the inverse quantizer, H is the feedback LPF, y is the compressed output signal, and $\hat{x}$ is the decompressed output signal.

We can further improve performance by taking into account the distribution of the data. Fig. 3 shows a histogram of one of the packets we analyzed that is representative of all the packets in the dataset. We see from Fig. 3 that the data has close to a Laplace distribution. For a random variable with a Laplace distribution, the probability density function (PDF) is

$$f(x) = \frac{1}{2b}\exp\left(-\frac{|x-\mu|}{b}\right) \tag{1}$$

and the cumulative distribution function (CDF) is

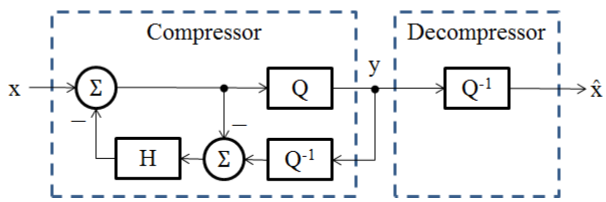

Fig. 2. Compressor and decompressor block diagram.

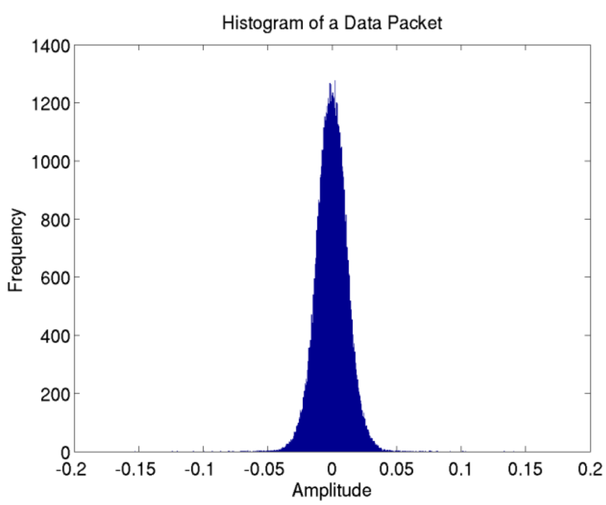

Fig. 3. Histogram of a data packet.

TABLE I.
MSE BETWEEN DATA DISTRIBUTION AND REFERENCE DISTRIBUTIONS

Reference Distribution	Mean Squared Error
Laplace	0.000549
Gaussian	0.001724
Logistic	0.001229

$$F(x) = \begin{cases} \frac{1}{2}\exp\left(\frac{x-\mu}{b}\right), & x < \mu \\ 1 - \frac{1}{2}\exp\left(\frac{x-\mu}{b}\right), & x \geq \mu \end{cases} \tag{2}$$

where μ and b are parameters that define a Laplace distribution. Here, μ is the mean and b is related to the standard deviation σ by the equation

$$b = \sigma/\sqrt{2}$$

To validate that the Laplace distribution is a good fit, we compute the mean square error (MSE) between the empirical CDF (ECDF) of the data and the CDF of the Laplace and similar distributions with the same mean and variance as those of the data. We generate an ECDF using a histogram with 1,000 evenly spaced bins, compute the error between the ECDF of the data and the CDF of a Laplace, Gaussian, and logistic distribution at the bin edge points, and average the squared errors. As Table I shows, the Laplace distribution has the best fit among common distributions similar to it. Using (1) and (2), we can design a non-uniform quantizer where the quantization levels resemble a Laplace distribution based on the statistics of the data [9].

III. METHODOLOGY

The first step in making the compressor was to design the non-uniform quantizer. When the quantizer is given σ, a lookup table of quantization thresholds is generated such that the quantization levels has a Laplace distribution with mean 0 and standard deviation σ, and 0 is one of the thresholds. The thresholds are created such that the difference between the CDF values of two adjacent thresholds is constant. Further, the CDF value of each threshold is halfway between the CDF values of the threshold's two adjacent levels.

We next took into account the fact that a Laplace distribution extends to infinity while we know the input will always be bounded. First, we set bounds to the quantization values based on the range of the ADC output. In this case, the range was -1 V to 1 V. Knowing this, we designed the quantizer such that the quantization levels represent a truncated Laplace distribution. Therefore, the quantization levels represent a distribution with PDF

$$g(x) = \begin{cases} \frac{1}{2bh}\exp\left(-\frac{|x|}{b}\right), & -1 \leq x \leq 1 \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

and CDF

$$G(x) = \begin{cases} \frac{1}{h}\left[\frac{1}{2}\exp\left(\frac{x}{b}\right) - F(-1)\right], & -1 \leq x < 0 \\ \frac{1}{h}\left[1 - \frac{1}{2}\exp\left(\frac{x}{b}\right) - F(-1)\right], & 0 \leq x \leq 1 \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

where

$$h \triangleq F(1) - F(-1)$$

evaluated from (2) when $\mu = 0$ [10].

Another consideration we made is what level value the quantizer would assign if the input was below the lowest threshold or above the highest threshold. Although we know the limits of the ADC output, it may not necessarily be good to have -1 and 1 as quantization levels, which correspond to a CDF of 0 and 1, respectively. Unless the ADC is at least close to being saturated, most likely there are no receiver samples that are near those bounds. Because of this, we assigned special levels for the two extreme cases. The special levels are determined using the formulae

$$L_1 = T_1 - (L_2 - T_1) = 2T_1 - L_2 \tag{5}$$

$$L_{2^B} = T_{2^B-1} + \left(T_{2^B-1} - L_{2^B-1}\right) = 2T_{2^B-1} - L_{2^B-1} \tag{6}$$

where L_n is the *n*th level and T_n is the *n*th threshold between L_n and L_{n+1}. Fig. 4 shows an example of a 4-bit Laplace quantizer with $\mu = 0$ and $b = 0.2$, with the aforementioned conditions included.

Although the quantization thresholds are zero-mean, an input signal with a nonzero mean that did not saturate the ADC can be accommodated by subtracting the mean from the compressor input, and then adding the mean to the decompressor output. If a signal with nonzero mean does saturate the ADC, then the quantization thresholds will likely need to be generated to have a nonzero mean for optimum performance. Since that was not a problem in this case because the minimum and maximum input values were not close to the minimum and maximum ADC output values, we can then design a quantizer that can tune to the input by adjusting both the mean and variance of the quantization levels based on the mean and variance of the input.

Note that σ does not have to be equal to the standard deviation of the data. In fact, setting σ to be the standard deviation of the input signal results in too high quantization errors for input values distant from the mean. By setting σ to be slightly greater than the standard deviation of the data, those input values can be better received at the cost of higher quantization errors for input values around the mean. If σ of the quantization levels is set too high, quantization errors for inputs around the mean will be too high. After trying different values of σ, we found that the Laplace distributed quantizer worked well for all the packets when σ was set to about 1.4 times the standard deviation of the input.

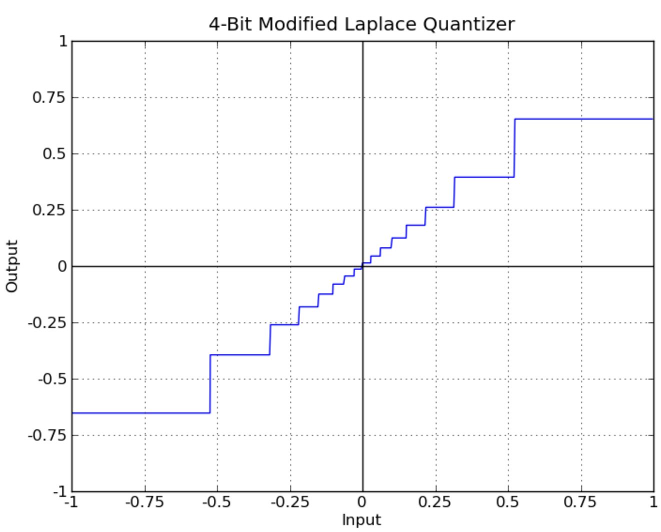

Fig. 4. 4-bit modified Laplace quantizer with $\mu = 0$ and $b = 0.2$.

The next step was to have a LPF that shapes the quantization noise to frequencies outside the band of interest. We used an eight-tap Hodie window [11] as the feedback filter. The Hodie window has significant stopband attenuation with a small transition region using only a small number of coefficients.

To test how well the compressor works, we compare the BER of each packet analyzed in [3] from each of the five receiver elements after Doppler correction without compression to the BER after compression, decompression, and Doppler correction. Since the receiver generated 16-bit samples, we tested performance using a quantizer with 2^B quantization levels, where $B \leq 16$. While [3] had several Doppler correction algorithms presented, we considered only the algorithm that was most reliable, which was to find the frequency with the highest power, remove the offset between the expected center frequency and that frequency, baseband the corrected signal, and apply an adaptive equalizer. We also considered the “overall” BER of each packet by comparing the input with an output of symbols made by taking the mode of the symbols decoded by the five receiver elements at each symbol time. In case of a tie, the symbol with the smallest magnitude and smallest phase in the range (-π, π] is chosen as the mode.

IV. Results

The initial performance results of the compressor are shown in Fig. 5 and 6. Fig. 5 shows results based on treating each element output of each packet separately. Fig. 6 shows results based on treating the five-element outputs of each packet as one output. For each output, we considered the Doppler correcting receiver with compression to have the “same” performance as the Doppler correcting receiver without compression if the BER with compression is within 5% of the BER without compression. This was done to take into the account the effect that small quantization noise introduced by the compressor could cause a symbol initially decoded correctly to be decoded incorrectly and vice versa. (From here onward, quantization noise will only refer to the quantization noise of the compressor and not the quantization noise of the ADC.) Accordingly, the output with compression is considered “better” than the output without compression if the BER with compression is less than 95% of the BER without compression, and the output with compression is considered “worse” if the BER with compression is greater than 105% of the BER without compression. Results for $6 \leq B \leq 16$ are not shown because either they are similar to the case when $B = 5$ or they show 100% “same” performance.

From Fig. 5 and 6, the results show there was a trend of a higher percentage of worse outputs as the number of quantization bits is lowered. This is a result we expected. One of the results that we did not expect is that there were outputs that performed *better* with compression than without. In addition, there was an increase in the amount of better outputs

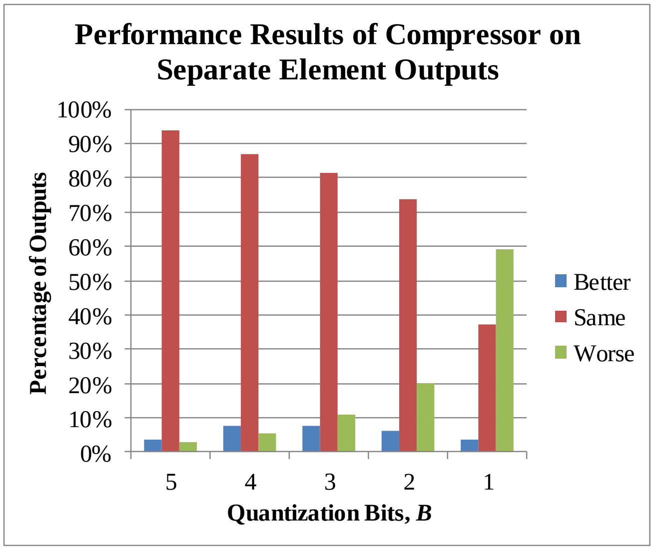

Fig. 5. Percentage of separate element outputs that performed better, same, and worse after being compressed than the original uncompressed separate element outputs.

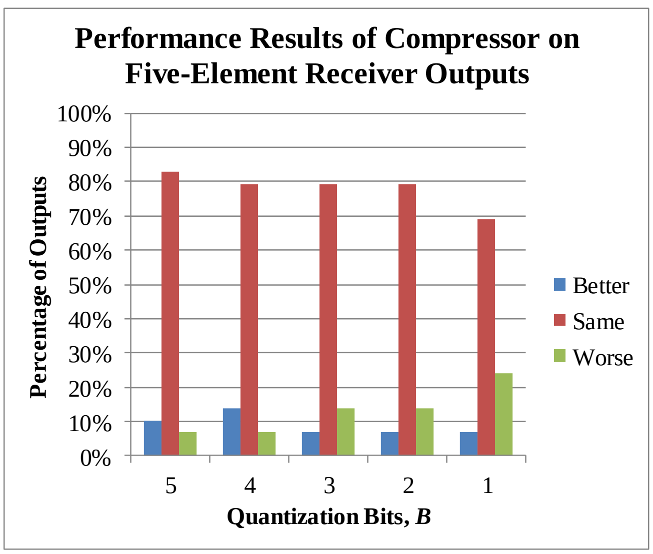

Fig. 6. Percentage of receiver outputs based on taking the mode of element outputs that performed better, same, and worse after being compressed than the original uncompressed packets.

from 5-bit to 4-bit quantization values. Furthermore, separate element outputs were more likely to perform the same or better with the compressor than the five-element receiver outputs except in the 1-bit and 2-bit quantizer case.

Upon further analysis, we found that two stages in the receiver were sensitive to quantization noise when the signal-to-noise ratio (SNR) of the input was sufficiently low. The first stage was the frame synchronizer. In this stage, the training sequence is correlated with the received packet, and the peak value is used to indicate where the data starts. When the SNR is low, the quantization noise can cause the peak correlation to be at a different point, causing the receiver to begin decoding somewhere else in the packet. The second stage was the

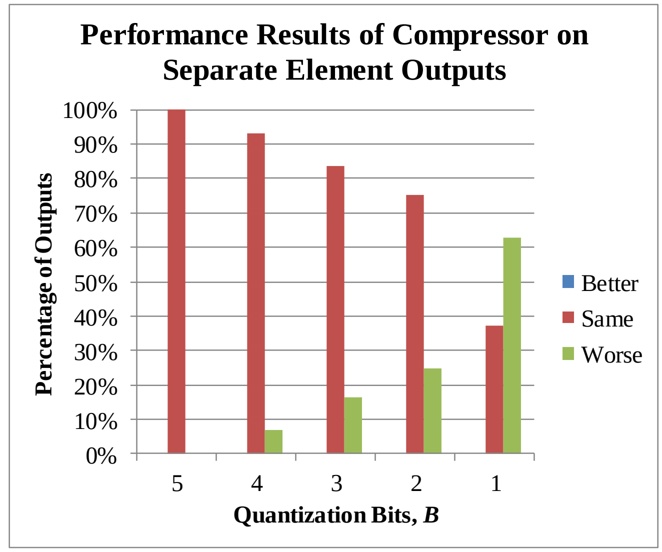

Fig. 7. Percentage of separate element outputs that performed better, same, and worse after being compressed than the original uncompressed separate element outputs by modifying the receiver with the compressor to determine the same data start and Doppler offset as the receiver without the compressor.

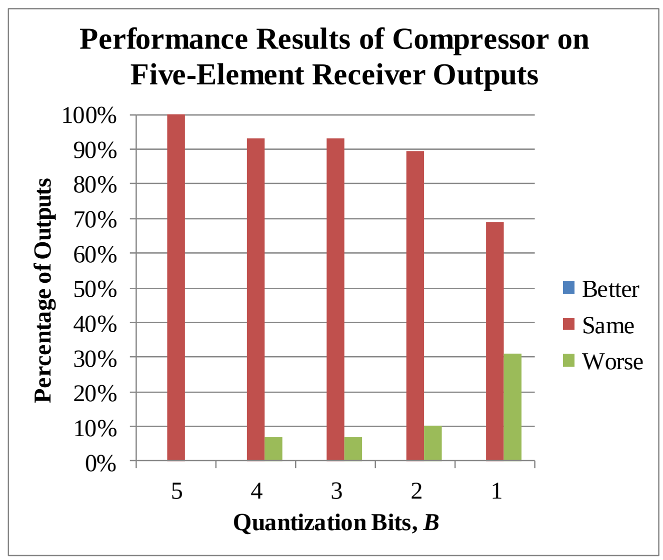

Fig. 8. Percentage of receiver outputs based on taking the mode of element outputs that performed better, same, and worse after being compressed than the original uncompressed packets by modifying the receiver with the compressor to determine the same data start and Doppler offset as the receiver without the compressor.

Doppler detector. In this stage, the receiver will look for the frequency with the highest magnitude and determine the offset between that frequency and the expected center frequency of a signal unaffected by Doppler effects. When the SNR is low, the quantization noise can cause the peak magnitude to be at a different frequency, causing the receiver to "correct" Doppler effects with a different offset. Slight changes in either the frame synchronization stage or the Doppler detection stage could cause a packet decoded well with the receiver without the compressor to be decoded poorly with the compressor.

Interestingly, the reverse effect can also happen, so a packet initially decoded poorly without the compressor can be decoded well with the compressor solely because the quantization noise by chance "corrects" the receiver in finding where the data starts or what the Doppler offset is. With this information, we then modified the receiver with compression to fix the data start location and the Doppler offset to the values determined by the receiver without compression and reran the results. The performance results are shown in Fig. 7 and 8. Fig. 7 shows results based on treating element outputs of each packet separately, and Fig. 8 shows results based on treating five-element outputs of each packet as one output.

From Fig. 7 and 8, there is once again a trend of a higher percentage of worse outputs as the number of quantization bits is lowered. However, there were no outputs of the receiver with compression that perform more than 5% better than their corresponding outputs of the receiver without compression. Under the assumption that the quantization noise does not affect the receiver's ability to determine where the data starts and what the Doppler offset is, the compressor using a 5-bit quantizer did not significantly alter the receiver's performance in decoding the packet. Finally, the five-element receiver outputs are just as likely as or more likely to have the same performance with the compressor as without the compressor than individual element outputs.

V. CONCLUSION

Our research has demonstrated that this sonar dataset can be compressed from 16 bits to five bits with little to no change in bit-error rates using a non-uniform quantizer tuned to the distribution of the data, and by shaping the quantization noise to frequencies outside the band of interest. We assumed that the signal-to-noise ratio is high enough so that the quantization noise of the compressor does not cause the receiver to significantly change where it determines the data starts and the center frequency of each packet. This technique could be applied to other types of sonar data so that the required storage or bandwidth can be lowered, or so arrays can increase in size or fidelity.

ACKNOWLEDGMENT

We would like to thank Karl Niemen, Kenneth Perrine, and the other authors of [3] for making their dataset public and for allowing us to use their code. We would also like to thank Todd Goodall for his technical contributions to this work.

REFERENCES

[1] J. Bridgman, G. Allen, B.L. Evans, "Scalable multi-core sonar beamforming with computational process networks," *Proc. Asilomar Conf. on Signals, Systems, and Computers*, Nov. 7-10, 2010.

[2] K. Nieman, K. Perrine, T. Henderson, K. Lent, T. Brudner, and B.L. Evans, "Wideband monopulse spatial filtering for large receiver arrays for reverberant underwater communication channels," *Proc. IEEE OCEANS*, Sept. 20-23, 2010.

[3] K. Perrine, K. Nieman, K. Lent, T. Henderson, T. Brudner, and B.L. Evans, "Doppler estimation and correction for shallow underwater acoustic communications," *Proc. Asilomar Conf. on Signals, Systems and Computers*, Nov. 7-10, 2010.

[4] K. Perrine, K. Nieman, K. Lent, T. Henderson, T. Brudner, and B.L. Evans, "The University of Texas at Austin Applied Research Laboratories Nov. 2009 five-element acoustic underwater dataset," http://users.ece.utexas.edu/~bevans/projects/underwater/datasets/index .html, 2009-2010.

[5] R. Cunha, M. Figueiredo, and C. Silvestre, "Simultaneous compression and denoising of side scan sonar images using the discrete wavelet transform," *OCEANS 2000 MTS/IEEE Conf. and Exhibition*, vol. 1, pp.195-199, 2000.

[6] B. Li and Q. Meng, "An improved SPIHT wavelet transform in the underwater acoustic image compression," *International Conf. on Measurement, Information, and Control*, vol. 2, pp. 1315-1318, Aug. 16-18, 2013.

[7] B. Li and X. Yang, "An adaptive lifting wavelet transform in the underwater acoustic image compression," *International Conf. on Measurement, Information, and Control*, vol. 1, pp. 479-483, May 18-20, 2012.

[8] S. Lipshitz and J. Vanderkooy, "Why 1-bit sigma-delta conversion is unsuitable for high-quality applications," *Proc. Convention of the Audio Engineering Society*, Amsterdam, May 12-15, 2001.

[9] A. Gersho and R.M. Gray, *Vector Quantization and Signal Compression*, Springer, 1992.

[10] N.L. Johnson, S. Kotz, N. Balakrishnan, *Continuous Univariate Distributions*, vol. 1, 2nd ed., 1994.

[11] E. Swanson, http://www-inst.eecs.berkeley.edu/~n247/matlab_files/ hodiewindow.m, 2001.