A Bi-level Block Coding Technique for Encoding Data Sequences with Sparse Distribution

Li Tan
Department of Electrical and Computer Engineering Technology
Purdue University North Central, Westville, Indiana
lizhetan@pnc.edu

Jean Jiang
Department of Electronics and Computer Engineering Technology
DeVry University, Decatur, Georgia
jjiang@devry.edu

Abstract

In this paper, we propose a simple and efficient method for encoding data sequences with the sparse distribution. The data sequence with a sparse distribution contains most data samples that can be encoded with a smaller number of bits per sample and a small number of large amplitude samples that require a larger number of bits to encode. The data sequence with the sparse distribution is often encountered as the residues from prediction in waveform, image, and video compression.

The proposed method divides the data sequence into two block sets. One is the level-0 block, where at least one sample in the block requires the maximum number of bits to encode, and the other one is the level-1 block, in which each sample in the block only needs a smaller number of bits to encode. Hence, coding efficiency could be achieved. We propose an algorithm to determine the optimal coding parameters, such as the block size and the number of bits for encoding each data sample in the level-1 blocks based on the sparse distribution of the given data sequence. Comparing with the traditional bi-level coding method [1], in which the coding parameters are obtained via an assumed Gaussian distribution function and pre-optimization, the proposed method achieves its coding efficiency using a real data amplitude distribution to determine the optimal coding parameters and is simpler to implement in real-time. In addition, the bi-level block coder is more robust to bit errors, as compared to instantaneous coding schemes such as Huffman and arithmetic coding schemes.

Introduction

Lossless compression of waveform data, such as audio, seismic, and biomedical signals [1, 2, 3, 4, 5, 6], plays a significant role in alleviating data storage and reducing the transmission bandwidth while achieving the recovered data in their original forms. One of the traditional lossless compression schemes involves two stages reported in references [1, 3, 4]. The first stage performs prediction, resulting in a residue sequence, in which each residue has reduced amplitude as compared to that of the original data. The residue sequence is ideally assumed to have a Gaussian distribution. The second stage further compresses the residue sequence using coding schemes such as bi-level coding [1], Huffman coding, and arithmetic coding [3, 4, 5] based on the statistical model of the Gaussian distribution. Although the Huffman and arithmetic algorithms offer high efficiency in compressing the residue sequence, they may suffer from several problems: 1) For a large sample size of residues, a large number of symbols must be used. Reference [3] successfully deals with such a problem by dividing the data sample size into equal intervals. First, it uses the arithmetic algorithm to encode the intervals with a less number of symbols. Second, it continues to encode each offset value, that is, the differences between the residue value and the center of its interval. This type of entropy coding often makes a real-time implementation difficult due to its complexity; 2) In practice, the predicted residue sequence exhibits a sparse distribution where some large peak amplitudes exit in the residue sequence; on the other hand, the residue sequence may not follow the Gaussian distribution well. In this case, compressing the residue sequence using the assumed statistical model may have less efficiency; 3) Huffman and arithmetic algorithms, without applying an error control scheme, are sensitive to bit errors due to the fact that these coding schemes produce instantaneous codes. A single bit error could damage all the decoded information.

This paper introduces a simple and efficient method, called bi-level block coding, for encoding a data sequence with a sparse distribution in general. The sparse distribution indicates that most of the data samples in the given data sequence have small amplitudes, requiring a small number of bits per sample to encode, and a few number of data samples have larger amplitudes that require a larger number of bits per sample to encode. The proposed method divides the data sequence into two block sets. One is the level-0 block, where at least one sample in the block requires the maximum number of bits to encode, and the other one is the level-1 block, in which each sample in the block only needs a smaller number of bits to encode. Hence, coding efficiency could be achieved. We propose an algorithm to determine the optimal coding parameters, such as the block size and the number of bits, for coding each data sample in the level-1 blocks according to the sparse distribution of the given data sequence. Comparing with the traditional bi-level coding method [1], in which the coding parameters are obtained via an assumed Gaussian distribution function and pre-optimization, the proposed method achieves its coding efficiency using a real data amplitude distribution to determine the coding parameters and is simpler to implement in real-time. In addition, the bi-level block coder is more robust to bit errors as compared to instantaneous coding schemes such as the Huffman and arithmetic coding schemes.

We first develop a bi-level blocking coding scheme and then apply it into a two-stage lossless compression scheme with applications to waveform data such as audio, seismic, and ECG (electrocardiography) signals.

Development of Bi-level Block Coding

In this section, we develop a bi-level block coding algorithm and determine its optimal coding parameters. Then, we verify its performance using a generated data sequence with the

Gaussian distribution. The performances will be presented in practical applications in the next section.

A. Bi-level Block Coding

To illustrate bi-level block coding, we consider the following 8-bit data sequence, which is considered to be sparsely distributed.

> Data sequence:
> 1, 10, 2, -1 -3, 126, 6, 14, -11, -10, 0, 12, -9, -10, 0, 2, -7, 15, -100, 2, 1,
> 0, 0, 1, -3, -4, 10, -8, 9, 11, -12, 10

Figure 1: Data Sequence with a Sparse Distribution

The above sequence has a sparse distribution. Thirty of 32 data samples have amplitudes less than 15, while only two of them (126, –100) have amplitudes close to the maximum magnitude value of 127. If we use an 8-bit sign magnitude format to encode these data, a total of 256 bits is required. Here, we describe a bi-level block coding scheme to take the advantage of data sequence with a sparse distribution. Bi-level block coding is depicted as follows.

1. We divide the data sequence with a length of $n = m \times x$ into m blocks, in which each block consists of x data samples; that is, x is the block size.
2. The sign magnitude format is used for encoding each sample. MSB (most significant bit) is used to encode the sign of data, while the rest of the bits are adopted for encoding the magnitude.
3. Two type blocks are defined below:
   a. The level-0 block of x samples is shown in Figure 2, where at least one of data samples in the block need the maximum number of bits including the sign bit, N_0. We encode each data sample in the level-0 block using N_0 bits; in our example, $N_0 = 8$ bits. To distinguish between the level-0 block and the level-1 block (described next), the prefix “0” is added to indicate the level-0 block.

0	$N_0, N_0, \ldots\ldots.., N_0$

Figure 2: Level-0 Block

   b. Second, the level-1 block of x samples is defined in Figure 3, where all samples in the level-1 block can be encoded by N_1 bits, including the sign bit. Correspondingly, the prefix “1” designates the block as a level-1 block.

1	N_1, N_1, …….., N_1

Figure 3: Level-1 Block

Notice that we require $N_1 < N_0$ such that the coding scheme is profitable.

4. With the given N_0, we must determine the optimal number N_1 and the block size x, such that the minimum number of bits is obtained for encoding the entire data sequence.

For example, assuming that we use $N_1 = 5$ bits for encoding each data sample in the level-1 blocks and the block size chosen as $x = 4$, then the encoding cost is as follows.

Encoded data	“1” 1 10 2 -1	“0” -3 126 6 14	“1” -11 -10 0 12	“1” -9 -10 0 2
Number of bits	1 5 5 5 5	1 8 8 8 8	1 5 5 5 5	1 5 5 5 5
Block type	Level-1 block	Level-0 block	Level-1 block	Level-1 block

“0”-7 15 -100 2	“1” 1 0 0 1	“1” -3 -4 10 -8	“1” 9 11 -12 10
1 8 8 8 8	1 5 5 5 5	1 5 5 5 5	1 5 5 5 5
Level-0 block	Level-1 block	Level-1 block	Level-1 block

Figure 4: Bi-level Block Coding Example

It is clear that two level-0 blocks are encoded with 25 bits each; whereas, six level-1 blocks are encoded using 21 bits each. We only need 176 bits for encoding the same sequence, as compared to 256 bits used in the case without using the bi-level block coding approach.

B. Optimal Coding Parameters

To derive the algorithm to obtain the optimal coding parameters N_1 and x, we make the following assumptions: 1) The probability of a data sample requiring less or equal to N_1 bits to encode is p, the probability of a data sample requiring the number of bits between N_1 and N_0 bits to encode is p_0, where p_0 is close to zero for a sparsely distributed data sequence; 2) All data samples are statistically independent. Then, the probability for a level-1 block with a block size of x samples could be written as

$$P_1 = p^x , \tag{1}$$

and the probability of a level-0 block is then expressed as

$$P_0 = 1 - P_1 = 1 - p^x . \tag{2}$$

Similar to [2], the coding length with k level-1 blocks and ($m - k$) level-0 blocks is given by

$$L(k) = m + N_0 x(m - k) + N_1 xk . \tag{3}$$

Again, note that using the binomial coefficient formula, the probability of a sequence having k level-1 blocks and ($m - k$) level-0 blocks is given below:

$$P(k) = \binom{m}{k} P_1^k \left(1 - P_1\right)^{m-k} \tag{4}$$

Substituting (1) in (4) leads to the following:

$$P(k) = \binom{m}{k} p^{xk} \left(1 - p^x\right)^{m-k} \tag{5}$$

We now obtain the average total length L_{ave} as the following:

$$\begin{aligned} L_{ave} &= \sum_{k=0}^{m} P(k)L(k) \\ &= (m + N_0 xm)\sum_{k=0}^{m} P(k) - (N_0 - N_1)x\sum_{k=0}^{m} kP(k) \end{aligned} \tag{6}$$

Equation (6) is further expressed in its closed form as follows:

$$L_{ave} = (m + N_0 xm) - (N_0 - N_1)xmp^x \tag{7}$$

With $n = x \times m$, we achieve

$$L_{ave} = \frac{n}{x} + nN_0 - (N_0 - N_1) \cdot np^x . \tag{8}$$

Equation (8) is very difficult to be minimized to find the optimal block size x and N_1. We now adopt the following approximation. Assuming that $xp_0 \le 0.3$, we can approximate the probability of the level-1 block by omitting the higher-order terms of its Taylor series expansion, that is,

$$P_1 = p^x = (1 - p_0)^x = 1 - p_0 x + ... \approx 1 - p_0 x . \tag{9}$$

Given the measured probability p_0 for N_0 and equation (9), we can simplify equation (8) to

$$L_{ave} = \frac{n}{x} + nN_1 - (N_0 - N_1)nxp_0 . \tag{10}$$

Taking the derivative of equation (10) to x and setting it to zero, we yield

$$\frac{dL_{ave}}{dx} = -\frac{n}{x^2} + (N_0 - N_1)np_0 = 0 . \tag{11}$$

Solving for equation (11) gives the optimal block size as

$$x^* = 1/\sqrt{(N_0 - N_1)p_0} . \tag{12}$$

Taking the second derivative of equation (10) to x leads to

$$\frac{d^2 L_{ave}}{dx^2} = \frac{2n}{x^3} > 0 . \tag{13}$$

Equation (13) shows that we can obtain the minimum average coding length. By substituting equation (12) in equation (10), we obtain the minimum average length as

$$\left(L_{ave}\right)_{\min} = 2n\sqrt{(N_0 - N_1)p_0} + nN_1 . \tag{14}$$

Dividing the minimum average length by a total number of the data samples, the average bits per sample is therefore yielded as

$$\left(\frac{L_{ave}}{n}\right)_{\min} = 2\sqrt{(N_0 - N_1)p_0} + N_1 . \tag{15}$$

As an example, when encoding a data sequence with a Gaussian distribution with a length of $n = 2^{14} = 16384$ and a standard deviation of 2^{α}, where $\alpha = 7$, we found $N_0 = 11$, $N_1 = 9$, and the probability for the samples requiring more than N_1 bits to encode as $p_0 = 0.0401$. Using equation (12) leads to the optimal block size as $x^* = 1/\sqrt{2 \times 0.0401} \approx 4$ samples for the bi-level block coding algorithm, noticing that $xp_0 = 0.1604$ in this case. Applying equation (15) gives the average bits per sample as $L_{ave}/n = 9.53$ bits. It is observed that 1.47 bits per sample are saved. For a more sparsely distributed data sequence, we may expect smaller probability of p_0 and a larger difference of ($N_0 - N_1$) bits. Hence, more saving in terms of bits per sample is expected. Finally, the bi-level block coding scheme with an optimal block size is summarized below:

1. Find N_0 for the given data sequence.
   Initially, set $N_1 = N_0 - 2$ and $x = 4$.
2. For $N_1 = 1, 2, 3, N_0 - 1$.
   Estimate p_0, the probability of the sample requiring more than N_1 bits to encode;
   Calculate the optimal block size:
   $$x^* = 1/\sqrt{(N_0 - N_1)p_0}$$
   Round up the block size to an integer value.
   If $x^* \times p_0 \le 0.3$, calculate the average bits per sample
   $$\left(\frac{L_{ave}}{n}\right) = 2\sqrt{(N_0 - N_1)p_0} + N_1,$$
   and record N_1 and x^* vales for the next comparison.
   After completing search loops, select N_1 and x^* corresponding to the minimum average bits per sample, $\left(\frac{L_{ave}}{n}\right)_{\min}$.
3. Perform bi-level block coding using the obtained optimal N_1 and x^*.

Computer Simulations

To examine the performance of bi-level block coding, we generated data sequences using the Gaussian distribution with a length of 16,384 samples and various standard deviations. We compare each theoretical value of the average bits per sample with the experimental one, as well as compare them with the zero-th order entropy, which is the lower bound of lossless compression, defined as

$$H = -\sum_i p_i \log_2 p_i, \tag{16}$$

where p_i is the estimated probability of data samples in the sequence. Figure 5 shows the results. The top plot is the generated data sequence with the Gaussian distribution using a standard deviation of four, while the middle plot describes its distribution. Bi-level block coding uses the following optimal coding parameters: $N_0 = 5$ bits, $N_1 = 3$ bits, and block size as $x = 4$ samples. We achieve the theoretical average bits per sample as 3.61 bits, experimental average bits per sample as 3.60 bits, and entropy value as 3.07 bits. Hence, we can conclude that the theoretical value of the average bits per sample is very close to the one from the experiment. The coding scheme has 0.53 bits per sample above the lower bound (zero-th order entropy value). Therefore, we save 1.40 (5-3.60) bits per sample. The bottom plot in Figure 5 demonstrates that our results are consistent when compressing the data sequences with various standard deviations.

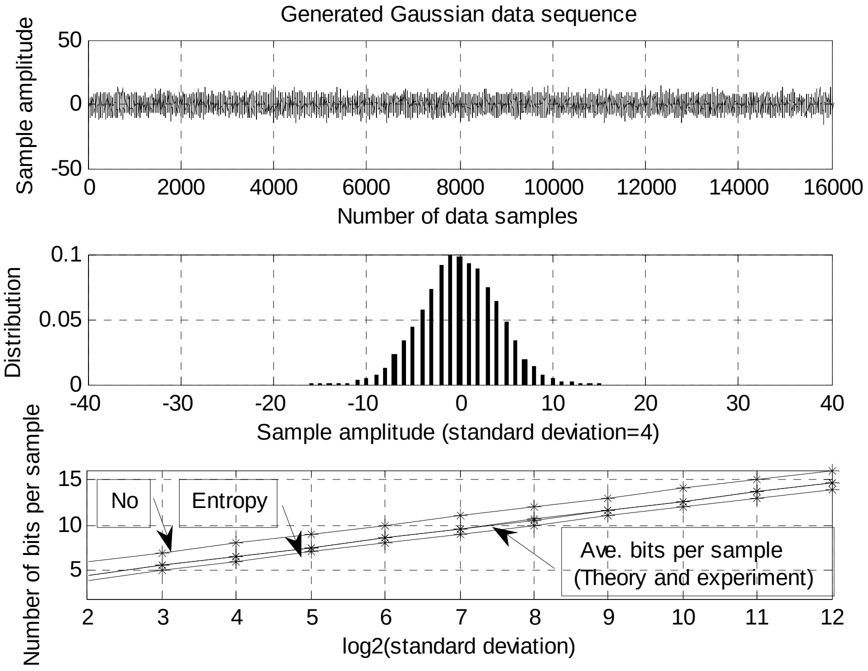

Figure 5: Bi-level Block Coding for Compressing Sequences with Gaussian Distribution

Since the bi-level block coding scheme is developed based on the data sequence with a sparse distribution, the use of the algorithm is not limited to the Gaussian sequence. As long as the percentage of data samples requiring more than N_1 bits to encode in the sequence is significantly small, applying the bi-level block coder would achieve its profitability.

Applications in the Two-stage Compression Scheme

In this section, we test bi-level block coding using a two-stage lossless compression scheme for compressing waveform data. Figure 6 shows the block diagram. For a simple illustrative purpose, the first stage of the scheme is chosen to be a linear predictor with an order of N , in which the linear predictor is designed using a traditional least-square design method [1, 7]. We use 16 bits to encode each linear predictor coefficient and each initial sample, respectively, and 4 bits for the linear predictor order. The bi-level block coder at the second stage requires 8 bits for storing each N_0 and N_1, respectively, 8 bits for block size, 1 bit for the block type indicator ("1" indicates the level-1 block, while "0" designates the level-0 block) for each block, and outputs all the residue bits. Finally, the packer packs the predictor and bi-level block coding information, which may be protected using an error control scheme to correct bit errors as a header, followed by the residue block bit steam. Hence, the measured average bits per sample (ABPS) is expressed as

$$ABPS = \frac{(2\times 16\times N + 28 + \text{bi-level block coding bits})}{\text{total number of samples}}. \qquad (17)$$

If the original data is represented by 16 bits each, the data compression ratio could be determined by

$$CR = \frac{16 \text{ bits}}{\text{ABPS}}. \qquad (18)$$

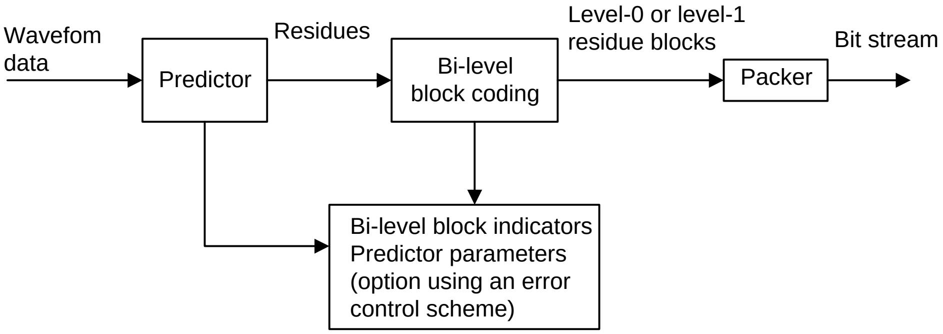
Wavefom data
Predictor
Residues
Bi-level block coding
Level-0 or level-1 residue blocks
Packer
Bit stream
Bi-level block indicators
Predictor parameters
(option using an error control scheme)

Figure 6: Two-stage Compression Scheme Using Bi-level Block Coding

Figure 7 shows the results of compressing audio signal. The audio is sampled at 44.1 kHz, and each audio sample is encoded using 16 bits. The two-stage compression scheme compresses audio samples frame by frame. We use a frame size as 1024 samples and a linear predictor order of 10. The final ABPS is obtained by averaging all the ABPS's from all the frames. The top plot in Figure 7 shows audio data samples, while the middle plot displays the predicted residues that have significantly reduced amplitudes and are de-correlated. The bottom plot depicts the distribution of the predicted residues from Frame 20. In this example, we achieve ABPS = 4.57 bits per sample and a compression ratio as CR =3.5.

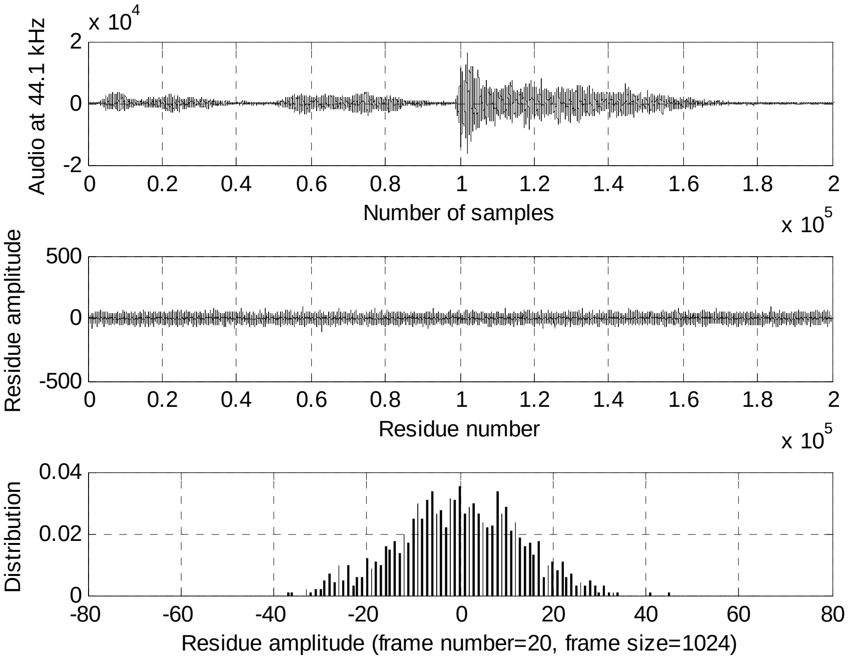

Figure 7: Lossless Compression of Audio Signal Using Bi-level Block Coding

Next, we examine the results of compressing of an ECG signal. As shown in Figure 8, the top plot of the ECG signal is sampled at 500 Hz, and each sample is encoded using 16 bits. The predicted residues are depicted in the middle plot, where the de-correlated residues with the reduced amplitudes are compressed using the bi-level coding algorithm. The bottom plot shows the distribution of the predicted residues from Frame 2. In this experiment, each frame consists of 1,000 samples. Compressing ECG samples frame by frame and using a linear predictor with an order of eight, we obtain the average bits per sample of ABPS =7.92 bits. A compression ratio of CR =2.02 is achieved.

Figure 9 shows the similar displays. The seismic data shown in the top plot is provided from USGS Albuquerque Seismological Laboratory by Professor S. D. Stearns. Each seismic data is presented using 32 bits. We use a linear predictor with an order of eight and a frame size of 835 in the two-stage compression scheme. The residues and residue distribution for Frame 4 are depicted in the middle plot and bottom plot, respectively. Finally, we obtain ABPS=9.80 bits per sample and CR=3.27.

The sample size, linear predictor order, frame size, ABPS, and compression ratio (CR) for each application are summarized in Table 1.

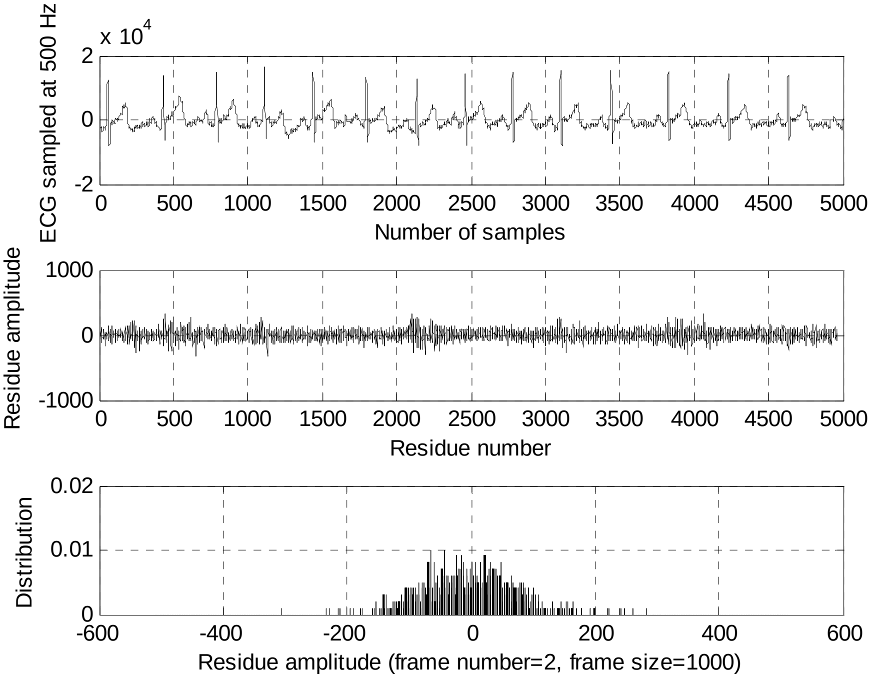

Figure 8: Lossless Compression of ECG Data Using Bi-level Block Coding

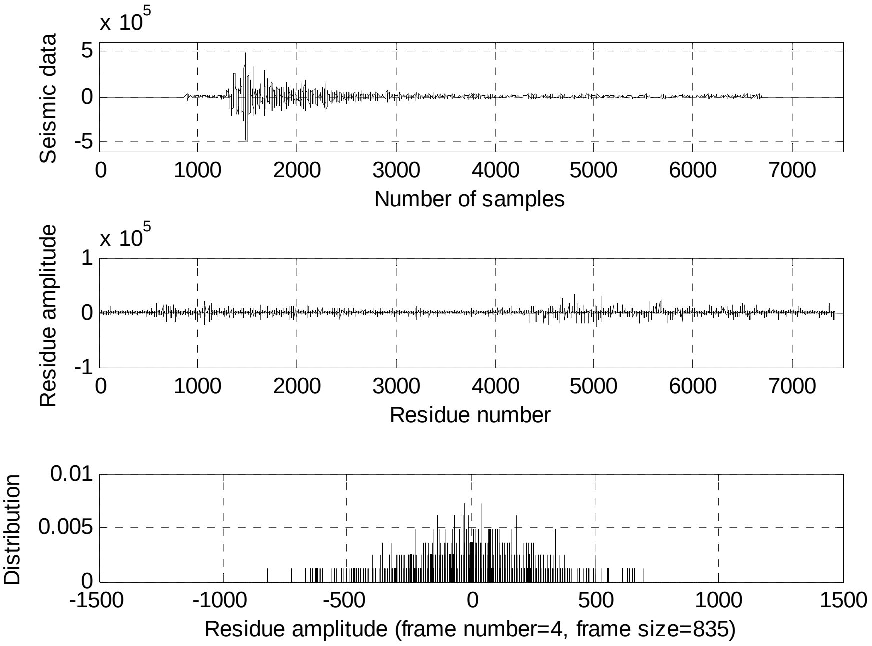

Figure 9: Lossless Compression of Seismic Data Using Bi-level Block Coding

Table 1: Performance Comparisons Using Bi-level Block Coding in the Two-stage Scheme

Data type	Sample size	LP order	Frame size	ABPS	CR
Audio	16 bits	10	1024 samples	4.58 bits	3.5
ECG data	16 bits	8	1000 samples	7.92 bits	2.02
Seismic	32 bits	8	835 samples	9.80 bits	3.27

Lossless compression in waveform data could be improved by choosing a more sophisticated predictor, such as the nonlinear predictor, neural network predictor, or others, as shown in references [4, 5, 6]. We are currently investigating lossless compression of waveform data using these predictors, along with bi-level block coding in a bit-error environment.

Conclusions

In this paper, a bi-level block coding scheme is developed, and its optimal coding parameters, such as the block size and the number of bits for encoding each data sample in the level-1 blocks, are obtained. The coding method is simple to apply and efficient for a sparsely distributed sequence, such as the predicted residue sequence from various prediction methods. Applications of bi-level block coding to audio, seismic, and ECG data are demonstrated using the two-stage compression scheme, where the first stage is linear prediction and the second stage is bi-level block coding. The bi-level block coding algorithm is also robust to bit errors if the coding parameters of the predictor and bi-level block coding are protected using the bit-error control scheme.

References

[1] S. D. Stearns, L. Tan, and N. Magotra, "Lossless Compression of Waveform Data for Efficient Transmission and Storage," *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 31, No. 3, pp. 645–654, May 1993.

[2] G. Zeng and N. Ahmed, "A Block Coding Technique for Encoding Sparse Binary Patterns," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. 37, No. 5, pp. 778–780, May 1989.

[3] S. D. Stearns, "Arithmetic Coding in Lossless Waveform Compression," *IEEE Transactions on Signal Processing*, Vol. 43, No. 8, pp. 1874–1879, 1995.

[4] R. Kannan and C. Eswaran, "Lossless Compression Schemes for ECG Signals Using Neural Network Predictors," *EURASIP Journal on Advances in Signal Processing*, (Special Issue on Advances in Electrocardiogram Signal Processing and Analysis) Vol. 2007, Article ID 35641, pp. 1–20.

[5] A. Koski, "Lossless ECG Coding," *Computer Methods and Programs in Biomedicine*, Vol. 52, No. 1 pp. 23–33, 1997.

[6] N.Sriraam and C.Eswaran, “Context Based Error Modeling for Lossless Compression of EEG Signals Using Neural Networks,” *Journal of Medical Systems*, Vol. 30, No.6, pp.439–448, December, 2006.

[7] S. D. Stearns, *Digital Signal Processing with Examples in MATLAB*. CRC Press, 2002.

Biography

LI TAN is currently with the Department of Electrical and Computer Engineering Technology at Purdue University North Central, Westville, Indiana. He received the M.S. and Ph.D. degrees in Electrical Engineering from the University of New Mexico in 1989 and 1992, respectively. He has taught analog and digital signal processing, as well as analog and digital communications for more than 10 years as a professor at DeVry University, Decatur, Georgia. Dr. Tan has also worked in the DSP and communication industry for many years.

Dr. Tan is a senior member of the Institute of Electronic and Electronic Engineers (IEEE). His principal technical areas include digital signal processing, adaptive signal processing, and digital communications. He has published a number of papers in these areas. He authored and co-authored two textbooks: *Digital Signal Processing: Fundamentals and Applications*, Academics Press/Elsevier, 2007; and *Fundamentals of Analog and Digital Signal Processing*, Second Edition, AuthorHouse, 2008.

JEAN JIANG is a professor of Electronic and Computer Engineering Technology at DeVry University, Decatur, Georgia. She received the Ph.D. degree in Electrical Engineering from the University of New Mexico in 1992. She has taught analog signal processing, digital signal processing, and control systems for many years.

Dr. Jiang is a member of the Institute of Electronic and Electronic Engineers (IEEE). Her principal technical areas are in digital signal processing, adaptive signal processing, and control systems. She has published a number of papers in these areas. She co-authored the textbook, *Fundamentals of Analog and Digital Signal Processing*, Second Edition, AuthorHouse, 2008.