A Novel and Comprehensive Compressive Sensing-based System for Data Compression

Ji Wu, Qilian Liang
Electrical engineering
University of Texas at Arlington
Arlington, TX 76013

Chiman Kwan
Signal Processing, Inc.
Rockville, MD 20850

Abstract
Data compression is one of challenging problems in data communication system due to the information explosion. In this paper, we propose a novel and comprehensive compressive sensing-based system for data compression. Performance of our proposed system is compared with conventional compression algorithm such as Huffman coding in terms of mean square error (MSE) after decompression, computation complexity and compression ratio, etc. As an application example, we implement this system to real world wind tunnel data. Simulation results show that our system can yield comparable or even better compression as Huffman coding in terms of information loss. The major drawback of Huffman coding is to calculate the probability of each symbol which means it is not be appropriate for real time coding due to large amount of calculation. Meanwhile, our proposed system can process data by multiplying original data with Gaussian or Bernoulli sensing matrix directly which is also easy to implement.

1 Introduction
Data compression is the process of encoding information using fewer bits than the original representation would use due to the limitation of bandwidth, power and storage, etc. It is used just about everywhere. All the images you get on the web are compressed, typically in the JPEG or GIF formats, most modems use compression, HDTV will be compressed using MPEG-2, and several file systems automatically compress files when stored, and the rest of us do it by hand. Fig. 1 shows a basic compression scheme. The raw data ($X$) is processed by an encoder and the result ($B$) is the compressed data, whose size is usually much smaller than original data size. To reconstruct the data, the compressed data is processed by a decoder. If the reconstructed data ($X'$) is different from the original data, the compression is lossy; otherwise, the compression is lossless. The compression ratio is defined by the ratio between compressed size and uncompressed size, i.e.

$$ r = \frac{\text{size}(B)}{\text{size}(X)} $$  \hspace{2cm} (1)

![Compression scheme](image)

**Figure 1: Compression scheme**

For lossy compression, there is a tradeoff between compression ratio and the result quality. The higher the quality is, the larger the ratio will be. A common quality measurement is peak signal to noise ratio (PSNR)

$$ PSNR = 20 \log_{10} \left( \frac{MAX}{RMSE} \right) $$  \hspace{2cm} (2)

where $MAX$ is the maximum possible value of the data and $RMSE$ is root mean squared error, which is given by

$$ RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (X'(i) - X(i))^2} $$  \hspace{2cm} (3)

One of the key ideas of compression is to first transform the data to a new domain and then only keep the large coefficients. Usually, after transformation, most of the coefficients are close to zeros. Therefore, the data in new domain are more suitable for compression (due to lots of zeros). For instance, JPEG-2000 uses wavelet transform. Other compression techniques include quantization, prediction, Huffman coding, and others. Different compression techniques can be combined together to achieve better performance. For instance, JPEG-2000 uses quantization to further compress wavelet coefficients.
All of the transformations share the same basic model:

\[ x = Dw \]  \hspace{1cm} (4)

where \( x \) is the signal to be approximated, \( D \) is the dictionary and \( w \) is the coefficients. There are lots of transformations such as FFT, DCT, Wavelet, etc. The difference between them is the dictionaries they use.

Dictionaries based on FFT, DCT and Wavelet are fixed and independent of data. These transformations do not require any prior knowledge of the data. For a specific domain which share similar characteristics, for instance, face images, it is possible to utilize the domain knowledge to achieve better compression performance. For instance, Principal component analysis (PCA) can be used to learn a dictionary based on the training data. The learned dictionary can be more suitable for the domain tasks than general dictionaries of FFT, DCT or Wavelet. Following the same scheme of PCA, we can apply sparse sensing to learn a dictionary and estimate the sparse coefficients for any input signal.

The remainder of this paper is presented as follows. In Section 2, we introduce experimental setup to collect real world data. Performance of Huffman coding using different quantization level in terms of mean square error, codeword length and compression is derived in Section 3. Background of compressive sensing and its performance with same real world data are presented in Section 4 and 5. Finally, the paper is concluded in Section 6.

2 Experimental Setup

The wind tunnel data we use here is from http://fris2.nist.gov/winddata/uwo-data/uwo-data.html. Wind pressure is measured at hundreds of taps (sensors) on a structure, as Fig. 2. At each tap, the pressure is sampled 500 times per second for about 100 seconds. There are 37 wind angles over the range between 1800 and 3600 at 50 increments. We use the data set ee1, ee2 from the website (http://fris2.nist.gov/winddata/uwo-data/ss20-test1/ee1-ee2.html). Based on a nominal full scale roof height wind speed of 84 mph (approximated Hurricane Andrew condition), the sampled data are equivalent to about 22 samples per second for 0.64 hours in full scale for the open exposure tests and equivalent to about 29 samples per second for 0.48 hours in full scale for the suburban exposure tests. A 650-tap building yields a 40 MB file when sampled at 500 Hz for 60 seconds or over 1.3 GB for a typical test with 36 wind directions. All of the samples were stored which means the amount of data is huge. Fig. 3 shows the first tap signal of data. The data looks like noise, which indicates that conventional compression scheme may not suitable for this data. Other novel and efficient data compression schemes should be implement to reduce the requirement of data. To analyze the time series data in later experiment, we cut each long signal into many short segments/blocks. Before describing the general frameworks of lossless and lossy compression approaches, let us consider some simple experiments. In this study, we use different methods to compress NIST wind tunnel dataset. Each dataset contains 49792 samples and we only retrieve 1000 samples from original data due to limitation of memory and computation capability of CPU.
3 Compression Using Huffman Coding

Huffman coding is a well known lossless compression algorithm. Quantization needs to be applied to the raw data before the coding process to increase the compression ratio. Therefore, information will be lost due to quantization errors. Although Huffman coding is simple, one major drawback of Huffman coding is to calculate the probability of each symbol which means it may not be appropriate for real-time coding due to large amount of calculation and considerable time required by computing.

First, we apply 8-level and 16-level uniform quantization to original data. Fig 4 depicts the waveform of 8-level and 16-level quantized data versus original ones.

![8 level Quantization](image)

![16 level Quantization](image)

Figure 4: Quantized data versus original one

After original data passing through the quantizer, each generated signal could be represented by 8 or 16 symbols. Huffman coding is also implemented by calculating the probability of appearance of each symbol. The average length and mean square error (MSE) of Huffman coding is shown in Table 1. If Huffman coding is not employed, the length required to represent original data should be 13000 bytes (For each symbol, 1 byte for “+” or “-” and 12 bytes to represent the amplitude range 0 to 4096.).

Since our goal is to keep quantization error as small as possible, we further explore the relationship between more quantization levels, mean square error and compression ratio. Fig. 5 show the waveform of 64-level, 128-level and 256-level quantized data compared with original ones. Mean square error (MSE), codeword length and compression ratio for each case are shown in Table 1 and 2. Non-uniform quantization using Lloyd algorithm is also considered in our investigation. Simulation results are given in Table 3 and 4. From those tables, we can conclude that performance of non-uniform quantization is much better than the uniform one. In order to achieve aggressive compression ratio, 256-level non-uniform quantization is adopted.

<table>
<thead>
<tr>
<th>Quantization level</th>
<th>8</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean square error</td>
<td>5.21%</td>
<td>3.54%</td>
</tr>
<tr>
<td>Codeword length</td>
<td>2112</td>
<td>3365</td>
</tr>
<tr>
<td>Compression ratio</td>
<td>32.41%</td>
<td>51.83%</td>
</tr>
</tbody>
</table>

Table 1: Performance of 8 and 16 level uniform quantization

<table>
<thead>
<tr>
<th>Quantization level</th>
<th>64</th>
<th>128</th>
<th>256</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean square error</td>
<td>0.17%</td>
<td>0.04187%</td>
<td>0.011217%</td>
</tr>
<tr>
<td>Codeword length</td>
<td>5602</td>
<td>7106</td>
<td>73100</td>
</tr>
<tr>
<td>Compression ratio</td>
<td>61.36%</td>
<td>51.18%</td>
<td>61.76%</td>
</tr>
</tbody>
</table>

Table 2: Performance of uniform quantization using more levels

<table>
<thead>
<tr>
<th>Quantization level</th>
<th>8</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean square error</td>
<td>1.85%</td>
<td>0.61%</td>
</tr>
<tr>
<td>Codeword length</td>
<td>3958</td>
<td>6334</td>
</tr>
<tr>
<td>Compression ratio</td>
<td>70.29%</td>
<td>55%</td>
</tr>
</tbody>
</table>

Table 3: Performance of 8 and 16 level non-uniform quantization

4 Compressive Sensing Background

Compressive sensing (CS) provides a framework for integrated sensing and compression of discrete-time signals that are sparse or compressible in a known basis or frame. Many natural signals have concise representations when expressed in the proper basis [9]. Mathematically speaking, consider a discrete signal \( f \in \mathbb{R}^N \) which can be expanded in an orthonormal basis \( \Psi = [\psi_1 \psi_2 \cdots \psi_N] \) as follows:
Figure 5: Quantized data versus original one using more level quantization

Table 4: Performance of non-uniform quantization using more levels

<table>
<thead>
<tr>
<th>Quantization level</th>
<th>64</th>
<th>128</th>
<th>256</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean square error</td>
<td>$3.6 \times 10^{-4}$</td>
<td>$4.1 \times 10^{-5}$</td>
<td>$3.0 \times 10^{-6}$</td>
</tr>
<tr>
<td>Codeword length</td>
<td>99226</td>
<td>15572</td>
<td>117259</td>
</tr>
<tr>
<td>Compression ratio</td>
<td>57.72%</td>
<td>39.69%</td>
<td>5.4%</td>
</tr>
</tbody>
</table>

$$f(t) = \sum_{i=1}^{N} x_i \psi_i(t),$$  \hspace{1cm} (5)

where $x$ is the coefficient sequence of $f$ that can be computed from signal $f$:

$$x_i = \langle f, \psi_i \rangle, i = 1, 2, \cdots, N.$$  \hspace{1cm} (6)

It will be convenient to express $f$ as $\Psi x$ (where $\Psi$ is the $n \times n$ matrix with $\psi_1, \psi_2, \cdots, \psi_n$ as columns). We can say the discrete signal $f$ is $K$-sparse in the domain $\Psi$, $K << N$, if only $K$ out of $N$ coefficients in the sequence $x$ are nonzero. Sparsity of signal is a fundamental principle used in the compressive sensing as well as in most modern lossy coders such as JPEG-2000 and many others, since a simple way for image compression would be to compute $x$ from $f$ and then only encode the values and locations of the largest $K$ coefficients. Examples in [7] show that perceptual loss is hardly noticeable from a megapixel image to its approximation obtained by throwing away 97.5% of the coefficients. Unfortunately, this compression process requires computing all $N$ coefficients of signal $f$ and the locations of the significant coefficients, which may not be known in advance.

Compressive sensing suggests this relevant information in the signal $f$ can be captured using a small number of nonadaptive (even random) measurements of the signal. It provides us a potential way to acquire the sparse data efficiently, or equivalently, highly accurate recovery of sparse data from undersampled measurements.

Note that there are only $K$ coefficients are nonzero, so we can remove this “sampling redundancy” by acquiring only $M$ samples of the signal $f$, where $K < M << N$. The new $M$-length observation vector $y$ can be represented as equation below:

$$y = \Phi f,$$  \hspace{1cm} (7)

where $\Phi$ is an $M \times N$ measurement matrix. The above equation can be written as

$$y = \Phi \Psi x = \Theta x,$$  \hspace{1cm} (8)
The signal $f$ can be perfectly recovered from $M$ equals to or a little bit more than measurements $K$, if $\Theta$ satisfies the so-called restricted isometry property (RIP)\[5\]. It suggests that $\Theta$ is sufficiently incoherent and $\Phi$ cannot sparsely represented basic vectors of matrix $\Psi$ and vice versa.

It has been shown in \[9\] that choosing an iid Gaussian random matrix as sensing matrix $\Phi$, $\Theta$ is also iid Gaussian for various orthonormal bases $\Psi$ such as spikes, sinusoids, wavelets, Gabor functions, curvelets, and so on. $\Theta$ is shown to have satisfied RIP with high probability, if $M \geq c K \log (N/K)$, where $c$ is a small constant and hence stable reconstruction is possible with high probability. Note that it is not known in advance which coefficients of $f$ are zeroes, or which samples are not needed.

With the new observation matrix $y$, we decide to recover the signal $f$ by $\ell_1$-norm minimization; the proposed reconstruction $f^*$ is given by $f^* = \Psi x^*$, where $x^*$ is the solution to the convex optimization program($\|x\|_{\ell_1} \equiv \sum_i |x_i|$)

$$\min_{x \in \mathbb{R}^N} \|\tilde{x}\|_{\ell_1} \quad \text{subject to} \quad y = \Phi \Psi \tilde{x}, \quad (10)$$

That is, among all the objects $\tilde{f} = \Psi \tilde{x}$ consistent with the data, we choose the one whose coefficient sequence has minimal $\ell_1$-norm. As is well known, minimizing $\ell_1$ subject to linear equality constraints can easily be reformulated as a linear program of $O(N^3)$ complexity. However, $\ell_1$-minimization is not the only way to recover sparse solutions; other methods, such as greedy algorithm \[4\], has also been proposed.

5 Compression Using DCT Dictionary based Compressive Sensing

We also developed a novel compressive sensing-based system to solve data reduction problem using the same data as in Huffman coding scheme. The first step of the algorithm is to choose the basis function of the original signal. After the process of data training, we will decompose the signal in terms of atoms from discrete cosine transform (DCT) dictionary. DCT is an example of a frequency dictionary. It converts data into sets of frequencies and compress the data by deleting the frequencies that are less meaningful. The dictionary elements are:

$$\frac{1}{\sqrt{n}} \cos \frac{2\pi ml}{n} \quad m, l = 0, 1, 2 \ldots n - 1 \quad \text{(the odd columns in A)}$$

and

$$\frac{1}{\sqrt{n}} \sin \frac{2\pi ml}{n} \quad m, l = 0, 1, 2 \ldots n - 1 \quad \text{(the even columns in A)},$$

where $n = 1000$ in our study.

After decomposing data in basis function, we sort the coefficient in a descending order and discard small ones which are less than 0.1. Fig. 6 shows the remaining coefficients. We can clearly see that less than 50 out of 1000 are kept and they are truly sparse in frequency domain.

Since there is noise when we receive the data, we would like to use the noisy version data $b'$ to estimate coefficient $x$ of original signal. We will use Basis Pursuit (BP) to do this. We alter the BP technique to solve above problem:

$$\min_{x \in \mathbb{R}^N} \frac{1}{2} \|b' - Ax\|_2 + \gamma \|x\|_1 \quad \text{subject to} \quad Ax = b', \quad (11)$$

Fig. 7 demonstrates all coefficients are perfect recovered, which means all information that original data contains is successfully retrieved. Fig 8 explains the role of probability that plays in the compressive sensing. The probability of perfect recovery decreases while we increasing compression ratio. If we want to guarantee the data is perfectly recovered, 200 samples are needed to represent the original data. Hence, the highest compression ratio we can achieve is 5.55 (Since the number of coefficients we kept for each data set is almost same, the number of samples that required by recovery algorithm is also the same.). Mean square error (MSE) for compressive is 3.75% which is on the same level of 16-level quantization. Since the data we used here is real world data which contains Gaussian noise, it is acceptable to have part of data distortion when we perform data recovery. It could also be treated as “signal denosing”.

Figure 6: Coefficients of original data

![Figure 6: Coefficients of original data](image-url)
6 Summary

We employ both Huffman coding and compressive sensing to compress the same data. One advantage of Huffman coding is simple to implement. It could also apply to all kinds of data. The major drawback of Huffman coding is to calculate the probability of each symbol which means it may not be appropriate for real-time coding due to large amount of calculation. Huffman coding is a lossless compression algorithm. However, it requires quantization of original data which will introduce information loss. One challenge of compressive sensing is to find the proper basis to decompose signal. Based on our study, compressive sensing doesn’t require any preprocessing or information about original data in advance. As we see, a careful design of CS algorithm can yield comparable or even better compression as Huffman coding algorithm in terms of information loss. One great advantage of compressive sensing is that it can process data by multiplying original data with Gaussian or Bernoulli sensing matrix which is also easy to implement.

Acknowledgements

This work was supported by Air Force Office of Scientific Research (AFOSR) under Grant FA9550-11-C-0062.

References


