Compression and Compressed Sensing in Bandwidth Constrained Sensor Networks

Sergey Negrashov

April 6, 2017

Contents

1 Introduction

Improvements in sensor and radio technologies allow for creation of cheap sensors interconnected via radio links and the Internet. These advancements opened the door for creation of large area autonomous monitoring networks referred to as sensor networks. Sensor networks consist of at least one sensor node and any number of processing nodes[4]. Sensor nodes are equipped with embedded processors, sensors and communication mechanisms. They are responsible for remote sensing and in some cases data transport[14]. Processing nodes lack sensors, but can be used to aggregate, process or route messages to their destination. Arguably the first sensor networks were developed by the US military in the 1970s [4]. Now sensor networks are finding uses in medicine [12], power generation[19], heavy industry[15], and many other fields.

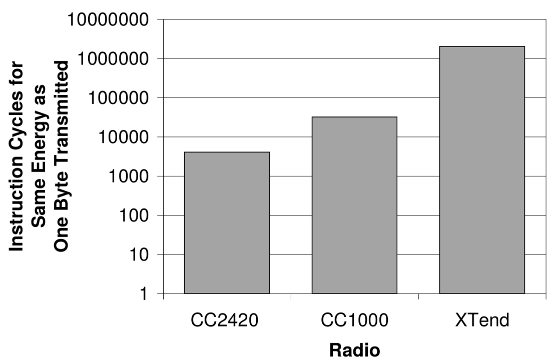

Figure 1: CPU cycles of of an MSP430 based MCU requiring the same amount of energy as transmitting a single byte over a particular radio[28].

Bandwidth requirements of a wireless sensor network has a direct effect on its performance. This is especially true in battery powered devices, where the power budget is commonly dominated by the radio module[28]. This is further illustrated in Figure 1. An embedded microprocessor may be able to execute hundreds of thousands of instructions in the same power budget as sending a single byte over the embedded radio. Furthermore, minimizing the bandwidth required to send a measurement, may in turn improve the latency of the sensor network. Since a lot of sensor network radios are low bandwidth, shorter messages propagate faster, even with the overhead of local message processing. Finally, with the finite bandwidth available to a sensor network, reducing the bandwidth requirements means that more sensors could be deployed, or a lower power radio may be utilized. There are three bandwidth reduction methods described in this paper.

- Lossless Compression
- Lossy Compression
- Compressed Sensing

Lossless compression reversibly reduces data size proportionally to its entropy. It can be very powerful in cases where expected signal characteristics are known a priori[15][22]. Lossy Compression reduces the data size by irreversibly changing the sampled waveform to a new form[35][1]. Compressed sensing combines both the sampling and compression into one step by crafting the data sampling proportional to the information content of the signal, as opposed to the frequency content[5][20].

1.1 Sensor network topologies

Sensor network topologies can be described as a directed graph with sensor nodes as source nodes and process nodes as inner nodes and sinks. Sensor network topologies may change over time as sensor nodes move, go-offline or are added to the network [4]. Common sensor network topologies used in industry are shown in figure 2.

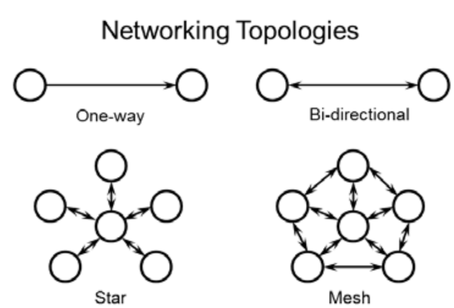

Figure 2: Common Network Topologies[17].

One way and bi-directional topologies are the simplest types of network, consisting of a single sensor and a single processing node. In the one way network topology the sensor node sends measurements at specific intervals. In the the bi-directional case the processing node is able to request data when it's required and change sampling parameters of the sensor node[17].

Star topology can be represented with a directed acyclic graph. This topology is well suited for sensor networks deployed in locations with existing local area network infrastructure. In this setting, processing nodes serving as gateways can aggregate data from local sensor nodes and pass it upstream via a local area network, or Internet[13]. Over the years this topology has become known as the Internet of Things (IOT). In the mesh topology each sensor node also performs a role of a router, passing messages between its neighbors. Industry standard mesh network protocols such as IEEE 802.15.4 based ZigBee provide peer discovery and message routing[17]. Mesh topology allows for self organization and redundancy. Furthermore mesh sensor networks do not require existing infrastructure to function. Mesh networks may have one or more sink nodes which function as aggregates for the sensor data[6]. These sink nodes may be connected to the Internet in order to transmit data to the acquisition server.

Some situations require more complicated network topologies. This is especially true in sensor networks tasked with event detection as opposed to general monitoring. In this case the sensor network must not only record sensor data, but also process it in search of anomalies. This is generally referred to as in-network processing[22]. In one of the fundamental sensor network papers "Distributed Detection With Multiple Sensors" R. Viswanathan and P Varshney described a multitude of sensor topologies useful for remote event detection[34]. The authors also authors developed a mathematical model for distributed detection efficiency based on Bayesian and Neyman-Pearson statistical models.

1.2 Spectral Analysis

Spectral Analysis is a technique ubiquitous in many disciplines. The applied view of spectral analysis extends matrix theory machinery, such as eigenvalue and eigenvectors, to discrete finite time series data. It should be noted that spectral analysis extends to many spaces beyond mathematical functions. For example the Probabilistically Checkable Proofs (PCP) theorem, which states that any proof regardless of the length or context can be verified to a 99% confidence in a fixed number of random checks, was proven via spectral analysis [7]. Spectral analysis as whole is beyond the scope of this paper, however there are two techniques which are important to understanding the compression techniques presented in the rest of this document: Fourier transform and Wavelet transform. Furthermore, since modern sensor networks are digital systems only the discrete versions of these transforms are considered, namely Discrete Time Fourier Transform(DTFT) and Discrete

Wavelet Transform(DWT). Formaly DTFT is defined as:

$$F_n = \sum_{k=0}^{N-1} x_k \cdot e^{-2\pi ikn/N} \tag{1}$$

Where $x_0...x_{N-1}$ are real valued samples, sampled at fixed time offsets, and $F_0..F_{N-1}$ are the Fourier coefficients. A useful interpretation of equation 1 is that $F(n)$ is resultant of an inner product of the the vector x with a complex sinusoid $e^{-2\pi ikn/N}$. This is useful because the set of all sinusoids of form $e^{-2\pi ikn/N}$ with $n \in 0..N-1$ are orthogonal to each other and form a complete basis for any vector x. As such, coefficient F_n represents the phase and amplitude of the n'th sinusoid in the basis set. Inverting the DFT operation is as simple as accumulating the contributions of each sinusoid in the basis set and normalizing the result. Another way to describe DFT is via matrix operations:

$$F = J \cdot x, \quad J_{jk} = e^{2\pi ijk/N} \tag{2}$$

The matrix J is an $n \times n$ which represents DFT operation in matrix form. This interpretation makes it easy to see that each row of matrix J is a bandpass filter for frequency $e^{2\pi jn/N}$, and is used extensively in compressed sensing theory. From equation 1 DFT can not measure frequencies beyond $e^{-2\pi i}$ or $\frac{1}{2}$ of the sampling rate of x. This is effect is known as aliasing, and formally known as the Nyquist limit. Frequencies $f > \frac{1}{2}f_{sampling}$ will contribute to the $\frac{f}{2}$ sinusoid basis instead. For this reason most digital data acquisition systems will include an analog filter fronted, which will filter all frequencies higher then $\frac{f_{sampling}}{2}$

Apart from the Nyquist limit DFT is limited by what is known as the uncertainty principle. The uncertainty principle states that no signal can be well localized in both frequency and time domain[23]. We can observe this in the extreme cases where the DFT of the Dirac delta function is equally distributed across all of the DFT coefficients, while a single coefficient spike in the DFT domain transforms into a sinusoid in time. This feature makes DFT a very powerful tool in compression, since a lot of signals that appear random in time domain will be sparse in the frequency domain.

While DFT is an effective strategy for analyzing periodic signals, it is not as useful for analysis of transient signals. Indeed, due to the uncertainty principle,a signal which is well localized in time will have a large spread in the frequency domain. DWT overcomes this limitation by switching the basis to a well localized functions called wavelets. A family of wavelets splits the wavelet domain into orthonormal components just as the sinusoids split frequency space in the DFT. Wavelets act as a filter, successively filtering out the low frequency components from the time series via the inner product. Because wavelets are orthonormal and split the space perfectly each filter operation is followed by a decimation, where $\frac{1}{2}$ of the resultant coefficients are discarded. Thus, for a vector of length n the first filter will produce $\frac{n}{2}$ low pass filtered wavelet coefficients and $\frac{n}{2}$ approximation coefficients. The next iteration will operate on the previous stage approximation coefficients and yield $\frac{n}{4}$ wavelet coefficients and $\frac{n}{4}$ approximation coefficients. Just as with DFT, DWT is easily reversible via accumulation of the inner product of wavelets and their coefficients followed by normalization. With the correct choice of a wavelet many common signals can be transformed via DWT into a sparse representation. Consider the power consumption of a household smart meter shown in Figure 3, along with its wavelet transform[32]. While the raw data looks complex and incoherent, a wavelet transform concentrates the signal energy in only a few coefficients, thus yielding a very sparse representation. This makes wavelets a popular tool for design of compression algorithms which operate on transient signals[35].

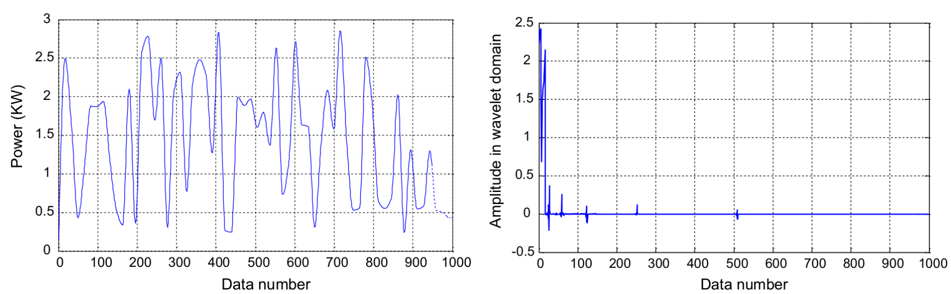

Figure 3: Left: Power consumption of a house as reported by a smart meter. Right: same data transformed into a Daubechies wavelet domain[32].

2 Lossless Compression

The goal of lossless compression is to create a smaller representation of the data. However that is not always possible. It is trivial to show that no lossless compression algorithm is capable of compressing every binary sting to a shorter form. A compression algorithm must be bijective, with every input string mapping to a unique output. By the pidginhole principle there are simply not enough strings shorter then N symbols to encode all strings up to length N. Lossless compression in network transport is often viewed as a straight forward trade between the processing power and bandwidth requirement. This is not necessarily the case with sensor networks, since a sensor node microprocessor often lacks math hardware functionality which is heavily used in general purpose lossless compression algorithms, forcing expensive software emulation.

2.1 F. Marcelloni, M. Vecchio

Sensor network designers can exploit prior knowledge about their measurement in order to design Ad-Hoc lossless compression algorithms to suit their needs. For example waveforms representing environmental factors such as temperature and humidity are generally smooth and differentiable. Additionally, environmental sensor networks will sample at much higher sampling rate then the bandwidth of the signal they are measuring. F. Marcelloni and M. Vecchio developed an algorithm that exploits the fact that the difference between the two consecutive measurements in an environmental monitoring system tend to have low local variance[22]. Their algorithms, computes the difference between consecutive ADC samples and encodes them using Huffman coding. Their simulations showed a compression ratio of 66% when applied to temperate and humidity sensor data.

2.2 F. Marcelloni, M. Vecchio

Another example of Ad-Hoc compression algorithm comes from industrial monitoring. In industrial settings, sensor networks are employed to detect failing equipment before a complete breakdown. Failed bearings are one of the most common failure modes associated with any rotating equipment. Furthermore, worn bearings have a distinctive acoustic and vibration signature, which makes them fairly easy to monitor in-situ. Unfortunately due to a high angular velocity of common industrial equipment such as CNC lathes and mills, the desired signal often falls in the 1-5kHz range. According to the Nyquist theorem sampling rate required to reconstruct this signal often falls into 10kSps range. Additionally, different failure modes result in different signal powers at varying frequencies which does not favor lossy compression or filtering. In order to bring down the bandwidth requirements Q. Huang and B. Tang developed a lossless compression method which allows for up to 85% compression ratio in bearing monitoring networks[15]. This algorithm combines the work done by F. Marcelloni with the traditional Discrete Cosine Transform based compression methods. Similar to DFT, DCT transforms a time domain signal into a frequency domain. However DCT only uses cosines and thus yields only real coefficients. Waveform is transformed via DCT, and the high energy frequency bands are encoded without compression. Low energy bands are encoded using the methodology described in [22] where each consecutive phase measurement is subtracted from the previous one, and then Huffman encoded. This method did not only result in high data compression the during author's tests, but also lowered the power consumption on the sensor node by 23% by lowering the duty cycle of the radio transmitter.

2.3 S. Kalaivani, C. Tharini

Computationally simple compression techniques are sometimes desirable for ultra-low power consumption devices. One such algorithm is presented by S. Kalaivani and C. Tharini[16]. Their work focuses on Rice-Golomb coding. While generaly not regarded as a compression algorithm, Rice-Golomb encoding can nontheless reduce the amount of bandwidth required to transmit a waveform. In their work they extended Rice-Golomb coding to support negative numbers, by creating a static offset equal to the absolute value of the most negative sample of the dataset, thus turning it into a positive valued dataset. This offset is encoded along with the data, thus allowing the decoder to extract the original waveform. The advantage of the author's method is in its ability to perform both lossy and lossless compression while maintaining compression ratio of up to 64% for lossless encoding and 50% for lossy encoding. The algorithm is simple enough to run on devices with limited mathematical primitives, and low memory.

2.4 S-LZW

Some general purpose algorithms can be adapted to the sensor network domain. The LZW algorithm is attractive, because it does not require complex mathematical primitives to operate. It has a small network overhead, since the dictionary is built from the bit stream and maintained independently by both sender and receiver and finally its computational overhead and power budget is fairly small. C. Sadler and M. Martonosi modified the LZW algorithm to operate on blocks of data, encoding each block separately[28]. This allowed the algorithm to operate over lossy channel, unlike a standard LZW implementation. Authors showed that the power requirements for running a modified S-LZW algorithm were significantly smaller compared to transmitting uncompressed data. As such for battery powered devices, on board lossless compression can improve battery life and reduce radio duty cycle.

Furthermore on multihop, and/or low bandwidth links, compressed stream resulted in lower latency, since it took much longer for the uncompressed message to propagate over the slow physical link.

S-LZW algorithm is also well suited for hardware acceleration. C. Lin and W. Wang developed a hardware based LZW compression engine and demonstrated it in a production microchip[18]. What makes their approach unique is the ability of their compression engine to predict the power consumption required to compress a block of data. This, along with close cooperation with the embedded processor, allowed the system to decide if the compression is energy beneficial for a data block. The authors show that their approach could yield significant energy savings if their methodology is combined with an accurate power consumption model for the radio transmitter.

2.5 Comparison of Lossless Compression Algorithms

Table 1 compares the compression algorithms discussed above. It is difficult to gauge the performance of these compression algorithms, since apart from S-LZW all of these algorithms operate in specific sensor network domains. For example Q. Huang, B. Tang algorithms exploits periodicity in the sensors measurements, and would perform poorly with environmental datasets. Instead we rank the algorithms based on their computational overhead.

Table 1: Comparison of lossless sensor network compression algorithms.

Algorithm	Type	Overhead	HW accelerated
F. Marcelloni M. Vecchio	Huffman	Low	No
Q. Huang B. Tang	DCT/Huffman	High	No
S. Kalaivani C. Tharini	Rice-Golomb	Low	No
S-LZW	Dictionary	Low	Yes

3 Lossy Compression

Lossy compression results in information loss. This implies that it is impossible to recover the original data set for the compressed data. It should be noted that lossy compression may in fact result in no useful information loss. In many cases the sensor's precision is higher then the sensor accuracy.

In these situations it is possible to compress the data while introducing an error without compromising the measurement, as long as this error is smaller then the accuracy of the sensor. For example T. Schoellhammer and B. Greenstein developed a lossy compression algorithm which which exploits this property, in order to compress a data stream without useful information loss. [31]

As with all bandwidth constraint methodologies lossy compression relies on the data sparcity. Generic lossy compression consists of three stages:[36]

1. Transformation: In this stage a sampled waveform is transformed into a desired domain. Common domains include a wavelet representation via a wavelet transform,[35] and a frequency domain, via a Fourier or Cosine transform. These domains are used because they exhibit a property called energy compaction, [Rao and Yip 1990] which means that most of the signal energy is concentrated in a few select coefficients.

2. Adaptive modeling: In this step the algorithm selects a subset of coefficients computed from the transformation. Adaptive modeling can be tailored to the specific accuracy requirement by varying the numbers of the candidate coefficient. Furthermore it may be tuned to the bandwidth requirement by fixing the maximum number of candidate coefficients. After the information selection a quantizer may be used to further reduce the number of symbols in the candidate coefficients.

3. Entropy coding: A lossless compression stage further reduces the data size, by compressing the coefficients acquired from adaptive modeling.

3.1 K-RLE

If the signal is already sparse in the domain it's sampled in, it is possible to compress it in situ, without performing the transformation step. In such cases the adaptive modeling is performed directly on the sampled data. K-RLE is one of the simplest sensor network data compression algorithms[24]. Given an error bound K, K-RLE will compare the current sample with temporally consecutive waveform samples. All consecutive samples which differ from the current sample by less than K are encoded with a current sample and number of occurrences. This algorithm is fast, memory efficient, and provides an upper bound for the error rate. Using an environmental monitoring data authors demonstrated up to 99.56% compression ratio with sufficiently high values of K. It should be noted that simple run length encoding (K=0) was able to achieve 61.6% compression ratio with this dataset.

3.2 LTC

T. Schoellhammer and B. Greenstein developed an algorithm LTC which is a logical evolution of K-RLE[31]. Adaptive modeling and entropy coding steps are essentially the same, but while K-RLE uses an identity function as it's transformation step, LTC algorithm uses the derivative operator. Thus by transforming the signal into tangent space, uniformly monotonically changing signals can be efficiently encoded. This algorithm uses linear interpolation in order to replace as many data points as possible with line segments. It starts with the first data point, leaving it unchanged. The second data point is used to generate a line between the two points. Following points are added to the line segment, and the coefficients for the line are reevaluated. If the error is within a bound ϵ, multiple points are encoded as a single line segment, otherwise a new line segment is started. In the best case a whole uniformly sampled waveform can be encoded as a starting point and a intercept and slope of a line. Authors demonstrated compression ration of upwards of 95% in environmental datasets.

3.3 Wisden

A wavelet transform is common as a transformation step in lossy compression. This is because a lot of signals are sparse in the wavelet domain. Traditional wavelet transforms are computationally expensive requiring a large number of floating point operations. However certain wavelet families such as Cohen-Daubechies-Feauvea(CDF) rely only on integer addition and bit-shifting for the transform, which makes them an ideal candidate for low power sensor nodes. N. Xu and S Rangwala developed a compression low power wavelet compression algorithm called Wisden based on the CDF transform[35]. After the wavelet transform Wisden thresholds the data in order to convert all of the coefficients smaller than the parameter ϵ to 0. Remaining coefficients are normalized and quantized to an N bit representation. Late normalization allows for a reduction of floating point operations, because the majority of wavelet coefficients are expected to be close to 0 anyways, and will not contribute much during the normalization process. Finally, remaining coefficients are quantized to N bits and run length encoded. Using simulated vibration data authors showed that the Wisden compression algorithm could deliver 95% compression ratio with 1bit RMS error when using 4 bit quantization.It should be noted that the datasets compressed in this study has a significantly higher information and frequency density then the environmental datasets used in testing of K-RLE and LTC.

While authors of Wisden did not provide a direct comparison to K-RLE and LTC, one would expect that it will perform significantly better then it's real space counterparts. An example of a CDF wavelet transform and its quantized representation is shown in Figure 4.

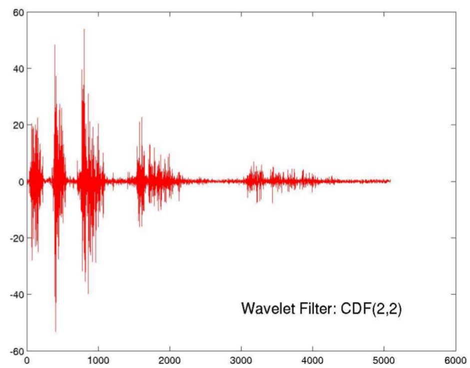

Figure 4: Comparison of the original and un-compressed wavelet transform using Wisden[35].

3.4 M. Alsheikh, P. Pih

Interestingly the transformation used in the first stage of the lossy compression does not have to be well defined. Instead an artificial intelligence approach can be used for both the transformation step and decoding step. This is especially true if the a large dataset of historical data is available. M. Alsheikh and P. Pih demonstrated that a neural network can be used for lossy data compression and decompression in sensor networks[1]. It should be noted that their approach while very computationally heavy during NN training is quite robust and lightweight during execution, as it is implemented using only linear and sigmoidal operations. Their approach trains the encoder and decoder NN in parallel constrained with the absolute error between the encoder output and the decoder input. An additional constraint is applied to minimize the L0 norm in the output of the encoder. This by minimizing the error and bandwidth, a NN can be trained to compress and decompress a waveform with minimal artifacts and bandwidth requirements. Their approach outperformed frequency domain lossy compression methods by an order of magnitude, while maintaining a similar average error rate of 0.1%. However, algorithms ability to deal with anomalous data is not discussed by the authors, and remains unclear.

3.5 Comparison of Lossy Compression Algorithms

Comparison of lossy algorithms discussed in this section is shown in Table 2. As with the lossless counterparts most of these algorithms use different datasets in ther evaluation, and as such direct comparison of their performance is impossible. Instead we compare them by the domain, adaptive modeling strategies and lossless compression steps. Furthermore we compare the computational overhead of the algorithms discussed in this section.

Table 2: Comparison of lossy sensor network compression algorithms.

Algorithm	Domain	Adaptive modeling	Compression	Overhead
K-RLE	Time	Fixed error	Run Length	Low
LTC	Tangent	Fixed error	Run Length	Low
Wisden	DCT	Thresholding/ Quantization	None	High
M. Alsheikh P. Pih	N/A	Neural Net	none	Medium

4 Compressed Sensing

From the perspective of information theory, methods described in the first two sections may seem very crude. Indeed both lossy and lossless compression methods described above rely on uniform time domain sampling. Such sampling is inherently limited by the Nyquist theorem, which states that in order to capture a signal with a frequency content of f without aliasing it must be sampled at a rate of $2f$. If nothing is known a priori about the signal measured apart from its maximum frequency content, the Nyquist theorem provides machinery to make a measurement. However, most signals are compressible, that is they are not simply white noise. Surely the more we know about the signal's information content, the fewer samples we need to collect in order to reconstruct it. Compressed Sensing(CS) theory uses this fact to reduce the bandwidth required to make a measurement by taking far fewer samples.

In order to formally define CS, we must first discuss the question of sparsity and norms. A vector is considered k sparse in some basis Ψ if it can be represented as $\sum_{i=0}^{k} a_i \Psi_i$, where Ψ_i is the i'th column vector of Ψ in matrix form. Let's define a k-sparse signal in basis Ψ as a vector of scalar values over the period of time $z_k = [z_1, z_2, z_3 ... z_n]$. Since z is sparse in the domain Ψ, the L_0 norm(number of samples which are not 0) of x where $z = \Psi x$ is much smaller then n. In CS our goal is to recover z_k from a series of measurements $y = [y_1, y_2, y_3 ... y_m]$ such that $m << n$, and $m < k$. Measurements y are a linear combination of z as defined by an $m \times n$ measurement matrix Φ:

$$y = \Phi z = \Phi \Psi x \tag{3}$$

Equation 3 is under-determined since $m << n$. While there are infinitely many solutions for x that satisfy 3. However through careful selection of Φ and Ψ, and constraining x to being sparse, it is possible to recover the original signal[2]. Mathematically this could be accomplished by finding a vector x such that:

$$x = arg\ min ||\widetilde{x}||_0 \quad y = \Phi \Psi x \tag{4}$$

Where $\widetilde{x}$ is the set of all possible solutions to $y = \Phi \Psi x$. By picking the solution which minimizes the L_0 norm of x we reconstruct the signal of interest. Unfortunately finding such a vector is NP-Hard, and computationally infeasible[2]. On the other hand L_1 norm($||x||_1 = \sum x_i$) minimization techniques are ubiquitous in literature, and have been shown to be equivalent to the L_0 norm minimization in most cases[26]. Thus we arrive at the mathematical definition of CS:[2]

$$x = arg\ min ||\widetilde{x}||_1 \quad y = \Phi \Psi x \tag{5}$$

There are three criteria required for correct reconstruction, which govern the selection of Φ, Ψ, and number of samples m [3][2].

1. Ψ must be selected such that x is sparse.

2. $m > C \cdot k \cdot log(n)$ where C is some small constant.

3. Measurements y must be an incoherent combination of Ψx.

While it is often impossible to know exactly which basis will lead to the sparsest representation, Wavelet and Fourier transforms are the sparse basis of many common signals. Second criteria sets a bound on the number of samples required for reconstruction with respect to sparsity of x. Incoherent property requires that each measurement y is an incoherent combination of all values of the reconstructed vector Ψx.In order to guarantee a stable recovery of x matrix Φ has to satisfy an Restricted Isometric Constraint(RIP)[33].

An $n \times m$ matrix Φ satisfies RIP if and only if any $n \times m$ submatrix Φ' $(n < m)$ satisfies:

$$(1-\delta)\frac{n}{m}||y||_2^2 \leq ||\Phi' y||_2^2 \leq (1-\delta)\frac{n}{m}||y||_2^2 \tag{6}$$

Where y is any k-sparse vector with $k < s$, and $\delta \in (0,1)$ and $||y||_2^2$ denotes the square of the L_2 norm($||y||_2 = \sum y_i^2$) of vector y. In layman terms, equation 6 implies is that as one removes columns from the matrix Φ it will remain a good basis for any sparse vector k. Unfortunately, finding a matrix that satisfies equation 7 is strongly NP-hard[33]. Luckily many random matrices such as Random Partial Fourier, Gaussian and Bernoulli matrices have been shown to satisfy equation 6 with high probability.

4.1 L_1 minimization

Most CS L_1 norm reconstruction algorithms fall into two categories.

- Linear programing/Convex relaxing algorithm.
- Greedy pursuit algorithms.

Linear programing algorithms such as basis pursuit attempt to solve equation 7 using convex optimization[11].

$$x = arg\ min||\widetilde{x}||_1, \quad ||y - \Phi\Psi x||_2^2 \leq \epsilon \tag{7}$$

These algorithms are iterative, and have the advantage of the adjustable precision parameter ϵ. If ϵ is tuned correctly with respect to the error of the measured signal, it is possible to reconstruct that signal without information loss.

Greedy pursuit algorithms, such as matched pursuit, work by decomposing the reconstructed signal into a linear expansion of projections over an over-complete dictionary of waveforms D[21]. D contains a large number of basis function for the space occupied by x, with a lot of redundancy. Thus by iteratively selecting the projection of x on D which minimizes the error one can very quickly converge to an approximation of x:

$$x \approx x_n = \sum_{n=1}^{N} a_n g_n, \quad g_n \in D, \quad y = \Phi\Psi x \tag{8}$$

Where a_n is the weighted coefficient of x_n projected on D. Intuitively as n approaches the cardinality of the D, $a_n \to 0$, however not all elements in D are required and a suitable approximation of x can be recovered in order k steps where k is the sparsity of x. Greedy CS algorithms tend to converge faster then their linear programing counterparts, while providing a less accurate reconstruction.

4.2 Analog and Digital Compressed Sensing

The first academic work published on the subject of CS was written in 2006[8]. Since CS is a relatively young field the publications describing a full CS sensor network are sparse. There is however, a large body of work advocating for such systems and presenting working components of CS based sensor networks. There are several ways that compressed sensing techniques can be applied to sensor networks. Compressed sensing can be applied at the node level, in place of lossy compression[20]. It is quite common for sensor networks to monitor a global state of the environment, where each measurement is correlated with neighbors. In this case it may be possible to collect the measurements from the subset of the network and use compressed sensing to reconstruct the entire state.

Performing CS at the node level has several advantages. While in lossy and lossless compression the sampling is still carried out according to the Nyquist theorem, in node level CS, the ADC sampling rate is significantly slower. Furthermore, once measured the data is already compressed, thus it requires no extra CPU cycles to process. This makes CS particularly well suited for low power devices. An example node level CS implementation is shown in figure 5.

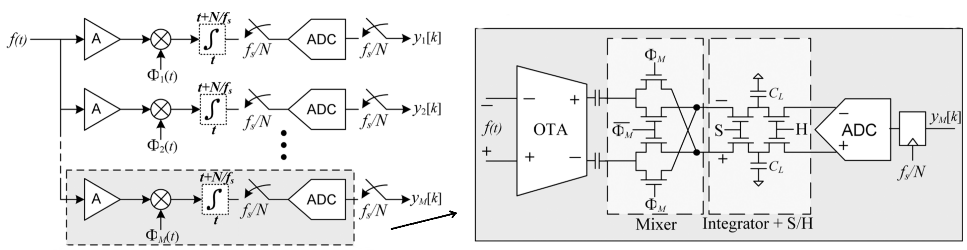

Figure 5: Common analog node level CS architecture[5].

In this topology the measurement matrix Φ is applied via an analog mixer and integrator. In order to make the measurement consistent all N measurements must be performed at the same time. Such that measurement would require N ADCs or at the very least a fast N channel ADC. Luckily it easy to parallelize ADC conversion in certain ADC topologies such as a

Ramp-Compare where each additional ADC may be implemented with an additional comparator and counter register [5].

Another way to perform node level CS is in the digital domain. In this case the single ADC still abides the Nyquist theorem, and specialized digital hardware applies Φ in the digital domain. A diagram of such architecture is shown in figure 6.

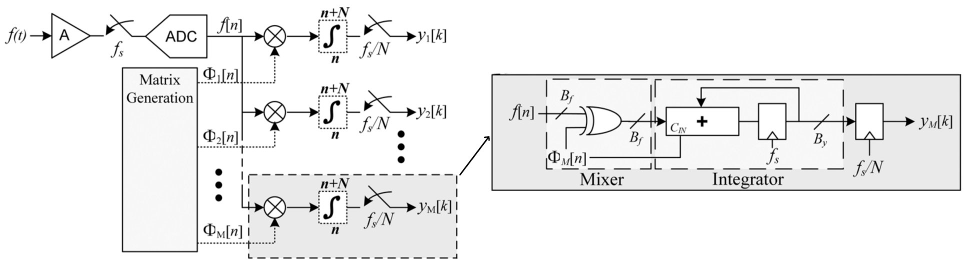

Figure 6: Common digital node level CS architecture[5].

In this case a digital mixer and integrator take the place of their analog counter parts. It has been shown that the theoretical power consumption of the analog CS systems is significantly higher then digital CS. The main contributor to this discrepancy is the transconductance amplifier used in the analog CS. This is illustrated in Figure 7

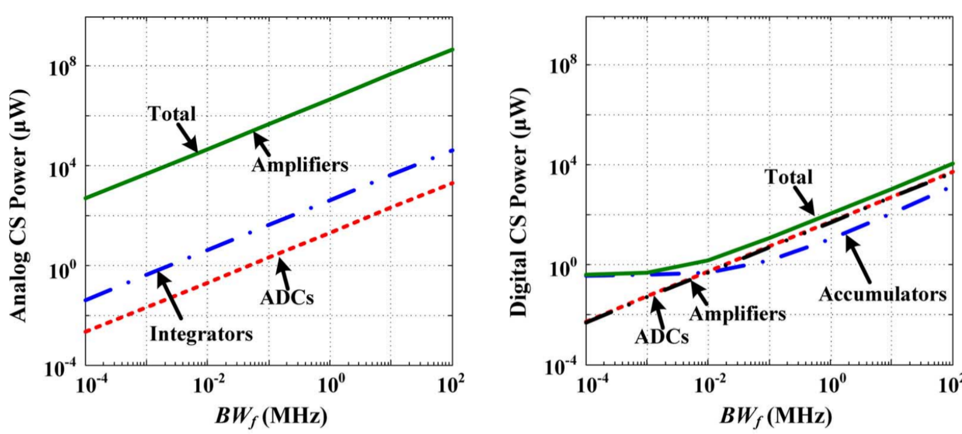

Figure 7: Theoretical minimum power consumption of node level CS for digital and analog methodologies[5].

4.3 Compressed Sensing Implementations

There are several node level CS implementations in the literature. F Chen and A Chandrakasan presented a digital, hardware CS implementation in 90nm CMOS process. They used ASIC system coupled with radio to demonstrate that for an EEG dataset their system consumed 16% of power required by lossless LZW. This system used a random Bernoulli matrix as the measurement matrix which was generated from a seed shared with the reconstruction end. This allowed further bandwidth saving since the Φ matrix does not need to be transmitted for reconstruction. CS is performed on 1000 8bit sample block with only 50 16bit accumulator values transmitted over the radio. This results in compression ration of 90%. Raw data and reconstruction data is shown in figure 8

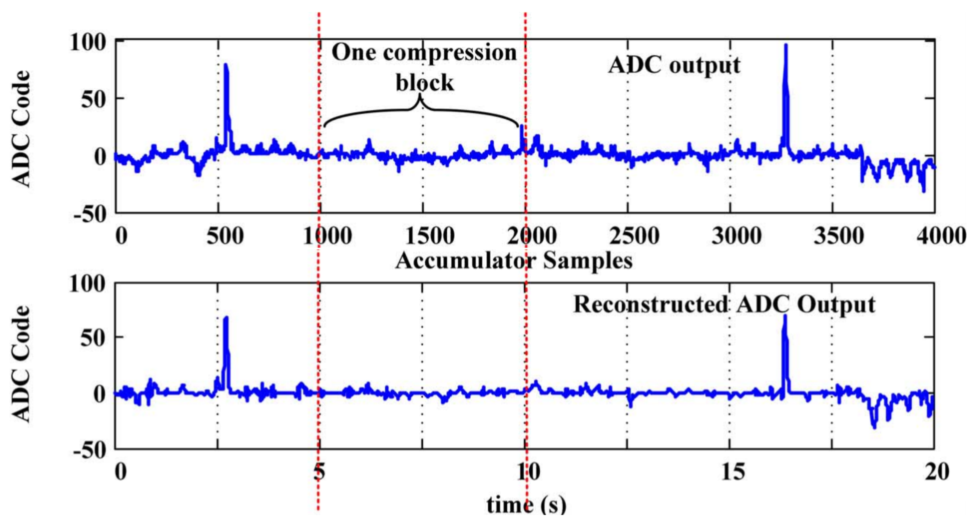

Figure 8: Raw Signal and reconstructed signal using the CS sampling system presented in[5].

There are situations where an analog CS frontend is desirable. In digital system [5] only 1000 sample block was used in the CS resulting in 50 measurements. Yet even in such a system the measurement matrix Φ is 1000×50 in size. This may not seem like a large amount of memory, but it has to be randomly generated via digital circuitry every sampling block. In cases of performing CS with images with number of pixels on the order of 10^6, generating and buffering a suitable Φ becomes difficult. One way to mitigate this issue was presented by M. Duarte and M. Davenport in their single pixel camera[9]. Their camera utilized a square array of computer controlled micromirrors called digital micromirror device(DMD), and a single pixel light sensor. An image is projected onto the DMD. Mirrors pointed towards the light sensor contributed reflected light to photo sensor signal, while those pointed away from the the camera contributed nothing. Thus such electro-mechanical system effectively computed a single row of $y = \Phi z$. By rapidly changing the DMD configuration and taking multiple exposures, authors effectively created an electromechanical CS front end. Their system was able to produce usable 256×256 pixel images from 1300 measurements. Since then a full silicon implementation of this device has been created, where an analog CS frontend takes place of the DMD[27]. Their integrated CS camera has been shown to outperform traditional DCT image compression methods especially at lower compression ratios[27].

In recent years academic scope has shifted from CS imaging to CS video. While video encoding is infeasible in many sensor network applications due to the high computational requirements, CS cameras offer low power consumption and high compression ratio. S. Pudlewski presented a CS video encoding system suitable for low power sesnor nodes[25]. Similarly to H.264 encoding, S. Pudlewski algorithms uses complete frames called Inter frames(I-frame) and incomplete Predicted frames(p-frames). In this CS video algorithm I-frames are composed of the CS measurements with $m \approx \frac{n}{10}$. In order to exploit the intra frame compressibility S. Pudlewski algorithm computes the difference in the CS measurements between two consecutive frames dv. If the dv vector is sparse enough another round of CS is performed in software, and only the $\approx \frac{m}{10}$ coefficients are encoded for every P frame. On the decoding side, I frames are extracted using a greedy L_1 minimization in the Wavelet domain. A similar minimization produces the dv vector for the P frames, which are accumulated, and minimized yet again to decode predicted frame. This approach has been shown to yield 170% compression improvement over H.264. Another advantage of this method is the low computational overhead, and high resilience to missing or corrupt measurements.

Compressed sensing has another advantage over traditional methods. It is very computationally difficult to reconstruct the vector x from the incoherent measurements without Φ[10]. Let's consider a system where matrix Φ is generated from a cryptographic key, and the sensor node and reconstruction server achieve a perfect key exchange. In such a system compressed sensing can combine measurements, compression and encryption in a single step. S. Chiu and H. Nguyen developed a prototype smart-meter system that demonstrates utility of CS in a privacy sensitive environment[10]. Their system, called Joint Data Compression and Encryption(JICE), uses a Gaussian random measurement matrix derived from a cryptographic primitive shared with the gateway node. Authors found that JICE significantly outperforms S-LZW and lossy wavelet compression algorithm similar to Wisden with AES encryption.

There are several key advantages of performing CS at the node level. Data sampling and compression can be combined yielding significant power savings. Analog to Digital converters in most CS systems operate at lower frequency, leading to additional power savings. Finally, compression ratio for many complex signals is comparable or better then the lossy and lossless compression methods. There are several disadvantages to node level CS. Firstly efficient incoherent measurement, which lies at the core of CS, requires specialized hardware. Because CS is a relatively young field measurement hardware is ad-hoc and tailored to the measurement. Secondly CS makes it harder to perform node level data analysis. In order to process the data, it must first be recovered via costly L_1 minimization, which defeats the purpose of CS.

5 Conclusions

Wireless sensor networks must overcome bandwidth, computational, and power limitations in order to make a distributed measurement. Lossless compression can minimize the bandwidth required for a sensor node to transmit data thereby trading power and bandwidth constraint for additional computational overhead. By exploiting redundancy in the input, lossless compression algorithms attempt to reduce the size of the input. In the sensor network domain, ad-hoc low power algorithms specifically tailored to the data are preferred to the general purpose counterparts. Lossy compression builds on the concept of lossless compression, but data is preprocessed to make the lossless compression easier. This preprocessing step irreversibly changes the data, making the recovery of the original input impossible. If the difference between the input and the output is less then the uncertainty of the measurements, no useful information is lost in the lossy compression process. Until 2006 lossy and lossless compression were the only tools available to sensor network designer for data compression. Since then, lossy compression research focus has shifted to compressed sensing. Compressed sensing combines sampling, compression and possibly encryption in a single step, without computational overhead inferred by lossy and lossless compression. While still a young field, compressed sensing has shown to significantly improve throughput of a sensor network, while maintaining a low power consumption[20]. Furthermore, since compressed sensing integrates well with hardware, in the next decade we are likely to see highly integrated, extremely low power, compressed sensing enabled general purpose sensor nodes become available to researchers. Lossless compression will prevail in fields where artifacts introduced by compressed sensing and lossy compression are unacceptable. Medical imaging for example relies almost entirely on lossless compression. If there is no node level data processing requirement compressed sensing allows for an efficient, low power, and low distortion method of data collection, compression and encryption. Lossy compression has a lot of utility in sensor network applications which require node level processing, and can tolerate some distortion. All three strategies can provide significant bandwidth and power savings, however it is important to select an approach which is suitable for a particular application.

References

[1] Mohammad Abu Alsheikh, Puay Kai Poh, Shaowei Lin, Hwee-Pink Tan, and Dusit Niyato. Efficient data compression with error bound guarantee in wireless sensor networks. In *Proceedings of the 17th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems*, pages 307–311. ACM, 2014.

[2] Emmanuel J Candes, Justin K Romberg, and Terence Tao. Stable signal recovery from incomplete and inaccurate measurements. *Communications on pure and applied mathematics*, 59(8):1207–1223, 2006.

[3] Emmanuel J Candès and Michael B Wakin. An introduction to compressive sampling. *IEEE signal processing magazine*, 25(2):21–30, 2008.

[4] Chee-Yee Chong and S.P. Kumar. Sensor networks: Evolution, opportunities, and challenges. *Proceedings of the IEEE*, 91(8):1247–1256, August 2003.

[5] Fred Chen, Anantha P Chandrakasan, and Vladimir M Stojanovic. Design and analysis of a hardware-efficient compressed sensing architecture for data compression in wireless sensors. *IEEE Journal of Solid-State Circuits*, 47(3):744–756, 2012.

[6] Abhimanyu Das and Debojyoti Dutta. Data acquisition in multiple-sink sensor networks. *ACM SIGMOBILE Mobile Computing and Communications Review*, 9(3):82–85, 2005.

[7] Irit Dinur. The pcp theorem by gap amplification. *Journal of the ACM (JACM)*, 54(3):12, 2007.

[8] David L Donoho. Compressed sensing. *IEEE Transactions on information theory*, 52(4):1289–1306, 2006.

[9] Marco F Duarte, Mark A Davenport, Dharmpal Takbar, Jason N Laska, Ting Sun, Kevin F Kelly, and Richard G Baraniuk. Single-pixel imaging via compressive sampling. *IEEE signal processing magazine*, 25(2):83–91, 2008.

[10] Juntao Gao, Xiuming Zhang, Hao Liang, and Xuemin Sherman Shen. Joint encryption and compressed sensing in smart grid data transmission. In *Global Communications Conference (GLOBECOM), 2014 IEEE*, pages 662–667. IEEE, 2014.

[11] Patrick R Gill, Albert Wang, and Alyosha Molnar. The in-crowd algorithm for fast basis pursuit denoising. *IEEE Transactions on Signal Processing*, 59(10):4595–4605, 2011.

[12] Prosanta Gope and Tzonelih Hwang. BSN-Care: A Secure IoT-Based Modern Healthcare System Using Body Sensor Network. *IEEE Sensors Journal*, 16(5):1368–1376, March 2016.

[13] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami. Internet of things (iot): A vision, architectural elements, and future directions. *CoRR*, abs/1207.0203, 2012.

[14] Chi-Fu Huang and Yu-Chee Tseng. The coverage problem in a wireless sensor network. *Mobile Networks and Applications*, 10(4):519–528, 2005.

[15] Qingqing Huang, Baoping Tang, Lei Deng, and Jiaxu Wang. A divide-and-compress lossless compression scheme for bearing vibration signals in wireless sensor networks. *Measurement*, 67:51–60, May 2015.

[16] S. Kalaivani and C. Tharini. Efficient Data Compression Technique using Modified Adaptive Rice Golomb Coding for Wireless Sensor Network. *ARPN Journal of Engineering and Applied Sciences*, 10(12):5395–5401, 2015.

[17] Silicon Labs. Evolution of wireless sensor networks. 2014.

[18] Che-Wei Lin, Chang Hong Lin, and Wei Jhih Wang. A power-aware code-compression design for RISC/VLIW architecture. *Journal of Zhejiang University SCIENCE C*, 12(8):629–637, August 2011.

[19] Yide Liu. Wireless Sensor Network Applications in Smart Grid: Recent Trends and Challenges. *International Journal of Distributed Sensor Networks*, 8(9):492819, September 2012.

[20] Jun Luo, Liu Xiang, and Catherine Rosenberg. Does compressed sensing improve the throughput of wireless sensor networks? In *Communications (ICC), 2010 IEEE International Conference on*, pages 1–6. IEEE, 2010.

[21] Stéphane G Mallat and Zhifeng Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on signal processing*, 41(12):3397–3415, 1993.

[22] F. Marcelloni and M. Vecchio. A Simple Algorithm for Data Compression in Wireless Sensor Networks. *IEEE Communications Letters*, 12(6):411–413, June 2008.

[23] Sangnam Nam. An uncertainty principle for discrete signals. *CoRR*, abs/1307.6321, 2013.

[24] Ranjeet S. Pisal. Implementation of Data Compression Algorithm for Wireless Sensor Network using K-RLE. *International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE)*, 3(11), 2014.

[25] Scott Pudlewski, Arvind Prasanna, and Tommaso Melodia. Compressed-sensing-enabled video streaming for wireless multimedia sensor networks. *IEEE Transactions on Mobile Computing*, 11(6):1060–1072, 2012.

[26] Carlos Ramirez, Vladik Kreinovich, and Miguel Argaez. Why l_1 is a good approximation to l_0: A geometric explanation. *Journal of Uncertain Systems*, 7(3):203–207, 2013.

[27] Ryan Robucci, Jordan D Gray, Leung Kin Chiu, Justin Romberg, and Paul Hasler. Compressive sensing on a cmos separable-transform image sensor. *Proceedings of the IEEE*, 98(6):1089–1101, 2010.

[28] Christopher M. Sadler and Margaret Martonosi. Data compression algorithms for energy-constrained devices in delay tolerant networks. In *Proceedings of the 4th international conference on Embedded networked sensor systems*, pages 265–278. ACM, 2006.

[29] Salman Salamatian, Asaf Cohen, and Muriel Mdard. Maximum Entropy Functions: Approximate Gacs-Korner for Distributed Compression. *arXiv preprint arXiv:1604.03877*, 2016.

[30] Thomas D Schneider. Information theory primer with an appendix on logarithms. In *National Cancer Institute*. Citeseer, 2007.

[31] Tom Schoellhammer, Ben Greenstein, Eric Osterweil, Michael Wimbrow, and Deborah Estrin. Lightweight temporal compression of microclimate datasets. *Center for Embedded Network Sensing*, 2004.

[32] Yi Sun, Can Cui, Jun Lu, and Qi Wang. Data compression and reconstruction of smart grid customers based on compressed sensing theory.

International Journal of Electrical Power & Energy Systems, 83:21–25, 2016.

[33] A. M. Tillmann and M. E. Pfetsch. The computational complexity of the restricted isometry property, the nullspace property, and related concepts in compressed sensing. *IEEE Transactions on Information Theory*, 60(2):1248–1259, Feb 2014.

[34] R. Viswanathan and P.K. Varshney. Distributed detection with multiple sensors I. Fundamentals. *Proceedings of the IEEE*, 85(1):54–63, January 1997.

[35] Ning Xu, Sumit Rangwala, Krishna Kant Chintalapudi, Deepak Ganesan, Alan Broad, Ramesh Govindan, and Deborah Estrin. A wireless sensor network for structural monitoring. In *Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 13–24. Acm, 2004.

[36] Davide Zordan, Borja Martinez, Ignasi Vilajosana, and Michele Rossi. On the performance of lossy compression schemes for energy constrained sensor networking. *ACM Transactions on Sensor Networks (TOSN)*, 11(1):15, 2014.