QUANTIFYING THE GAINS OF COMPRESSIVE SENSING FOR TELEMETERING APPLICATIONS

Philip Davis
Faculty Adviser: Dr. Charles Creusere
Klipsch School of Electrical and Computer Engineering
New Mexico State University

ABSTRACT

In this paper we study a new streaming compressive sensing (CS) technique that aims to replace high speed analog to digital converters (ADC) for certain classes of signals and reduce the artifacts that arise from block processing when conventional CS is applied to continuous signals. We compare the performance of both streaming and block processing methods on several types of signals and quantify the signal reconstruction quality when packet loss is applied to the transmitted sampled data.

KEYWORDS

Compressive Sensing, Sparse Signal Reconstruction

INTRODUCTION

Telemetering systems rely heavily on the continuous acquisition of sensed signals (e.g., airspeed, attitude, etc.) for control, navigation, and command decisions. This is crucial for the operation of aircraft, missiles, and other remote platforms. These systems contain hundreds of sensors and data sources that must be digitized and wirelessly transmitted to a base station, and much of the collected data needs to be compressed before transmission because of telemetry system bandwidth constraints. Thus, system designers must pick and choose what telemetry data gets wirelessly transmitted, and compression is also often applied to this data to increase throughput. The transition of telemetry systems from analog to digital has increased the dependence on high speed analog to digital converters (ADC) which must, according to the Nyquist criterion, sample an incoming signal at a rate equal to or greater than twice the highest frequency present. The use of high speed ADC's combined with compression increases power consumption as well as the cost of the system which can force the system's user to have to limit what kind of data is collected or how much of it gets transmitted. In an attempt to reduce cost and power consumption, we examine the use of compressive sensing (CS) on continuous signals as an alternative to normal Nyquist sampling followed by digital compression.

CS is a method of data collection that reduces the sampling rate for signals that meet certain criteria while at the same time non-adaptively compressing the signal prior to quantization. CS extracts the information content of a signal in the analog domain as it is being sensed by mixing the signal with a known sensing matrix, A. This process is referred to as Analog to Information Conversion (AIC) in [1]. In CS, it is know that if the signal being sampled is composed of only a few coefficients with respect to some representational basis (e.g., wavelet or Fourier domains) then it can be reconstructed perfectly later at the receiver using $\ell 0$ or $\ell 1$-sparse reconstruction techniques [2][3]. This reduces cost, complexity, and power usage at the sensing side and offloads it to the receiver where these issues are less of a concern. This paper puts forth a system utilizing CS for data acquisition and quantifies the real world restraints on such a system. We show that under certain circumstances telemetry systems can greatly benefit from CS.

Researchers have only started investigating the problem of compressively sensing continuously streamed analog data within the past few years [4][5][6][7] and many of the methods developed thus far process incoming signals as non-overlapping blocks. Block processing of the signal can lead to delays and artifacts in signal reconstruction. Recently, systems have been purposed that utilize overlapping sampling windows to mitigate the effects of blocking and to allow continuous real-time transmission of data with continuous reconstruction at the receiver [4][5]. In this paper we consider a sensing matrix that is composed of offset overlapping windows in an attempt to removing blocking artifacts in the reconstructed signal and to increase the reconstruction performance while using fewer multiplexing channels in the physical implementation. We compare the performance of these overlapping-windowed sensing matrices to that of blocked sensing matrices systems using varying window sizes and at several different compression ratios. The two main issues tackled in this paper that affect system performance are the effects of fixed-point quantization at the sensing side and packet loss during transmission. The performance of several classes of sensing matrix A are also quantified, including those whose elements are real normalized Gaussian-distributed, those whose elements are selected from a ternary alphabet (+, –, and 0), and those whose elements are selected from a binary alphabet (1, 0).

COMPRESSIVE SENSING BACKGROUND

CS is a relatively new in the field of signal processing wherein certain types of signals can be sampled at sub-Nyquist rates and thus compressed prior to quantization. Here, we present some basic background information on the subject. Consider the signal $\boldsymbol{x}$ to be a column vector of dimension N x 1, which can be written in terms of an N x 1 'sparse' vector $\mathbf{s}$ as $\boldsymbol{x} = \Psi\mathbf{s}$. In this context, by saying that $\mathbf{s}$ is sparse, we mean only K of its N elements are not equal to zero [2][3]

[8]. CS theory states that this signal can be sampled at much lower than the Nyquist rate and still be reconstructed without loss as long as that $K << N$ [2][8]. Furthermore, it is not necessary to sample the signal in the sparse basis Ψ; in fact, the signal can be sampled using a fixed (i.e., non-signal dependent) M x N sensing matrix A as long as A is largely incoherent with Ψ [8][1]. As it turns out, a number of sensing bases having been shown to satisfy the incoherence requirement with respect to a wide range of sparsity bases. This sensing matrix A samples a length N signal and effectively "compresses" it to M samples where $M < N$. Mathematically, the sampling of the signal can be seen in (1) where $\boldsymbol{e}$ is M x 1 zero mean Gaussian noise with variance σ.

$$\boldsymbol{y} = A\boldsymbol{x} + \boldsymbol{e} \qquad (1)$$

In (1) M inner products of the rows of the sensing matrix A are taken with respect to the column vector $\boldsymbol{x}$ in order to form the samples $\boldsymbol{y}$, which is an M x 1 column vector. It should be made clear that in order to capture information on all K elements we must take at least $M > K$ samples since this would be the minimum number of coefficients need to represent the signal if it was sampled directly in its basis of sparsity (i.e., without CS) [2]. In order to use CS, the sparsity of the signal being sensed should be known or estimated beforehand so that a sufficient but not excessive sampling density is achieved. Conversely, it is not necessary to know the basis of sparsity at the time of compressive sampling, but this knowledge *is* required in order to reconstruct the signal at the receiver. The signal $\boldsymbol{x}$ is compressible because most of its information is represented by only a few coefficients with respect to the Ψ basis. In order to efficiently sample $\boldsymbol{x}$ so that each individual sample, y_i, contains new information content, it is important that the sensing matrix A be maximally incoherent with Ψ. Several random distributions can be used to form A so that it satisfies this low incoherence criteria including independent identically distributed (iid) Gaussian elements, ternary elements, and binary elements [2][3]. Several methods for signal recovery have been developed, including a number that are based on the principles of basis and matching pursuit. Matching pursuit algorithms use stepwise greedy gradient decent procedures to iteratively reconstruct a sensed signal. This is often faster and easier to implement than the convex programming involved in basis pursuit, but it requires more measurements to achieve the same level of reconstruction quality [9]. Basis pursuit methods rely heavily on $\ell 1$ minimization for signal reconstruction. The typical $\ell 1$ optimization criteria (with the required quadratic constraint) is given below in (2) where ϵ is a user-selected tolerance and $\tilde{\boldsymbol{x}}$ is the estimated signal [9].

$$\min \|\tilde{\boldsymbol{x}}\|_1 \text{ subject to } \|A\tilde{\boldsymbol{x}} - \boldsymbol{y}\|_2 \leq \epsilon \qquad (2)$$

STREAMING COMPRESSIVE SENSING

In real world and telemetering applications, the signal $\boldsymbol{x}$ is rarely of finite length and can often times be viewed as infinite in length from a signal processing point of view. In this paper, we compare two methods of dealing with continuously streaming signals. The first way is by breaking the signal up into nonoverlapping finite-length blocks and applying the sampling matrix A to each of these blocks separately. Thus, for each block of N time samples, we transmit M sampled values. The physical implementation of this that we simulate uses M channels, each of which effectively multiplies a different row of the sampling matrix A with the input signal, integrates the result, and feeds this into a low-speed (in relative terms) ADC. One channel of this physical implementation can be seen in Figure 1. Integration is performed over a time period of T which is equivalent to the duration of the block of input samples. Even though M ADCs are required in this implementation, the sampling rate of each is so low that the combined sampling density of all M ADCs is significantly less than what Nyquist theory requires for a conventional single ADC system operating directly on the time domain signal. The formulation above is similar to those proposed in [6] and [7]; in [6], however, the authors use slightly overlapping blocks to suppress blocking artifacts while [7] uses low pass filters instead of integrators.

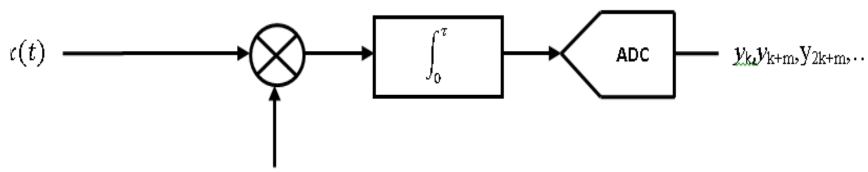


Figure 1. One channel of physical implementation of block processing method

The second sampling approach evaluated in this paper uses a sampling matrix, A_s, which has staggered, overlapping row vectors so that the signal is effectively continuously sampled. This concept of streaming compressive sensing is presented in [4] and [5]. A_s is generated this way to facilitate real-time data acquisition while decreasing hardware complexity and removing blocking artifacts that can arise from the first method. Each row of A_s has a window size of W whose elements are chosen from a random distribution and is offset from the previous window with a shift of V elements. This is seen in Figure 2. M, the number of rows is set and the number of columns is determined by $N = M*V$, which results in a 1:1/V the compression ratio [4][5].

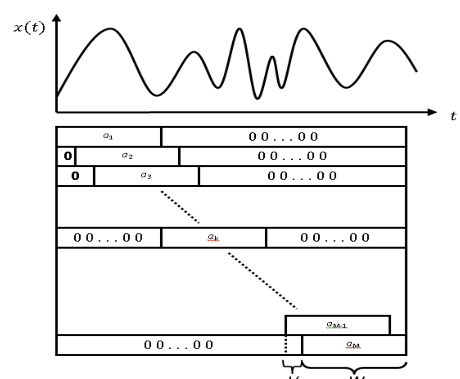


Figure 2. Overview of overlapping windowed sampling matrix A_s

In this research we consider ternary, binary, and floating-point Gaussian distributions for the sensing matrices A and A_s. For A_s the rows of the Gaussian distribution are made mutually orthonormal using a Gram-Schmidt process to maximize the sampling efficiency. The first row of A_s is normalized Gaussian (i.e., its magnitude is 1) and contains only W consecutive non-zero values. The next row contains another W Gaussian samples but is shifted to the right by V positions with the remaining samples being zero again. Each successive row is made orthonormal with the ones above it. This process is then repeated. Furthermore, A_s is made to be cyclic so that when its rows wrap around and start repeating the ending coefficients of rows a_{m-j}, a_{m-j+1}, … , a_m are still orthonormal with rows a_1, a_2, … , a_j where $j = (W/V) - 1$. Being cyclic greatly simplifies the hardware implementation. One of the key benefits of this method is that it reduces the number of multiplexer channels that are required in hardware because once one window has been completed in a channel the coefficients corresponding to the next non-overlapping row of A_s can be applied to the signal. For binary sampling, the W elements of the of each row are randomly selected from $\left(\frac{+1}{\sqrt{(N)}}, -\frac{1}{\sqrt{(N)}}\right)$ and for ternary sampling the W elements in each row are randomly selected from $\left(\frac{+1}{\sqrt{(N)}}, -\frac{1}{\sqrt{(N)}}, 0\right)$. The hardware implementation of this method is seen in Figure 3.

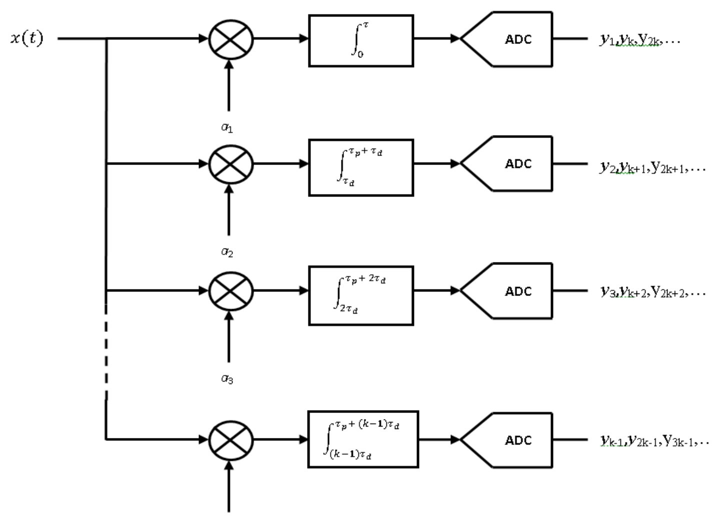


Figure 3. Hardware implementation of overlapping windowed CS

A similar scheme is used in [6] for spectrum sensing in cognitive radio networks, but in that work the A matrix is not staggered which requires the hardware to possess as many integrator/ADC pairs as samples per block and would likely result in blocking artifacts for more general applications. Generating the sensing matrix beforehand allows its values to be stored in on board memory of the transmitter (sampling) circuitry and does away with the need to generate random values in hardware which decreases system complexity.

SIGNAL RECOVERY

In CS there are several methods of signal reconstruction based on different optimization principles. In this study we rely on basis pursuit (BP) to recover our sparse signal of interest from sampled data that was collected in blocked segments. Specifically, we use $\ell 1$ minimization with a quadratic constraints due to the noise simulated in the data collection. The software package in [9] is utilized here for this reconstruction. Reconstruction of the continuous CS method is done with the streaming greedy algorithms described in [4] and [5]. The main metric for signal recovery in the paper is Signal to Error Ratio (SER) and is given below in (3) [4] [5].

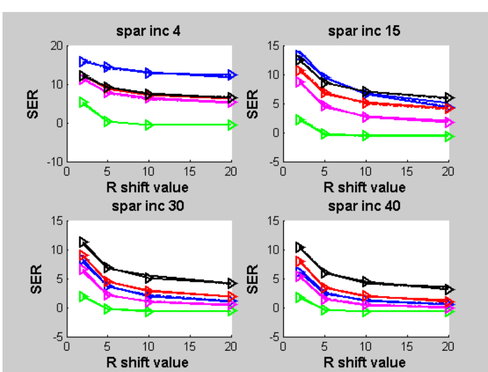
$$\mathrm{SER} = 20\log_{10} \|\boldsymbol{x}\|/\|\boldsymbol{x}-\tilde{\boldsymbol{x}}\|_2 \tag{3}$$

RESULTS

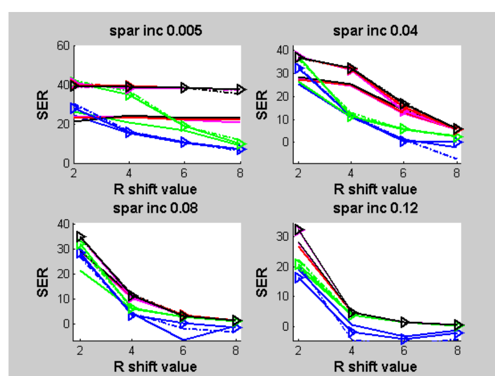
First, we compare the blocking CS versus the performance of the streaming method on signals that are sparse in the time domain and frequency domain. By sparse in the time domain, we mean that the signal is zero valued (or effectively so) much if not most of the time; sparse in the frequency domain, on the other hand, implies that the signal is composed of a relatively small number of sinusoidal components, the exact mixture of which may vary with time. For time domain signals, we evaluate the two methods over the following compression ratios: 50%, 75%, 83.33%, and 87.5% which correspond to shift values of V = 2, 4, 6, 8 when using the streaming CS method. We also tested each approach over a range of sparsity rates, specifically 0.005, 0.04, 0.08, and 0.12. For each of these test signals 6000 measurements were taken and the results were averaged over 12 trials. The window size for the streaming CS method was set at W = 80 here. For the signals that were sparse in the frequency, the compression rates tested were 50%, 80%, 90%, and 95% which was equivalent to shift values of V = 2, 5, 10, and 20. The frequency-sparse signals were composed of 10 segments whose lengths were selected randomly but whose total length was forced to be V*3000. Sparsity for each segment was tested at 4, 15, 30, and 40 frequency components. 3000 measurements were taken and the results were averaged over 12 trials. For these tests, the window size for the streaming CS method was set to W = 40. For both types of signals, the measurement samples (i.e., the elements of vector $\mathbf{y}$) were quantized to 8 bits, 24 bits, and double precision floating point accuracies so that the transmission performance could be quantified. Quantization is simulated at the ADC and memory storage is assumed for the Gaussian sensing matrix coefficients while for the ternary and binary distributions it is assumed that an analog switch along with a voltage divider is used (to effectively scale the sampling vectors by $\sqrt{(N)}$) so that no memory or digital-to-analog conversion is required. Quantization of the measurements is tested to determine what accuracy is required to achieve the optimal results since requiring higher resolution ADCs typically increases their cost. When comparing the cyclic-windowed sensing matrices to the blocking method, we test against 4 different block sizes. For reconstruction of signals that are sparse in the frequency domain, the cyclic-windowed sensing matrix A_s is taken with $M_s = W/V - 1 + 70$

and N_s is set accordingly. For signals sparse in the time domain, A_s is taken with $M_s = 3*W$ and N_s is again set accordingly. The first block size tested, labeled recSig1 in Figure 4, has M set equal to the number of analog multiplexing channels that would be needed to implement the cyclic-windowed sensing matrix where multiplexing channels are reusable every W/V samples due to their overlapping nature. For the second block size tested, recSig2, N is set equal to N_s, which is the number of columns used in A_s. For the third and forth block sizes, labeled recSig3 and recSig4 in Figure 4, $N = 2*N_s$ and $N = 4*N_s$ respectively.

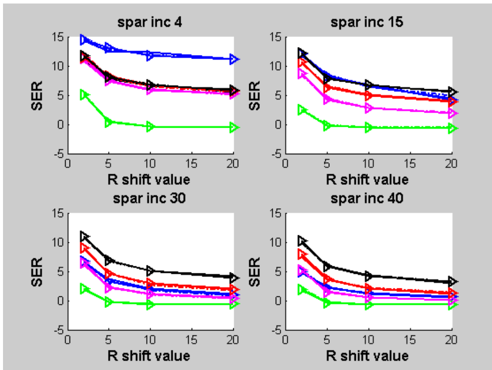
Figure 4 shows a summery of the results from these simulations. Figures 4(a)(c)(e) are for signals sparse in the frequency domain while Figures 4(b)(d)(f) are for signals sparse in the time domain and Figure 4(g) give the legend for the figures. Figures 4(a)(b) were generated using the Gaussian sensing matrix, Figures 4(c)(d) were generated using the binary sensing matrix, and Figures 4(e)(f) were generated using a ternary sensing matrix. As can be seen in Figures 4(a)(c)(e), signals sparse in the frequency domain suffer little from quantization and digitization. In fact, there seems to be no performance advantage of using 24 bits over 8 bits for both the blocking and streaming methods. Also, from Figures 4(a)(c)(e) it can be seen that cyclic-windowed sensing matrices outperform blocked sensing matrices with the same number of effective rows (which equate to the number of analog integrator channels in front-end hardware) and they also outperform larger sized reconstruction blocks at low sparsities. From Figures 4(a)(c)(e) it can be seen that the Gaussian sensing matrices outperform the ternary and binary distributions. Reconstruction of signals sparse in the time domain behave differently than reconstruction of signals sparse in the frequency domain. Figures 4(b)(d)(f) show that block processing of time domain sparse signals seems to do better in all cases except at the lowest sparsity rate at a 50% compression ratio for all three sensing matrix distributions. It can be seen that the largest three block processing sensing matrices, recSig2, recSig3, and recSig4, perform almost identically in Figures 4(b)(d)(f). Also, for block processing, floating point and 24 bit quantization perform at about the same level with no real significant performance gain from floating point, however, at sparsity rates of 0.005 and 0.04 it can be seen that 24 bit quantization outperforms 8 bit quantization significantly, while at the higher sparsity rates cyclic-windowed CS and block processing all start to perform similarly.

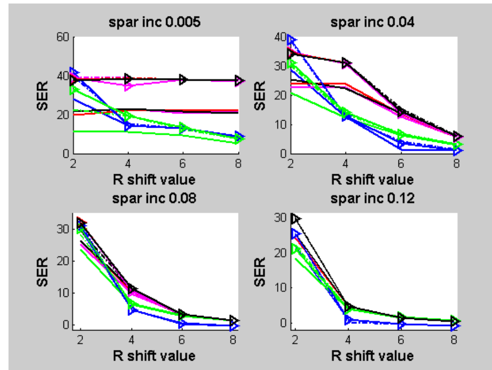


(a) Freq. Sig. Gaussian Dist.

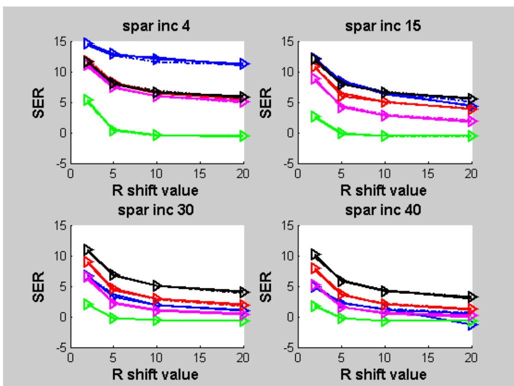


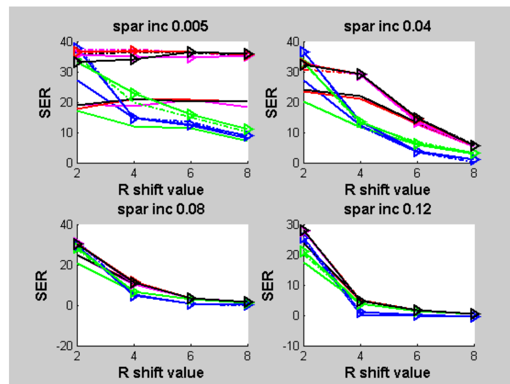
(b) Time Sig. Gaussian Dist.



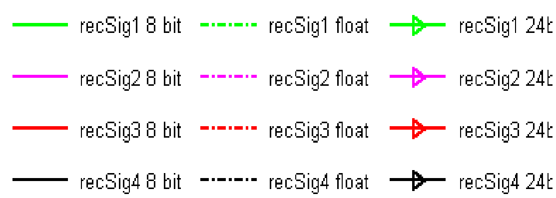


(c) Freq. Sig. Binary Dist. (d) Time Sig. Binary Dist.





(e) Freq Sig. Ternary Dist. (f) Time Sig. Ternary Dist.

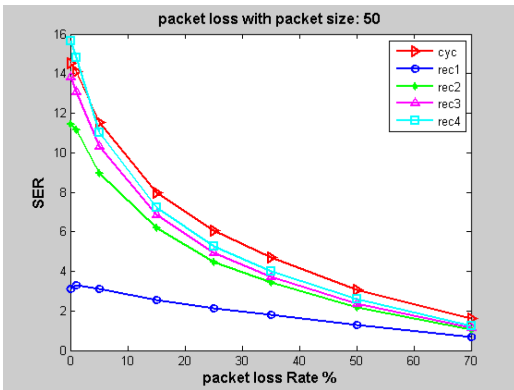


(g) Legend

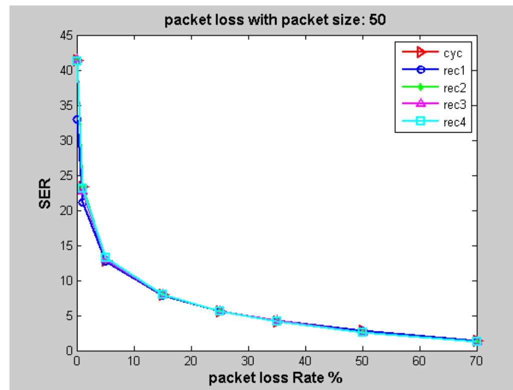
Figure 4. Results for varying degrees of measurement quantization

In any digital wireless transmission environment, packet loss is a major factor that must be examined since in some real-time telemetry applications there may be little or no time for packet retransmission. Some studies on packet loss in CS systems have been previously presented in [10] and [11]. Here, we compare the effects of packet loss on signal reconstruction using the blocking and streaming CS methods discussed earlier, assuming that retransmission is not an allowable option. We first look at frequency-sparse signals composed of 10 segments with 8 frequency components per segment. Packets are composed of 50 measurement samples and we test over a range of packet loss rates. The results of this can be seen in Figure 6(a). Next, packet loss is tested on a signals sparse in the time domain. The time signal has a sparsity rate of 0.04. Figure 6(b) shows the results from these test. For both the time and frequency sparse test signals

10000 time domain samples are collected and results are averaged over 20 trials. In each case a packet size of 50 samples is used, an SNR of 40 dB is simulated at the sensing side, and a compression ratio of 50% is implemented. As noted in[11], it is seen here that for both time and frequency sparse signals reconstruction performance decreases smoothly as the packet loss rate increases. For frequency sparse signals staggered sampling seems to perform slightly better than blocked sampling as packet loss rate increases. For time sparse signals blocking and staggered sampling appear to perform equally as the packet loss rate increases.



(a) Freq. Sparse Sig.



(b) Time Sparse Sig

Figure 6. Signal reconstruction over range of packet loss percentages

CONCLUSION

From these experiments we see that staggered overlapping CS is a viable option for sampling signals sparse in the frequency domain. While extremely large blocks might outperform streaming CS, these block sizes might be unrealistic to implement in hardware and the effects of blocking artifacts have not been quantified here. For signals sparse in the time domain, block processing outperforms the streaming method and there is a noticeable gain in reconstruction when using 24 bit resolution over 8 bit at lower sparsities. Block processing may perform better than streaming on time-sparse signals because the structure of the signal is time independent while the frequency sparse signals have information content that can span several blocks. The next logic step in this research is to start testing these methods on real world signals such as those sent through telemetry systems to quantify the effectiveness in reducing hardware complexity while at the same time efficiently sending data. Furthermore, while artifacts from block processing were easily noticeable in early tests on audio signals, comparisons with the streaming method have yet to be performed. Testing on signals that are sparse in other domains, such as the Hadamard basis, should also be done to see which type of sampling method performs best.

REFERENCES

[1] S. Kirolos, J. N. Laska, M. B. Wakin, M. F. Duarte, D. Baron, T. Ragheb, Y. Massoud, and R. G. Baraniuk, "Analog to information conversion via random demodulation," in Proc. 2006 IEEE Dallas/CAS Workshop on Design, Applications, Integration and Software, Dallas, TX, Oct. 2006, pp. 714.

[2] R. Baraniuk, "Compressive Sensing," IEEE Signal Processing Magazine, 2007.

[3] E. Candes, M. Wakin, "An Introduction To Compressive Sampling*,"* IEEE Signal Processing Magazine, 2008.

[4] T.B. Petros, M.S. Asif, "Compressive Sensing for streaming signals using the Streaming Greedy Pursuit," Military Communications Conference, MILCOM 2010, vol., no., pp.1205-1210, Oct. 31 2010-Nov. 3 2010

[5] P. Boufounos, M.S. Asif, "Compressive sampling for streaming signals with sparse frequency content," *Information Sciences and Systems (CISS),* 2010 44th Annual Conference on , vol., no., pp.1-6, 17-19 March 2010

[6] Z. Yu, X. Chen, S. Hoyos, B. Sadler, J. Gong, and C. Qian, "Mixed-Signal Parallel Compressive Spectrum Sensing for Cognitive Radios," International Journal of Digital Multimedia Broadcasting, vol. 2010, Article ID 730509, 10 pages, 2010.

[7] M. Mishali, Y.C. Eldar, O. Dounaevsky, E. Shoshan, "Xampling: Analog to digital at sub-Nyquist rates," *Circuits, Devices & Systems, IET* , vol.5, no.1, pp.8-20, January 2011

[8] W. Barakat, R. Saliba, "Compressive Sensing For Multimedia Communications in Wireless Sensor Networks," 2008.

[9] J. Romberg, E. Candes, "L1-Magic: Recovery of Sparse Signals via Convex Programming," October 2005. http://www.acm.caltech. edu/l1magic/.

[10] Pinheiro, E.C., Postolache O.A., and Girao P.S., "Implementation of Compressed Sensing in Telecardiology Sensor Networks," International Journal of Telemedicine and Applications, vol. 2010.

[11] H. Garudadri, P.K. Baheti, "Packet Loss Mitigation for Biomedical Signals in Healthcare Telemetry," 2009 Annual International Conference of the IEEE Engineering in Medicine and Biology Society