Low Complexity Sensing for Big Spatio-Temporal Data

Dongeun Lee and Jaesik Choi*
School of Electrical and Computer Engineering
Ulsan National Institute of Science and Technology
Ulsan 689-798, Korea
**Corresponding Author*
eundong@unist.ac.kr, jaesik@unist.ac.kr

***Abstract*—Many large scale sensor networks produce tremendous data, typically as massive spatio-temporal data streams. We present a *Low Complexity Sensing* framework that, coupled with novel compressive sensing techniques, enables to reduce computational and communication overheads significantly without much compromising the accuracy of sensor readings. More specifically, our sensing framework randomly samples time-series data in the temporal dimension first, then in the spatial dimension. Under some mild conditions, our sensing framework holds the same theoretical bound of reconstruction error, but is much simpler and easier to implement than existing compressive sensing frameworks. In experiments with real world environmental data sets and a synthetic data set, we demonstrate that the proposed framework outperforms two existing compressive sensing frameworks designed for spatio-temporal data.**

***Keywords*-compressive sensing; random sampling; spatio-temporal data; energy efficient sensing; sparse signal recovery**

I. Introduction

Various sensing devices including mobile phones and biomedical sensors are essential nowadays. Individually operating sensors usually form correlated sensor networks in large scale. Thus, these sensors generate continuous flows of *big sensing data* that pose important challenges: how to sense and transmit massive data all in efficient manner.

An efficient sensing scheme is especially demanding in resource-limited sensors where computational power, network bandwidth, and energy are limited. Apart from the case of resource-limited sensors, full-fledged sensors such as mobile phones can also benefit from the efficient sensing scheme: broader sensing coverage and participation of users are made possible with the efficient sensing.

In this paper, we propose an efficient compressive sensing (CS) framework that significantly reduces computational and communication overheads of participating sensors. CS has a salient feature of shifting the complexity burden of encoding process to the decoder that is the collection point of big sensing data in our context. Many conventional distributed sensing schemes demand a certain degree of cooperation among neighboring sensors [1]–[3]. The key difference of our CS framework is that the proposed sensing framework does not require any additional coordinated action or routing structure on the network of sensors.

Specifically, our CS framework is based on random sampling [4] and includes a new multi-dimensional random sampling when sensor data have sparse representations in spatial and temporal dimensions (i.e., small numbers of principal components). The new sampling suits the needs of resource-limited sensors, and also shows an improved coding efficiency compared to other CS schemes that mostly take account of a single dimension [2], [3].

In experiments with environmental sensor data, we demonstrate that our sensing framework outperforms a state-of-the-art CS designed for the joint coding of spatio-temporal correlated sensor data [5], in terms of the coding efficiency. We also present the computational and communication complexities of our scheme compared with existing schemes.

II. Background on Compressive Sensing

Data compression is required in the context of dense sensing environment where network bandwidth is a scarce resource: each sensor simply transmitting raw data incurs a bottleneck on the channel toward the collection point (base station in case of wireless sensor networks). When the compression scheme is applied, more sensing data can be transmitted to the collection point.

A. Compressive Sensing

However the near-optimal coding of conventional compression schemes is not applicable to many resource constrained devices due to its complexity. Compressive sensing or compressed sampling (CS) can be an option for shifting the complexity burden to the decoder (the collection point, etc.) where original signal is estimated in best-effort manner [6], which can be applied to various types of resource-limited ones such as biosensors.

Most phenomena captured by sensors can be represented with only a few components using approximations of Karhunen-Loève transform. CS operates very differently from conventional compression schemes as if it were possible to directly acquire these significant coefficients of transforms, i.e., just the important information. In CS, a signal is projected onto random vectors whose cardinality is far below the dimension of the signal. For instance, consider a signal $\mathbf{x} \in \mathbb{R}^N$ that can be compactly represented in some orthogonal basis $\mathbf{\Psi}$ with only a few large coefficients and many small coefficients close to zero as follows:

$$\mathbf{x} = \mathbf{\Psi}\mathbf{s}, \tag{1}$$

where $\mathbf{s} \in \mathbb{R}^N$ is the vector of transformed coefficients with a few significant coefficients.

In (1), $\mathbf{\Psi}$ could be any orthogonal basis that makes $\mathbf{x}$ sparse in transform domain such as discrete cosine transform (DCT) and wavelet transforms. The signal $\mathbf{x}$ is called ***K*-sparse** if it is a linear combination of only $K \ll N$ basis vectors. Note that a real world signal in general is not exactly K-sparse; rather it can be closely approximated with K basis vectors.

CS projects[1] $\mathbf{x}$ onto a random sensing basis $\mathbf{\Phi} \in \mathbb{R}^{M \times N}$ as follows ($M < N$):

$$\mathbf{y} = \mathbf{\Phi}\mathbf{x} = \mathbf{\Phi}\mathbf{\Psi}\mathbf{s}, \tag{2}$$

where $\mathbf{\Phi}$ is generally constructed by sampling independent identically distributed (i.i.d.) entries from the Gaussian or other sub-Gaussian distributions whose moment-generating function is bounded by that of the Gaussian (e.g., Rademacher distribution).

Consequently, $\mathbf{\Phi}$ is dense with virtually every entry set to non-zero real numbers, which causes two issues: (i) the sensing matrix $\mathbf{\Phi}$ occupies a substantial amount of storage, (ii) this leads to $O(MN)$ multiplication and summation operations. Both of these issues can be costly to resource-limited sensors without specific CS-supporting architectures [6].

There has been workarounds for these problems that replace the Gaussian entries of $\mathbf{\Phi}$ with structured random matrices such as the random selection of rows from (forward) Fourier transform matrix [7]. In this case, the signal $\mathbf{x}$ has to be scrambled via a random permutation of its coefficients, in order to bring a randomness as in the Gaussian matrix. Using structured random matrices, one need only store indices of randomly selected rows and permutation sequence instead of storing entire entries of Fourier matrix, which is a significant saving for the storage space. In addition, structured random matrices such as Fourier transform and DCT matrices have the fast algorithms available whose computational complexity is $O(N \log N)$ ($\log N < M$ in general).

The system shown in (2) is ill-posed as the number of equations M is smaller than the number of variables N: there are infinitely many $\mathbf{x}$'s that satisfy $\mathbf{y} = \mathbf{\Phi}\mathbf{x}$. Nevertheless this system can be solved with overwhelming probability provided that $\mathbf{s}$ is sparse and M is large enough such that $M = O(K \log(N/K))$ in the case of Gaussian sensing matrix and $M = O(K \log N)$ in the case of the structured random matrices [4].

B. General Signal Recovery

A signal recovery algorithm takes measurements $\mathbf{y} \in \mathbb{R}^M$, a random sensing matrix $\mathbf{\Phi}$, and the sparsifying basis $\mathbf{\Psi}$. In a typical setup, the only information an encoder always has to send is $\mathbf{y}$. The sensing matrix $\mathbf{\Phi}$ can be explicitly sent [3] or reconstructed using meta information such as the seed of pseudorandom number generator [2], depending on application. In contrast, the sparsifying basis $\mathbf{\Psi}$ is assumed to be known to a decoder. In fact, if a better sparsifying basis is found, then the same $\mathbf{y}$ can be used to reconstruct an even more accurate view of the original signal $\mathbf{x}$.

The signal recovery algorithm then reconstructs $\mathbf{s}$ knowing that $\mathbf{s}$ is sparse. Given $\mathbf{s}$, the original signal $\mathbf{x}$ can be recovered through (1). It has been shown that the following linear program gives an accurate reconstruction of $\mathbf{s}$:

$$\operatorname{argmin} \|\tilde{\mathbf{s}}\|_1 \qquad \text{subject to} \qquad \mathbf{\Phi}\mathbf{\Psi}\tilde{\mathbf{s}} = \mathbf{y}. \tag{3}$$

There are many efficient algorithms that solve (3) using either linear program approaches or iterative, greedy searches [6].

C. Noisy Signal Recovery

Suppose $\mathbf{y}$ were corrupted with a noise $\mathbf{z} \in \mathbb{R}^M$ that is a stochastic or deterministic unknown error term, which could be from various sources such as communication and quantization. The corrupted $\hat{\mathbf{y}}$ can be represented as

$$\hat{\mathbf{y}} = \mathbf{\Phi}\mathbf{\Psi}\mathbf{s} + \mathbf{z}. \tag{4}$$

It has been shown that (4) can be solved using the following minimization problem with relaxed constraints for reconstruction:

$$\operatorname{argmin} \|\tilde{\mathbf{s}}\|_1 \qquad \text{subject to} \qquad \|\mathbf{\Phi}\mathbf{\Psi}(\mathbf{s} - \tilde{\mathbf{s}}) + \mathbf{z}\|_2 \le \eta\sqrt{M}, \tag{5}$$

where $\eta\sqrt{M}$ bounds the amount of noise in the signal. Problem (5) is often called LASSO and can also be solved using efficient algorithms similar to the noiseless case [6].

III. Low Complexity Sensing Framework

The sensing/sampling paradigm explained in Section II can be applied to signals that can be represented in one-dimensional vectors, i.e., $\mathbf{x}$. In a distributed sensing context, these vectors not only correspond to time-series data of individual sensors in temporal dimension, but also to data in spatial dimension from a group of sensors at a specific time instant, in which case two-dimensional data can be vectorized into an one dimension.

[1] This can also be seen as inner product operations.

A. Handling Spatio-Temporal Dimension

However, applying the compressive sensing (CS) technique to both spatial and temporal dimensions is not immediately evident. One may want to vectorize one dimension first and the other dimension next, and so on (e.g., vectorize the spatial dimension and then the temporal dimension). Yet this naive approach has problems. First, finding $\mathbf{\Psi}$ that sparsifies $\mathbf{x}$ could be a daunting task. Next, even if we could find a proper order in vectorizing mutually correlated signals, the vectorizing would incur additional complexity that brings control and communication overheads.

Due to these difficulties, most CS literature focused only on one dimension, especially the spatial dimension and proposed their own ways of coordination among distributed sensors to achieve the random measurements of distributed data [2], [3]. On the other hand, CS in the temporal dimension is straightforward: each sensor can take random measurements in its temporal dimension and send them to the collection point using generic routing schemes.

Nevertheless, more efficient sensing techniques that exploit the joint correlation of spatio-temporal dimension are solicited for improved coding efficiency. In this regard, a few pioneering works of *distributed compressive sensing* were proposed that exploited the joint correlation of spatio-temporal dimension [5], [8]. Their schemes show a significant performance improvement compared to other CS schemes that take account of a single dimension as shown in Section V. However they suffer from computational and communicational overheads as shown in Section IV.

B. Random Sampling in Spatio-Temporal Dimension

As discussed in Section II-A, the random sensing matrix $\mathbf{\Phi}$ in (2) is a dense matrix, which causes complexity in both storage and computation. This can be mitigated with the use of structured random matrices that reduces storage requirement and brings down computational requirement from $O(MN)$ to $O(N \log N)$. Nevertheless, even this requirement could be unacceptable to resource-limited sensors that should follow the rapid generation rate of *big sensing data*.

Thus it is imperative that a more efficient way of sensing is investigated which is far less complex than projecting the original signal onto a *dense* random sensing basis. The sparse random sensing matrix composed of a few ones for each column and zeros for all other entries is one of solutions [9], which has the computational complexity of $O(N \log(N/K))$.

In this paper, we utilize an even more efficient sensing mechanism based on *random sampling* whose computational time complexity is constant [4]. The random sampling scheme is based on the fact that it is possible to construct $\mathbf{\Phi}$ in (2) from a random selection of rows (without replacement) from the identity matrix $\mathbf{I}$, which is equivalent to the random sampling of coefficients in $\mathbf{x}$.[2] Note that this scheme works only if the sparsifying basis $\mathbf{\Psi}$ is dense such as the DCT and wavelet transform bases, in order not to violate *incoherence*[3], which is a sufficient condition for the successful recovery of the original signal [4], [6]. (The specific choice of $\mathbf{\Psi}$ depends on how well it sparsifies data sets a decoder handles as explained in Section II-B.) The random sampling of signal in the CS setup is illustrated in Fig. 1. Here, the number of required measurements M is the same as in the case of the structured random matrices, that is, $M = O(K \log N)$.

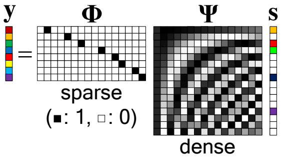

Figure 1. Random sampling of a signal in (2).

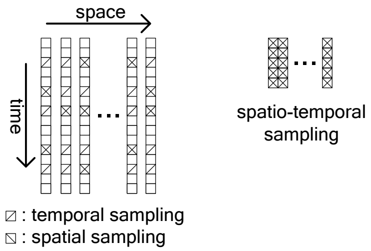

Figure 2. Low complexity sensing of spatio-temporal data.

1) Low Complexity Sensing: Our *low complexity sensing framework* randomly samples the original signal in both spatial and temporal dimensions. First, each sensor randomly samples its time-series data in the temporal dimension using the same indices shared across different sensors, which is equivalent to using the same random sensing matrix $\mathbf{\Phi}$ across different sensors for the same time frame. This sampling reduces the lengths of original time-series data of sensors.

Next, the random sampling is performed in the spatial dimension: the group of randomly sampled coefficients from each sensor is sampled again in the spatial dimension, which is illustrated in Fig. 2. Therefore, in reality each sensor needs to sample and transmit only coefficients that are selected in both spatial and temporal dimensions. Here the rationale behind using the same random sampling indices in the temporal dimension is to maximize the spatial correlation of coefficients across different sensors: the spatial correlation is likely to be stronger at the same time instant than with different time instants. In this way, the joint correlation of spatio-temporal dimension can be exploited.

[2] Therefore, a subset of coefficients is randomly selected.

[3] The two bases $\mathbf{\Phi}$ and $\mathbf{\Psi}$ are (maximally) incoherent when the the largest correlation between any two elements of $\mathbf{\Phi}$ and $\mathbf{\Psi}$ is $1/\sqrt{N}$ where N is the order of two square matrices.

This sensing framework is especially useful in the distributed sensing context thanks to the *opt-in* and *opt-out* nature of participating sensors. In a distributed sensing, each sensor may want to participate in or not depending on its remaining energy [1] or its willingness to volunteer in the context of *participatory sensing* [10].

Regarding how to generate random numbers used for random sampling of spatio-temporal data, popular approaches are to use pseudorandom numbers [2]. These random numbers should be synchronized between encoder (sensors) and decoder (the collection point, etc.), in order to ensure the correct recovery of the original signal. Randomness can also be improved via periodically updating random seeds between encoder and decoder.

Furthermore, the random indices for the spatial sampling need not be explicitly synchronized between encoder and decoder if each sensor determines whether to transmit or not at every time instant that corresponds to the temporal sampling point. The decision can be made based on the opt-in and opt-out policy or the transmission probability defined for each sensor. This is possible because the collection point can recognize the spatial index required for the reconstruction when it receives sampled sensing data from each sensor.

2) Signal Recovery: The collection point of sensing data takes a random sample measurement matrix as described in Fig. 2. Note that the recovery problem in this case is different from conventional CS recovery in the sense that we are dealing with a measurement *matrix*, not a measurement *vector*. Thus we need to decode the measurement matrix. (When the random indices for the spatial sampling is not explicitly synchronized between encoder and decoder, the spatio-temporal measurement no longer has a matrix form since the number of spatial sampling can vary between time instants. However, the decoding process does not impose the matrix form on the spatio-temporal measurement.)

First, each row of the measurement matrix is decoded following the recovery procedure explained in Section II-B. Specifically, the solution $\mathbf{s}^\star$ to (3) obeys

$$\|\mathbf{s}^\star - \mathbf{s}\|_2 \leq C_0 \cdot \|\mathbf{s} - \mathbf{s}_K\|_1 \tag{6}$$

for some constant C_0, where $\mathbf{s}_K$ is the vector $\mathbf{s}$ with all but the largest K components set to 0: the quality of recovered signal is proportional to that of the K most significant pieces of information. We get progressively better results as we compute more measurements since $M = O(K \log N)$ [6]. Therefore, $\mathbf{\Psi}\mathbf{s}^\star$ also makes progress on its quality as the number of measurement increases: the error bound follows (6) as well if $\mathbf{\Psi}$ is an orthogonal matrix, which is usually the case.

We now have the half-decoded matrix. Next, each column of the half-decoded matrix is decoded once more to obtain the full-decoded matrix. In contrast to the case of decoding each row, decoding each column follows the recovery procedure explained in Section II-C, because error occurs during the recovery of spatially sampled coefficients following the error bound in (6).

In particular, the solution $\mathbf{s}^\star$ to (5) obeys the following reconstruction error bound:

$$\|\mathbf{s}^\star - \mathbf{s}\|_2 \leq C_0 \cdot \|\mathbf{s} - \mathbf{s}_K\|_1 + C_1 \cdot \sqrt{K}\eta, \tag{7}$$

where C_1 is another constant for the additional term in the new error bound. Thus, (7) accounts for not only the measurement error due to an insufficient M, but the measurement error carried over from the previous recovery stage, which is explained by η in (5). Accordingly, the total error of our low complexity sensing framework is given by (7).

IV. Complexity Comparison with Popular Compressive Sensing Schemes

The complexity of a particular sensing scheme for each sensor can be broken down into three categories: computational complexities (time and space) and communication complexity. Here, the time complexity is the number of multiplication and summation operations involved with the projection operation in (2). The space complexity is the amount of storage required for the projection operation, which are pseudorandom numbers in the sensing matrix $\mathbf{\Phi}$ or merely random sampling indices. Lastly, the communication complexity is the number of random measurements that should be transmitted to the collection point. (We do not analyze the synchronization overhead of random numbers since it is negligible if random seeds are used for the synchronization.) It should be noted that these complexities are all related to the energy consumption of each sensing device, although the effects of each complexity on energy can vary depending on architectures.

Table I shows the three complexities involved with *individual* sensing devices, where N is the number of data in temporal dimension; J is the number of distributed sensors; K is the number of significant coefficients in temporal dimension; K' is the number of significant coefficients in spatial dimension; K'' is the number of significant coefficients shared across distributed sensors (joint sparsity).

We present five CS schemes in this paper to compare with our low complexity sensing framework, which are outlined as follows:

- Temporal CS: Each sensor takes random measurements in the temporal dimension and sends them to the collection point. In fact, this is a basic and natural way of applying CS to individual sensing devices.
- Temporal Random Sampling: Basically the same as Temporal CS, but employs the random sampling in the temporal dimension as opposed to Temporal CS where a dense sensing matrix $\mathbf{\Phi}$ is incorporated.
- Spatial CS: Random measurements are taken in the spatial dimension and hence suitable for the distributed sensing context. This group of sensing schemes is

usually coupled with specific routing schemes which enables various kinds of CS in the spatial dimension. Most literature in distributed sensing belongs to this group [2], [3].

- Spatial Random Sampling: Analogous to the relationship between Temporal CS and Temporal Random Sampling, and employs the random sampling in the spatial dimension instead of the dense random sensing.
- Model-Based CS: This particular scheme exploits the joint correlation of spatio-temporal dimension (block sparse model) [5]. This resembles Temporal CS in the sense that sensors need not cooperate with each other. On the contrary, total rate (compressed sensing data size) required for the decoder to recover the group of original signals is less than the case of Temporal CS, since this scheme also leverages *inter-signal correlations* as well as *intra-signal correlations* that is only taken into account in Temporal CS [8].

In Table I, time complexities of random sampling schemes including ours are constants because the random sampling does not involve multiplication and summation operations. (Rather, it corresponds to a table lookup.) On the contrary, other schemes all require notable amounts of operations. These differences lead to differences in the space complexities between these two groups: sensing schemes that employ dense sensing matrices have the space complexities the same as the time complexities, whereas random sampling schemes have the space complexities the same as the communication complexities. Specifically, random sampling schemes use random indices only for the sampling, in contrast to dense sensing schemes that should store entries for the random sensing matrices. These entries may occupy more space (e.g., four or eight bytes for a real number representation) than random sampling indices to guarantee an enough level of precision.

It is interesting to observe that in Table I, Temporal CS and Random Sampling are fundamentally identical to Spatial CS and Random Sampling. Suppose $K' = K$ and $J = N$: we obtain the same complexities between two groups. Thus it is more advantageous to select sensing schemes with larger numbers of data if we were to choose between temporal and spatial groups. Nonetheless, in reality, the temporal group often turns out to be more beneficial than the spatial group because (i) we can almost always have N larger than J if we wish, (ii) the intra-signal correlations are usually stronger than the inter-signal correlations.

Model-Based CS utilizes *common sparse supports* among the group of signal vectors in the temporal dimension: it assumes all data vectors in the temporal dimension randomly measured by sensors share the same K'' basis vectors when represented in a sparsifying basis $\mathbf{\Psi}$ [5], which is insufficient to be applied to real world signals since sparse supports can vary between different sensors. Therefore, one usually selects the union of individual supports to contain all of sparse supports from different sensors, which obviously makes $K'' > K$.

Furthermore, Model-Based CS has higher complexities than all other sensing schemes owing to the fact that it *maximally* utilizes the joint correlation by getting every sensor involved in measuring each other's sensing data [5]. From the perspective of individual sensing devices, these complexity burdens are undesirable especially when they are resource-limited. On the contrary, our Low Complexity Sensing is the least complex scheme in Table I, which can be intuitively appreciated by Fig. 2. Compared with Temporal Random Sampling and Spatial Random Sampling, we can further reduce both space and communication complexities by factors of $(K' \log J)/J$ and $(K \log N)/N$.

V. Experimental Results

We now compare the coding efficiency that accounts for how small the error between the original signal and the recovered signal can be with a compressed data size. In particular, we are interested in the coding efficiency per the measurements of *individual* sensors. We compared our Low Complexity Sensing with Spatial Random Sampling and Model-Based CS. Sensing in the spatial dimension is the most popular approach that utilizes compressive sensing (CS) [2], [3]; we adopted the random sampling instead of the dense random sensing. Model-Based CS is so far a state-of-the-art CS scheme that takes account of the joint correlation in the spatio-temporal dimension.

Since we are interested in real world signals, we employed environmental data sets downloaded from the SensorScope website that has various wireless sensor network deployment scenarios [11]. We employed (i) ambient temperature (°C) and (ii) relative humidity (%), which can be approximated with K basis vectors. In particular, we utilized DCT throughout the experiments as the sparsifying basis $\mathbf{\Psi}$. As discussed in Section III-B2, recovered signal quality is improved (i.e., the signal error is decreased) as more random measurements are sent by individual sensors.

Fig. 3 and 4 show the experimental results. We here consider sum of squared error (SSE) normalized with respect to the norm of signal as the performance metric. Spatial Random Sampling shows unsatisfactory results in both cases. This is mostly attributed to stronger intra-signal correlations inside each sensor than inter-signal correlations between sensors: sole consideration of the spatial dimension fails to capture significant correlations in the temporal dimension. On the contrary, our Low Complexity Sensing and Model-Based CS show superior results. In particular, Low Complexity Sensing outperforms Model-Based CS where M is small.

VI. Conclusion

We have proposed a low complexity sensing framework that can facilitate big data sensing with low computational

Table I
COMPLEXITY COMPARISON FOR INDIVIDUAL SENSORS

Sensing Schemes	Time	Space	Communication
Temporal CS	$O(NK\log(N/K))$	$O(NK\log(N/K))$	$O(K\log(N/K))$
Temporal Random Sampling	$O(1)$	$O(K\log N)$	$O(K\log N)$
Spatial CS	$O(NK'\log(J/K'))$	$O(NK'\log(J/K'))$	$O((N/J)K'\log(J/K'))$
Spatial Random Sampling	$O(1)$	$O((N/J)K'\log J)$	$O((N/J)K'\log J)$
Model-Based CS	$O(NJK''+NK''\log(N/K''))$	$O(NJK''+NK''\log(N/K''))$	$O(JK''+K''\log(N/K''))$
Low Complexity Sensing	$O(1)$	$O((KK'/J)\log N\log J)$	$O((KK'/J)\log N\log J)$

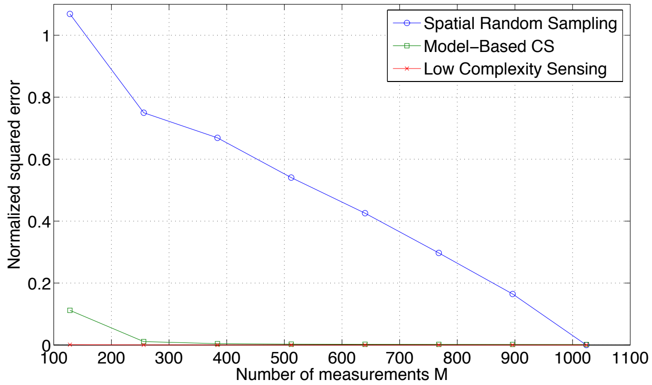

Figure 3. Normalized SSE comparison of ambient temperature signals from 32 sensors ($J = 32$) with the temporal data length $N = 1024$.

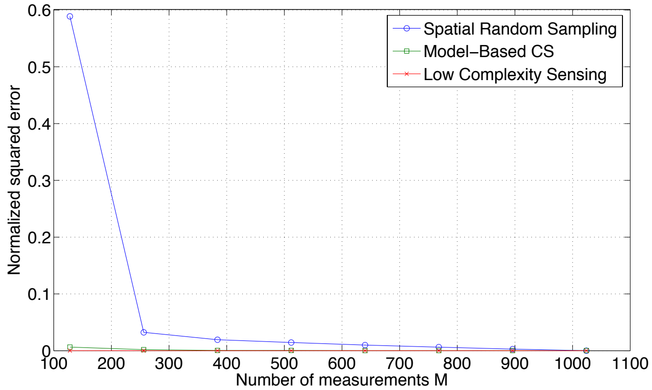

Figure 4. Normalized SSE comparison of relative humidity signals from 32 sensors ($J = 32$) with the temporal data length $N = 1024$.

and communication overheads. We compared the complexities of our sensing framework with other compressive sensing schemes in order to prove the low complexity characteristic of our scheme. In addition to the low complexity that meets the requirement of resource-limited sensors, our scheme showed the improved coding efficiency compared to other schemes.

ACKNOWLEDGMENT

This research was supported in part by Basic Science Research Program through the National Research Foundation of Korea (NRF) grant funded by the Ministry of Science, ICT & Future Planning (NRF-2014R1A1A1002662) and in part by the 2013 Research Fund (1.130056.01) of UNIST (Ulsan National Institute of Science and Technology).

REFERENCES

[1] D. Noh, D. Lee, and H. Shin, “QoS-aware geographic routing for solar-powered wireless sensor networks,” *IEICE Trans. Commun.*, vol. 90, no. 12, pp. 3373–3382, Dec. 2007.

[2] C. Luo, F. Wu, J. Sun, and C. W. Chen, “Compressive data gathering for large-scale wireless sensor networks,” in *Proc. MobiCom ’09*, 2009, pp. 145–156.

[3] G. Quer, R. Masiero, D. Munaretto, M. Rossi, J. Widmer, and M. Zorzi, “On the interplay between routing and signal representation for compressive sensing in wireless sensor networks,” in *Proc. Inf. Theory Appl. Workshop (ITA ’09)*, 2009, pp. 206–215.

[4] S. Foucart and H. Rauhut, *A Mathematical Introduction to Compressive Sensing*. Springer, 2013.

[5] R. G. Baraniuk, V. Cevher, M. F. Duarte, and C. Hegde, “Model-based compressive sensing,” *IEEE Trans. Inf. Theory*, vol. 56, no. 4, pp. 1982–2001, Apr. 2010.

[6] E. J. Candès and M. B. Wakin, “An introduction to compressive sampling,” *IEEE Signal Process. Mag.*, vol. 25, no. 2, pp. 21–30, Mar. 2008.

[7] T. T. Do, L. Gan, N. H. Nguyen, and T. D. Tran, “Fast and efficient compressive sensing using structurally random matrices,” *IEEE Trans. Signal Process.*, vol. 60, no. 1, pp. 139–154, Jan. 2012.

[8] D. Baron, M. F. Duarte, M. B. Wakin, S. Sarvotham, and R. G. Baraniuk, “Distributed compressive sensing,” *arXiv preprint arXiv:0901.3403*, 2009.

[9] D. Lee, J. Choi, and H. Shin, “Low-complexity compressive sensing with downsampling,” *IEICE Electron. Express*, vol. 11, no. 3, pp. 20 130 947–20 130 947, Feb. 2014.

[10] J. A. Burke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Reddy, and M. B. Srivastava, “Participatory sensing,” *Center for Embedded Network Sensing*, 2006.

[11] Sensorscope: Sensor networks for environmental monitoring. [Online]. Available: http://lcav.epfl.ch/op/edit/sensorscope-en