Compressed Sensing for Networked Data

Jarvis Haupt, Waheed U. Bajwa, Michael Rabbat, and Robert Nowak

I. Introduction

Imagine a system with thousands or millions of independent components, all capable of generating and communicating data. A man-made system of this complexity was unthinkable a few decades ago, but today it is a reality – computers, cell phones, sensors, and actuators are all linked to the Internet, and every wired or wireless device is capable of generating and disseminating prodigious volumes of data. This system is not a single centrally-controlled device, rather it is an ever-growing patchwork of autonomous systems and components, perhaps more organic in nature than any human artifact that has come before. And we struggle to manage and understand this creation, which in many ways has taken on a life of its own. Indeed, several international conferences are dedicated to the scientific study of emergent Internet phenomena.

This article considers a particularly salient aspect of this struggle that revolves around large-scale distributed sources of data and their storage, transmission, and retrieval. The task of transmitting information from one point to another is a common and well-understood exercise. But the problem of efficiently transmitting or sharing information from and among a vast number of distributed nodes remains a great challenge, primarily because we do not yet have well developed theories and tools for distributed signal processing, communications, and information theory in large-scale networked systems.

The problem is illustrated by a simple example. Consider a network of n nodes, each having a piece of information or data x_j, $j = 1, \ldots, n$. These data could be files to be shared, or simply scalar values corresponding to node attributes or sensor measurements. Let us assume that each x_j is a scalar quantity for the sake of this illustration. Collectively these data $\mathbf{x} = [x_1, \ldots, x_n]^T$, arranged in a vector, are called *networked data* to emphasize both the distributed nature of the data and the fact that they may be shared over the underlying communications infrastructure of the network. The networked data vector may be very large; n may be a thousand or a million or more.

Thus, even the process of gathering $\mathbf{x}$ at a single point is daunting (requiring n communications at least), and yet this global sense of the networked data is crucial in applications ranging from network security to wireless sensing. Suppose, however, that it is possible to construct a highly compressed version of $\mathbf{x}$, efficiently and in a decentralized fashion. This would offer many obvious benefits, provided that the compressed version could be processed to recover $\mathbf{x}$ to within a reasonable accuracy.

There are several decentralized compression strategies that could be utilized. One possibility is that the correlations between data at different nodes are known a priori. Then distributed source coding techniques, such as Slepian-Wolf coding, can be used to design compression schemes without collaboration between nodes (see [1] and the references therein for an excellent overview of such approaches). Unfortunately, in many applications prior knowledge of the precise correlations in the data is unavailable, making it difficult or impossible to apply such distributed source coding techniques. This situation motivates collaborative, in-network processing and compression, where unknown correlations and dependencies between the networked data can be learned and exploited by exchanging information between network nodes. But, the design and implementation of effective collaborative processing algorithms can be quite challenging, since they too rely on some prior knowledge of the anticipated correlations and depend on somewhat sophisticated communications and node processing capabilities.

This article describes a very different approach to the decentralized compression of networked data. Specifically, consider a compression of the form $\mathbf{y} = \mathbf{A}\mathbf{x}$, where $\mathbf{A} = \{A_{i,j}\}$ is a $k \times n$ "sensing" matrix with far fewer rows than columns (i.e., $k \ll n$). The compressed data vector $\mathbf{y}$ is $k \times 1$, and therefore it is much easier to store, transmit, and retrieve compared to the uncompressed networked data $\mathbf{x}$. The theory of compressed sensing guarantees that, for certain matrices $\mathbf{A}$, which are non-adaptive and often quite unstructured, $\mathbf{x}$ can be accurately recovered from $\mathbf{y}$ whenever $\mathbf{x}$ itself is compressible in some domain (e.g., frequency, wavelet, time) [2]–[5].

To carry the illustration further, and to motivate the approaches proposed in this article, let us look at a very concrete example. Suppose that *most* of the network nodes have the same nominal data value, but the few remaining nodes have different values. For instance, the values could correspond to security statistics or sensor readings at each node. The networked data vector in this case is mostly constant, except for a few deviations in certain locations, and it is the minority that may be of most interest in security or sensing applications. Clearly $\mathbf{x}$ is quite compressible; the nominal value plus the locations and values of the few deviant cases suffice for its specification.

Consider a few possible situations in this networked data compression problem. First, if the nominal value were known to all nodes, than the desired compression is accomplished simply by the deviant nodes sending a notification of such. Second, if the nominal value were not known, but the deviant cases were assumed to be isolated, then the nodes could simply compare their own values to those of their nearest neighbors to determine the nominal value and any deviation of their own. Again, notifications from the deviant nodes would provide the desired compression. There is a third, more general, scenario in which such simple local processing schemes can break down. Suppose that the nominal value is unknown to the nodes a priori, and that the deviant cases could be isolated or clustered. Since the deviant nodes may be clustered together, simply comparing values between neighboring nodes may not reveal them all, and perhaps not even the majority of them depending on the extent of clustering. Indeed, distributed processing schemes in general are difficult to design without prior knowledge of the anticipated relations among data at neighboring nodes. This serves as a motivation for the theory and methods discussed here.

Compressed sensing offers an alternative measurement approach that does not require any specific prior signal knowledge and is an effective (and efficient) strategy in each of the situations described above. The values of all nodes can be recovered from the compressed data $\mathbf{y} = \mathbf{A}\mathbf{x}$, provided its size k is proportional to the number of deviant nodes. As we shall see, $\mathbf{y}$ can be efficiently computed in a distributed manner, and by virtue of its small size, it is naturally easy to store and transmit. In fact, in certain wireless network applications (see Section IV-B for details), it is even possible to compute $\mathbf{y}$ in the air itself, rather than in silicon! Thus, compressed sensing offers two highly desirable features for networked data analysis. The method is *decentralized*, meaning that distributed data can be encoded without a central controller, and *universal*, in the sense that sampling does not require a priori knowledge or assumptions about the data. For these reasons, the advantages of compressed sensing have already caught on in the research community, as evidenced by several recent works [6]–[10].

II. Compressed Sensing Basics

The essential purpose of sensing and sampling systems is to accurately capture the salient information in a signal of interest. Generically, such systems can be viewed as having the following core components. First, in a preconditioning step, the system introduces some form of sensing diversity, which gives each physically distinct signal from a specified class of candidates a distinct signature or fingerprint. Next, the "diversified" signal is sampled and recorded, and finally the system reconstructs the original signal from the sampled data. Because inadequate sampling of a signal can induce aliasing, meaning that the same set of samples may describe many different signals, the preconditioning step is necessary to eliminate spurious (incorrect) solutions. For example, low-pass filtering is a type of preconditioning that maps every signal having frequency less than the filter cutoff frequency to itself, while all higher frequency components are mapped to zero, and this step is sufficient to ensure that the signal reconstructed from a set of uniform samples is unique and equal to the original signal.

The theory of compressed sensing (CS) extends traditional sensing and sampling systems (designed with bandlimited signals in mind) to a much broader class of signals. According to CS theory, *any* sufficiently compressible signal can be accurately recovered from a small number of non-adaptive, randomized linear projection samples. For example, suppose that $\mathbf{x} \in \mathbb{R}^n$ is m-sparse (i.e., it has no more than m nonzero entries) where m is much smaller than the signal length n. Sparse vectors are very compressible, since they can be completely described by the locations and amplitudes of the non-zero entries. Rather than sampling each element of $\mathbf{x}$, CS directs us to first precondition the signal by operating on it with a diversifying matrix, yielding a signal whose entries are mixtures of the non-zero entries of the original signal. The resulting signal is then sampled k times to obtain a low-dimensional vector of observations. Overall, the acquisition process can be described by the observation model $\mathbf{y} = \mathbf{A}\mathbf{x} + \boldsymbol{\epsilon}$, where the matrix $\mathbf{A}$ is a $k \times n$ CS matrix that describes the joint operations of diversification and subsampling, and $\boldsymbol{\epsilon}$ represents errors due to noise or other perturbations. The main results of CS theory have established that if the number of CS samples is a small integer multiple greater than the number of non-zero entries in $\mathbf{x}$, then these samples sufficiently "encode" the salient information in the sparse signal and an accurate reconstruction from $\mathbf{y}$ is possible. These results are very promising because at least $2m$ pieces of information (the location and amplitude of each nonzero entry) are generally required to describe any m-sparse signal, and CS is an effective way to obtain this information in a simple, non-adaptive manner. The next few subsections explain, in some detail, how this is accomplished.

Compressed Sensing for Networked Data

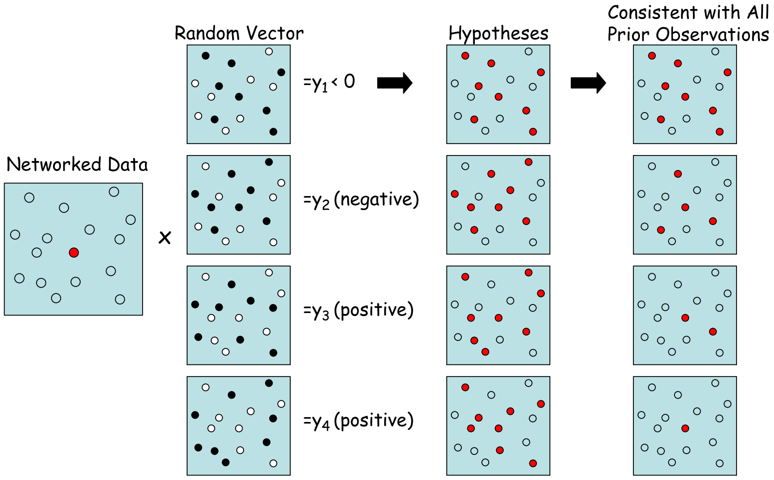


Fig. 1. A simple reconstruction example on a network of $n = 16$ nodes. One distinguished sensor observes a positive value while the remaining $n-1$ observe zero. The task is to identify which sensor is different using as few observations as possible. One effective approach is to project the data onto random vectors, as depicted in the second column (where nodes indicated in black multiply their data value by -1 and those in white by $+1$). The third column shows that about $n/2$ hypothesis sensors are consistent with each random projection observation, but the number of hypotheses that are simultaneously consistent with *all* observations (shown in the fourth column) decreases exponentially with the number of observations. The random projection observations are approximately performing binary bisections of the hypothesis space, and only about $\log n$ observations are needed to determine which sensor reads the non-zero value.

To illustrate the CS random projection encoding and reconstruction ideas, consider a simplification of the example described in the introduction. Suppose that in a network of n sensors, only one of the sensors is observing some positive value, while the rest of the sensors observe zero. The goal is to identify which sensor is different using a minimum number of observations. Consider making random projection observations of the data, where each observation is the projection of the sensor readings onto a random vector having entries ± 1 each with probability $1/2$. The value of each observation, along with knowledge of the random vector onto which the data was projected, can be used to identify a set of about $n/2$ hypothesis sensors that are consistent with that particular observation. The estimate of the anomalous sensor given k observations is simply the intersection of the k hypotheses sets defined by the observations. It is easy to see that, on average, about $\log n$ observations are required before the correct (unique) estimate is obtained. Define the ℓ_0 quasi-norm $\|\mathbf{z}\|_0$ to be equal to the number of nonzero entries in the vector $\mathbf{z}$. Then this simple procedure can be thought of as the solution of the optimization problem

$$\arg\min_{\mathbf{z}} \|\mathbf{z}\|_0 \text{ subject to } \mathbf{y} = \mathbf{A}\mathbf{z}. \tag{1}$$

A. Encoding Requirements

Suppose that for some observation matrix $\mathbf{A}$ there is a nonzero m-sparse signal $\mathbf{x}$ such that the observations $\mathbf{y} = \mathbf{A}\mathbf{x} = \mathbf{0}$. One could not possibly hope to recover $\mathbf{x}$ in this setting, since the observations do not provide any information about the signal. Another similar problem arises if two distinct m-sparse signals, say $\mathbf{x}$ and $\mathbf{x}'$, are mapped to the same compressed data (i.e., $\mathbf{A}\mathbf{x} = \mathbf{A}\mathbf{x}'$). These two scenarios describe situations where certain sparse vectors lie in the null space of the observation matrix.

Matrices that are resilient to these ambiguities are those that satisfy the Restricted Isometry Property (RIP), sometimes also called the Uniform Uncertainty Principle (UUP) [2], [11]. Formally, a $k \times n$ sensing matrix with unit-norm rows (i.e., $\sum_{j=1}^{n} A_{i,j}^2 = 1$ for $i = 1, 2, \ldots, k$) is said to satisfy a RIP of order s whenever

$$(1-\delta_s)\frac{k}{n}\|\mathbf{x}\|_2^2 \leq \|\mathbf{A}\mathbf{x}\|_2^2 \leq (1+\delta_s)\frac{k}{n}\|\mathbf{x}\|_2^2 \tag{2}$$

holds simultaneously for all s-sparse vectors $\mathbf{x} \in \mathbb{R}^n$ for sufficiently small values of δ_s. The RIP is so-named because it describes matrices that impose a near-isometry (approximate length preservation) on a restricted set of subspaces (the subspaces of s-sparse vectors).

In practice, sensing matrices that satisfy the RIP are easy to generate. It has been established that $k \times n$ matrices whose entries are independent and identically distributed realizations of certain zero-mean random variables with variance $1/n$ satisfy a RIP with very high probability when $k \geq const \cdot \log n \cdot m$ [2], [3], [12]. Physical limitations of real sensing systems motivate the unit-norm restriction on the rows of $\mathbf{A}$, which essentially limits the amount of "sampling energy" allotted to each observation.

B. Decoding: Algorithms and Bounds

Because compressed sensing is a form of subsampling, aliasing is present and needs to be accounted for in the reconstruction process. The same compressed data could be generated by many n-dimensional vectors, but the RIP implies that only one of these is sparse. This might seem to require that any reconstruction algorithm must exhaustively search over all sparse vectors, but fortunately the process is much more tractable. Given a vector of (noise-free) observations $\mathbf{y} = \mathbf{A}\mathbf{x}$, the unknown m-sparse signal $\mathbf{x}$ can recovered exactly as the unique solution to

$$\arg\min_{\mathbf{z}} \|\mathbf{z}\|_1 \text{ subject to } \mathbf{y} = \mathbf{A}\mathbf{z}, \tag{3}$$

where $\|\mathbf{z}\|_1 = \sum_{i=1}^{n} |z_i|$ denotes the ℓ_1-norm, provided the restricted isometry constants satisfy $\delta_m + \delta_{2m} + \delta_{3m} < 1$, which is a slightly stronger condition than necessary to ensure that neither of the encoding ambiguities described earlier can happen [2]. The recovery procedure can be cast as a linear program, so it is very easy to solve even when n is very large.

Compressed sensing remains quite effective even when the samples are corrupted by additive noise, which is important from a practical point of view since any real system will be subjected to measurement inaccuracies. A variety of reconstruction methods have been proposed to recover (an approximation of) $\mathbf{x}$ when observations are corrupted by noise. For example, estimates $\widehat{\mathbf{x}}$ can be obtained as the solutions of either

$$\arg\min_{\mathbf{z}} \|\mathbf{z}\|_1 \text{ subject to } \|\mathbf{A}^T(\mathbf{y} - \mathbf{A}\mathbf{z})\|_\infty \leq \lambda_1, \tag{4}$$

where $\|\mathbf{z}\|_\infty = \max_{i=1,\ldots,n} |\mathbf{z}(i)|$ [5], or the penalized least squares minimization

$$\arg\min_{\mathbf{z}} \left\{ \|\mathbf{y} - \mathbf{A}\mathbf{z}\|_2^2 + \lambda_2 \|\mathbf{z}\|_0 \right\} \tag{5}$$

as proposed in [4], for appropriately chosen regularization constants λ_1 and λ_2 that each depend on the noise variance. In either case, the reconstruction satisfies

$$\mathbb{E}\left[\frac{\|\mathbf{x} - \widehat{\mathbf{x}}\|_2^2}{n}\right] \leq const \cdot \left(\frac{k}{m \log n}\right)^{-1}, \tag{6}$$

where the leading constant does not depend on k, m or n. In practice, the optimization (4) can be solved by a linear program, while (5) is often solved by convex relaxation – replacing the ℓ_0

penalty with the ℓ_1 penalty.

The appeal of CS is readily apparent from the error bound in (6) which (ignoring the constant and logarithmic factors) is proportional to m/k, the variance of an estimator of m parameters from k observations. In other words, CS is able to both identify the locations and estimate the amplitudes of the non-zero entries without any specific prior knowledge about the signal except its assumed sparsity. For this reason CS is often referred to as a universal approach, since it can effectively recover *any* sufficiently sparse signal from a set of nonadaptive samples.

C. Transform Domain Sparsity

Suppose the observed signal $\mathbf{x}$ is not sparse, but instead a suitably transformed version of it is. That is, if $\mathbf{T}$ is a transformation matrix then $\boldsymbol{\theta} = \mathbf{T}\mathbf{x}$ is sparse. The CS observations can be written as $\mathbf{y} = \mathbf{A}\mathbf{x} = \mathbf{A}\mathbf{T}^{-1}\boldsymbol{\theta}$, and if $\mathbf{A}$ is a random CS matrix satisfying the RIP, then in many cases so is the product matrix $\mathbf{A}\mathbf{T}^{-1}$ [12]. Consequently, CS does not require prior knowledge or assumptions regarding the domain in which the networked data are compressible, again highlighting its universality.

The sparse vector $\boldsymbol{\theta}$ (and hence $\mathbf{x}$) can be accurately recovered from $\mathbf{y}$ using the reconstruction techniques described above. For example, in the noiseless setting one can solve

$$\widehat{\boldsymbol{\theta}} = \arg\min_{\mathbf{z}} \|\mathbf{z}\|_1 \text{ subject to } \mathbf{y} = \mathbf{A}\mathbf{T}^{-1}\mathbf{z}, \tag{7}$$

to obtain an exact reconstruction of the transform coefficients of $\mathbf{x}$. Note that, while the samples do not require selection of an appropriate sparsifying transform, the reconstruction does.

In other settings signals of interest may not be exactly sparse, but instead most of the energy might be concentrated on a relatively small set of entries while the remaining entries are very small. The degree of effective sparsity of such signals can be quantified with respect to a given basis. Formally, for a signal $\mathbf{x}$ let $\mathbf{x}^s$ be the approximation of $\mathbf{x}$ formed by retaining the s coefficients having largest magnitude in the transformed representation $\boldsymbol{\theta} = \mathbf{T}\mathbf{x}$. Then $\mathbf{x}$ is called α-compressible if the approximation error obeys

$$\frac{\|\mathbf{x} - \mathbf{x}^s\|_2^2}{n} \leq const \cdot s^{-2\alpha} \tag{8}$$

for some $\alpha = \alpha(\mathbf{x}, \mathbf{T}) > 0$. This model could describe, for example, signals whose ordered (transformed) coefficient amplitudes exhibit a power-law decay. Such behavior is associated with images that are smooth or have bounded variation [3], [11], and is often observed in the wavelet coefficients of natural images. In this setting, CS reconstruction techniques can again be applied to obtain an estimate of the transformed coefficients directly. For example, the estimate $\widehat{\mathbf{x}} = \mathbf{T}^{-1}\widehat{\boldsymbol{\theta}}$, obtained by solving

$$\widehat{\boldsymbol{\theta}} = \arg\min_{\mathbf{z}} \left\{ \|\mathbf{y} - \mathbf{A}\mathbf{T}^{-1}\mathbf{z}\|_2^2 + \lambda \|\mathbf{z}\|_0 \right\}, \tag{9}$$

satisfies

$$\mathbb{E}\left[\frac{\|\mathbf{x} - \widehat{\mathbf{x}}\|_2^2}{n}\right] \leq const \cdot \left(\frac{k}{\log n}\right)^{-2\alpha/2\alpha+1}, \tag{10}$$

which quantifies the simultaneous balancing of the errors due to approximation and estimation [4]. The result guarantees that even when signals are only approximately sparse, consistent estimation is still possible.

III. Sparsifying Networked Data

Compressed sensing can be very effective when $\mathbf{x}$ is sparse or highly compressible in a certain basis or dictionary. But, while transform-based compression is well-developed in traditional signal and image processing domains, the understanding of sparsifying/compressing bases for networked data is far from complete. There are, however, a few promising new approaches to the design of transforms for networked data. It is natural to associate a graph with a given network, where the vertices of the graph represent the nodes of the network, and edges between vertices represent anticipated relationships among the data at adjacent nodes. The edges may reflect relationships due to communication links or correlations and dependencies that are anticipated between data at neighboring nodes. Exploiting the structure of the connectivity is the key to obtaining effective sparsifying transformations for networked data, and a few methods are described below.

A. Spatial Compression

Suppose a wireless sensor network is deployed to monitor a certain spatially-varying phenomenon such as temperature, light, or moisture. The physical field being measured can be viewed as a signal or image with a degree of spatial correlation or smoothness. If the sensors are geographically placed in a uniform fashion, then sparsifying transforms may be readily borrowed from traditional signal processing. Figure 2(a) illustrates a typical such situation where the underlying graph is a regular lattice. In these settings, the sensor locations can be viewed as *sampling locations* and tools like the Discrete Fourier Transform (DFT) or Discrete Wavelet Transform (DWT) may be used to decorrelate and sparsify the sensor data. In more general settings, wavelet techniques can be extended to also handle the irregular distribution of sampling locations [13].

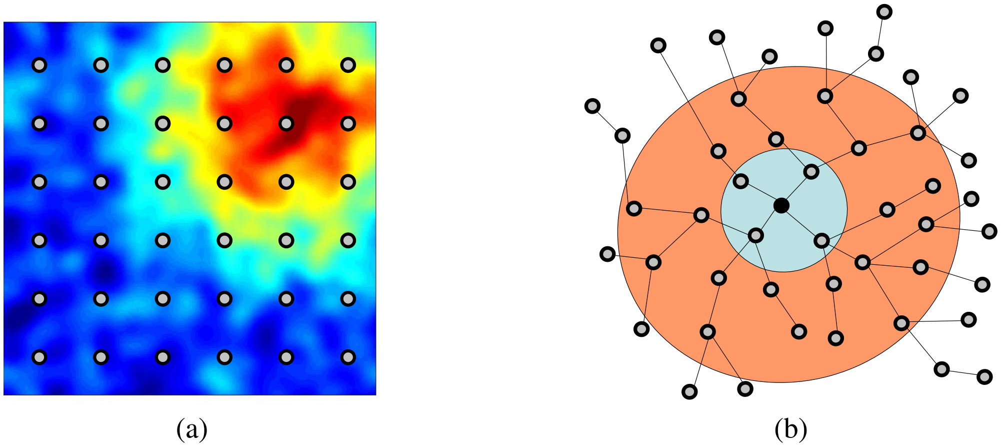


Fig. 2. Sparsifying transformation techniques depend on graph topologies. The smoothly varying field in (a) is monitored by a network of wireless sensors deployed uniformly over the region, and standard transform techniques can be used to sparsify the networked data. For more abstract graph topologies, graph wavelets can be effective. In (b), the graph (Haar) wavelet coefficient at the location of the black node and scale three is given by the difference of the average data values at the nodes in the red and blue regions.

B. Graph Wavelets

Standard signal transforms cannot be applied in more general situations. For example, many network monitoring applications rely on the analysis of communication traffic levels at the network nodes. Changes in the behavior of traffic levels can be indicative of variations in network usage, component failures or misconfigurations, or malicious activities. There are strong correlations between traffic levels at different nodes, but the topology and routing affect the nature of these relationships in complex ways. Moreover, since network topology is rarely based on a regular lattice, the graphs needed to represent such networks can be quite complex as well. Graph wavelets, developed with these challenges in mind, adapt the design principles of the DWT to arbitrary graphs [14].

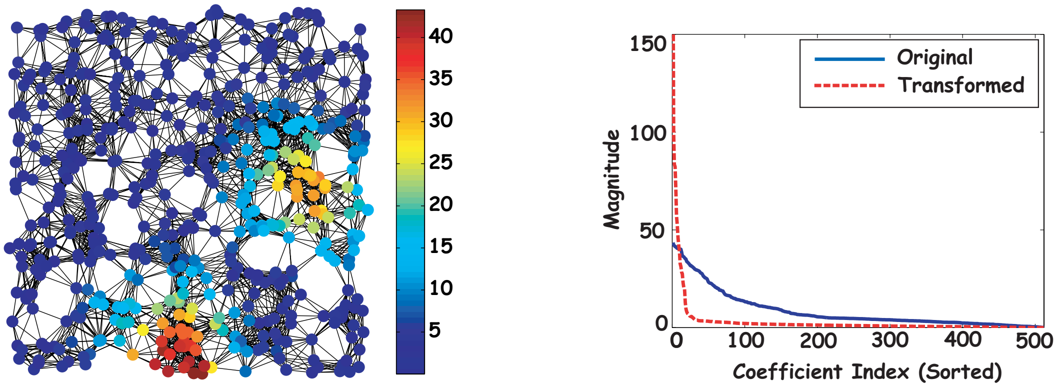
To understand the construction of graph wavelets, it is useful to first consider the Haar wavelet transform, which is the simplest form of DWT. The wavelet coefficients are essentially obtained as digital differences of the data at different scales of aggregation. The coefficients at the first scale are differences between neighboring data points, and those at subsequent spatial scales are computed by first aggregating data in neighborhoods (dyadic intervals in one dimension and square regions in two dimensions) and then computing differences between neighboring aggregations. Other versions of the DWT are distinguished by more general aggregation/averaging and differencing operations.

Graph wavelets are a generalization of this construction, where the number of hops between nodes in a network provides a natural distance measure that can be used to define neighborhoods. The size of each neighborhood (with radius defined by the number of hops) provides a natural measure of scale, with smaller sizes corresponding to finer spatial analysis of the networked data. Graph wavelet coefficients are then defined by aggregating data at different scales, and computing differences between aggregated data, as shown in Figure 2(b). Further details and generalizations, along with an application of graph wavelets to the analysis of network traffic data, may be found in [14].

C. Diffusion Wavelets

Diffusion wavelets provide an alternative approach to constructing a multi-scale representation for data defined on a graph. Unlike graph wavelets which produce an overcomplete dictionary, diffusion wavelets construct an orthonormal basis for functions supported on a graph. The diffusion wavelet construction process produces a basis tailored to a specific graph by analyzing eigenvectors of a diffusion matrix derived from the graph adjacency matrix (hence the name “diffusion wavelets”). The resulting basis vectors are generally localized to neighborhoods of varying size and may also lead to a sparsifying representation of data on a graph. A thorough treatment of this topic can be found in [15].

One example of sparsification using diffusion wavelets is shown in Fig. 3, where the node data correspond to traffic rates through routers in a computer network. There are several highly-localized regions of activity, while most of the remaining network exhibits only moderate levels of traffic. The traffic data are sparsely represented in the diffusion wavelet basis, and a small number of coefficients can provide an accurate estimate of the actual traffic patterns.



(a) Local traffic rates

(b) Ordered coefficients

Fig. 3. An illustration of the compressibility of spatially correlated networked data using diffusion wavelets. The actual networked data shown in (a) are not sparse, but can be represented with a small number of diffusion wavelet coefficients, as seen in (b).

IV. Networked Data Compression in Action

This section describes two techniques for obtaining projections of networked data onto general vectors, which can be thought of as the rows of the sensing matrix $\mathbf{A}$. As described in Section II, random projections are a useful choice when the underlying data is sparse, since consistent estimation is possible without prior knowledge of the sparsifying (or compressing) basis or representation. In addition, a variety of methods exist to sparsify data on arbitrary networks, as outlined in Section III.

The first approach described below assumes that the network is any general multi-hop network. This model could explain, for example, wireless sensor networks, wired local area networks, weather or agricultural monitoring networks, or even portions of the Internet. In the multi-hop setting the projections can be computed and delivered to every subset of nodes in the network using gossip/consensus techniques, or they might be delivered to a single point using clustering and aggregation. The second more specific approach described below is motivated by many wireless sensor networks where explicit routing information is difficult to obtain and maintain. In this setting, each sensor instead contributes its measurement in a joint fashion by simultaneous transmission to a distant processing location, and the observations are accumulated and processed at that (single) destination point.

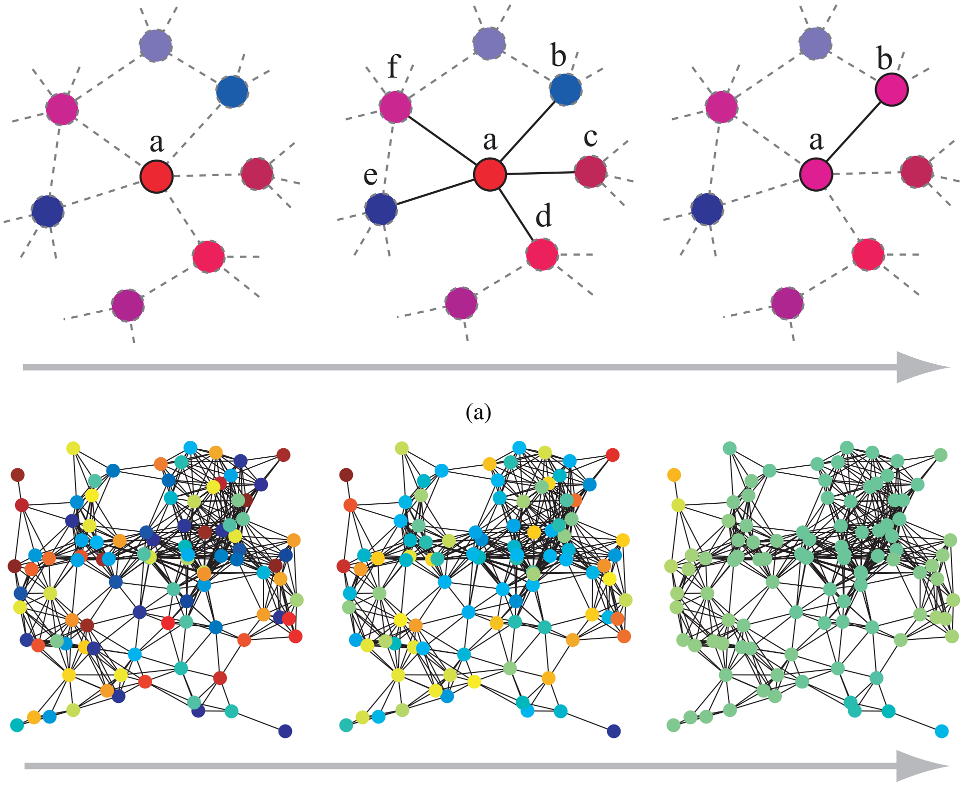
A. *Compressed Sensing for Networked Data Storage and Retrieval*

In general multi-hop networks, CS projections of the form $y_i = \sum_{j=1}^{n} A_{i,j} x_j$ can be computed in an efficient decentralized fashion because each compressed data value y_i is a simple linear combination of the values at each node. Two simple steps are required for the computation and distribution of each CS sample y_i, $i = 1, \ldots, k$:

Step 1: Each of the n sensors, $j = 1, \ldots, n$, locally computes the term $A_{i,j} x_j$ by multiplying its data with the corresponding element of the compressing matrix. The compressing matrix can be generated in a distributed fashion by letting each node locally generate a realization of $A_{i,j}$ using a pseudo-random number generator seeded with its identifier (in this example, the integers $j = 1, \ldots, n$ serve as this identifier). Given the identifiers of the nodes in the network, the requesting node can also easily reconstruct the random vectors $\{A_{i,j}\}_{i=1}^{k}$ for each sensor $j = 1, \ldots, n$.

Step 2: The local terms $A_{i,j} x_j$ are simultaneously aggregated and distributed across the network using randomized gossip, which is a simple iterative decentralized algorithm for computing linear functions such as $y_i = \sum_{j=1}^{n} A_{i,j} x_j$ (see Fig. 4). Because each node only exchanges information with its immediate neighbors in the network, gossip algorithms are resilient to failures or changes in the network topology. Moreover, when randomized gossip terminates, the value of y_i is available at every node in the network, so the network data cannot be compromised by eliminating a single server or fusion center.

In many scenarios, gossip algorithms are efficient since they use network resources to simultaneously route *and* compute information. For example, in networks with power-law degree distributions, such as the Internet, an optimized gossip algorithm uses on the order of kn transmissions to compute all of the samples [16]. Generally $k \ll n$, so this is much more efficient than exhaustively exchanging raw data values which would take about n^2 transmissions. In addition, the gossip procedure ensures that the samples are known at every node, so a user can query any node in the network, request the compressed data values, and compute $\widehat{\mathbf{x}}$ via one of the reconstruction methods outlined in Section II. Of course, one could replace gossip computation with aggregation up a spanning tree or around a cycle, if the network provides reliable routing service. This may be more efficient if it is known ahead of time that the compressed data values



(a)

(b)

Fig. 4. **Randomized Gossip:** The top panel depicts one gossip iteration, where the color of a node corresponds to its local value. To begin, the network is initialized to a state where each node has a value $x_i(0)$, $i = 1, \ldots, n$. Then in an iterative, asynchronous fashion, a random node a is "activated" and chooses one of its neighbors b at random. The two nodes then "gossip" – they exchange their values $x_a(l)$ and $x_b(l)$, or in the CS setting the values multiplied by pseudo-random compression vector elements, and perform the update $x_a(l+1) = x_b(l+1) \leftarrow \big(x_a(l) + x_b(l)\big)/2$, while the data at all the other nodes remains unchanged. The bottom panel shows an example network of 100 nodes with (left) random initial values, (middle) after each node has communicated five times with each of its neighbors, and (left) after each node has communicated 50 times with each of its neighbors. It can be shown that for this simple procedure, $x_i(l)$ converges to the average of the initial values, $1/n \sum_{j=1}^{n} x_j(0)$, at every node in the network as l tends to infinity as long as the random choice of neighbors is sufficient to ensure that information will eventually propagate between every pair of nodes.

will only be retrieved at one location. For more on using gossip algorithms to compute and distribute compressed data representations in multi-hop networks, see [7].

B. *Compressed Sensing in Wireless Sensor Networks*

Sensor networking is an emerging technology that promises an unprecedented ability to monitor the physical world via a spatially distributed network of small, inexpensive wireless devices that have the ability to self-organize into a well-connected network. A typical wireless sensor network, as shown in Fig. 5, consists of a large number of wireless sensor nodes, spatially distributed over

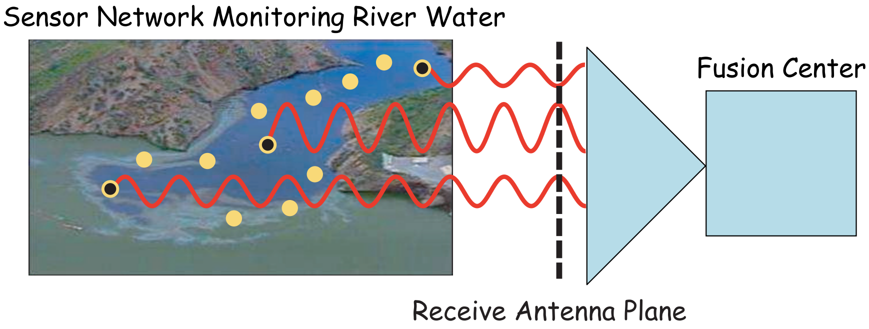


Fig. 5. An illustration of a wireless sensor network and fusion center. A number of sensor nodes monitor the river water for various forms of contamination and periodically report their findings over the air to the fusion center. CS projection observations are obtained by each sensor transmitting a sinusoid with amplitude given by the product of the sensor measurement and a pseudo-random weight. When the transmissions arrive in phase at the fusion center, the amplitude of the resulting received waveform is the sum of the component wave amplitudes.

a region of interest, that can sense (and potentially actuate) the physical environment in a variety of modalities, including acoustic, seismic, thermal, and infrared. A wide range of applications of sensor networks are being envisioned in a number of areas, including geographical monitoring, inventory management, homeland security, and health care.

The essential task in many applications of sensor networks is to extract some relevant information from distributed data and wirelessly deliver it to a distant destination, called the fusion center (FC). While this task can be accomplished in a number of ways, one particularly attractive technique leverages the theory of CS and corresponds to delivering random projections of the sensor network data to the FC by exploiting recent results on uncoded (analog) coherent transmission schemes in wireless sensor networks [17]–[20]. The proposed distributed communication architecture – introduced in [6], [8] and refined in [21] – requires only one (network) transmission per random projection and is based on the notion of so-called "matched source-channel communication" [19], [20]. Here, the CS projection observations are simultaneously calculated (by the superposition of radio waves) and communicated using amplitude-modulated coherent transmissions of randomly weighted sensed values directly from the nodes in the network to the FC via the air interface. Algorithmically, sensor nodes sequentially perform the following steps in order to communicate k random projections of the sensor network data to the FC:

Step 1: Each of the n sensors locally draws k elements of the random projection vectors $\{A_{i,j}\}_{i=1}^{k}$ by using its network address as the seed of a pseudo-random number generator. Given the seed values and the addresses of the nodes in the network, the FC can also easily reconstruct the random vectors $\{A_{i,j}\}_{i=1}^{k}$ for each sensor $j = 1, \ldots, n$.

Step 2: The sensor at location j multiplies its measurement x_j with $\{A_{i,j}\}_{i=1}^{k}$ to obtain a k-tuple

$$t_j \;=\; (A_{1,j}\, x_j\,, \ldots,\, A_{k,j}\, x_j)^T\,, \quad j = 1, \ldots, n, \tag{11}$$

and all the nodes coherently transmit their respective t_j's in an analog fashion over the network-to-FC air interface using k time-slots (transmissions). Because of the additive nature of radio waves, the corresponding received signal at the FC at the end of the k-th transmission is given by

$$\mathbf{y} \;=\; \sum_{j=1}^{n} t_j \;+\; \boldsymbol{\epsilon} \;=\; \mathbf{A}\mathbf{x} + \boldsymbol{\epsilon}, \tag{12}$$

where $\boldsymbol{\epsilon}$ is the noise generated by the communication receiver circuitry of the FC.

The steps above correspond to a completely decentralized way of delivering k random projections of the sensed data to the FC by employing k (network) transmissions. Another possibility for realizing the same goal is to assume that the sensors are capable of local communications and that a route which forms a spanning tree through the network to some nominated clusterhead has been established. Then, each sensor node can locally compute $\{t_{i,j} = A_{i,j} x_j\}_{i=1}^{k}$ and these values can be aggregated up the tree to obtain $\mathbf{t} = \mathbf{A}\mathbf{x}$ at the clusterhead which then encodes and transmits this $\mathbf{t}$ to the FC. The main difference here is that the wireless method described above can be implemented *without* any complex routing information and as a result might be a more suitable and scalable option in many sensor networking applications.

Digital vs. Analog Communications: Which is better?

It has been long known in the communications research community that digital transmissions are not always the best option in all communication scenarios, and often the performance of analog communications in network settings can far surpass that of digital communications (see [22] for an excellent tutorial review). As a simple illustration of why amplitude modulated, analog transmissions are well-suited for the problem of communicating random projections of the sensor network data to the FC, consider the following toy example:

Suppose two nodes A and B sense values 0 and 1, respectively and they need to communicate (in a distributed manner) the average of their sensed data to node C. Using analog communications, the nodes can multiply their values with $1/2$ and then coherently transmit the resultant values to node C resulting in $(1/2)\cdot 0+(1/2)\cdot 1 = 0.5$ at the destination. On the other hand, if the nodes were to transmit their data using digital communications then transmitting simultaneously over the same time/frequency slot can only result in node C decoding the received signal as either 0 or 1 (because of the digital nature of its receiver) and consequently, the two nodes would either need to take turns in transmitting their values to node C or they would need to transmit over different frequency slots. Either option results in double the time (or bandwidth) and energy. In addition, the receiver would need to perform an arithmetic operation to achieve the final result.

V. Conclusions and Extensions

This article described how compressed sensing techniques could be utilized to reconstruct sparse or compressible networked data in a variety of practical settings, including general multi-hop networks and wireless sensor networks. Compressed sensing provides two key features, universal sampling and decentralized encoding, making it a promising new paradigm for networked data analysis. The focus here was primarily on managing resources during the encoding process, but it is important to note that the decoding step also poses a significant challenge. Indeed, the study of efficient decoding algorithms remains at the forefront of current research [23]–[25].

In addition, specialized measurement matrices, such as those resulting from Toeplitz-structured matrices [26] and the incoherent basis sampling methods described in [27], lead to significant reductions in the complexity of convex decoding methods. Fortunately, the sampling matrices inherent to these methods can be easily implemented using the network projection approaches described above. For example, Toeplitz-structured CS matrices naturally result when each node uses the same random number generation scheme and seed value, where at initialization each node advances its own random sequence by its unique (integer) identifier. Similarly, random samples from any orthonormal basis (the observation model described in [27]) can easily be obtained in the settings described above if each node is preloaded with its weights for each basis element in the corresponding orthonormal transformation matrix. For each observation, the requesting node (or fusion center) broadcasts a random integer between 1 and n to the nodes to specify which transform coefficient from the predetermined basis should be obtained, and the projection is delivered using any suitable method described above.

Finally, it worth noting that matrices satisfying the RIP also approximately preserve additional geometrical structure on subspaces of sparse vectors, such as angles and inner products, as shown in [28]. A useful consequence of this result is that an ensemble of CS observations can be "data mined" for events of interest [29], [30]. For example, consider a network whose data may contain an anomaly that originated at one of m candidate nodes. An ensemble of CS observations of the networked data, collected without any a priori information about the anomaly, can be analyzed "post-mortem" to accurately determine which candidate node was the likely source of the anomaly. Such extensions of CS theory suggest efficient and scalable techniques for monitoring large-scale distributed networks, many of which can be performed without the computational burden of reconstructing the complete networked data.

REFERENCES

[1] S. S. Pradhan, J. Kusuma, and K. Ramchandran, "Distributed compression in a dense microsensor network," *IEEE Signal Processing Mag.*, vol. 19, no. 2, pp. 51–60, Mar. 2002.

[2] E. J. Candes and T. Tao, "Decoding by linear programming," *IEEE Trans. Inform. Theory*, vol. 51, no. 12, pp. 4203–4215, Dec. 2005.

[3] D. L. Donoho, "Compressed sensing," *IEEE Trans. Inform. Theory*, vol. 52, no. 4, pp. 1289–1306, Apr. 2006.

[4] J. Haupt and R. Nowak, "Signal reconstruction from noisy random projections," *IEEE Trans. Inform. Theory*, vol. 52, no. 9, pp. 4036–4048, Sept. 2006.

[5] E. Candès and T. Tao, "The Dantzig selector: statistical estimation when p is much larger than n," *Annals of Statistics*, to appear.

[6] W. U. Bajwa, J. Haupt, A. M. Sayeed, and R. Nowak, "Compressive wireless sensing," in *Proc. IPSN'06*, Nashville, TN, Apr. 2006, pp. 134–142.

[7] M. Rabbat, J. Haupt, A. Singh, and R. Nowak, "Decentralized compression and predistribution via randomized gossiping," in *Proc. IPSN'06*, Nashville, TN, Apr. 2006, pp. 51–59.

[8] W. U. Bajwa, J. Haupt, A. M. Sayeed, and R. Nowak, "A universal matched source-channel communication scheme for wireless sensor ensembles," in *Proc. ICASSP'06*, Toulouse, France, May 2006, pp. 1153–1156.

[9] D. Baron, M. B. Wakin, M. F. Duarte, S. Sarvotham, and R. G. Baraniuk, "Distributed compressed sensing," pre-print. [Online]. Available: http://www.ece.rice.edu/~drorb/pdf/DCS112005.pdf

[10] W. Wang, M. Garofalakis, and K. Ramchandran, "Distributed sparse random projections for refinable approximation," in *Proc. IPSN'07*, Cambridge, MA, April 2007, pp. 331–339.

[11] E. J. Candes and T. Tao, "Near-optimal signal recovery from random projections: Universal encoding strategies?" *IEEE Trans. Inform. Theory*, vol. 52, no. 12, pp. 5406–5425, Dec. 2006.

[12] R. Baraniuk, M. Davenport, R. A. DeVore, and M. B. Wakin, “A simple proof of the restricted isometry property for random matrices,” *Constructive Approximation*, to appear.

[13] R. Wagner, R. Baraniuk, S. Du, D. Johnson, and A. Cohen, “An architecture for distributed wavelet analysis and processing in sensor networks,” in *Proc. IPSN’06*, Nashville, TN, April 2006, pp. 243–250.

[14] M. Crovella and E. Kolaczyk, “Graph wavelets for spatial traffic analysis,” in *Proc. IEEE Infocom*, vol. 3, Mar. 2003, pp. 1848–1857.

[15] R. Coifman and M. Maggioni, “Diffusion wavelets,” *Applied Computational and Harmonic Analysis*, vol. 21, no. 1, pp. 53–94, July 2006.

[16] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, “Randomized gossip algorithms,” *IEEE Trans. Inform. Theory*, vol. 52, no. 6, pp. 2508–2530, June 2006.

[17] M. Gastpar and M. Vetterli, “Source-channel communication in sensor networks,” in *Proc. IPSN’03*, Palo Alto, CA, Apr. 2003, pp. 162–177.

[18] K. Liu and A. M. Sayeed, “Optimal distributed detection strategies for wireless sensor networks,” in *Proc.* 42*nd Annual Allerton Conference on Commun., Control and Comp.*, Oct. 2004.

[19] M. Gastpar and M. Vetterli, “Power, spatio-temporal bandwidth, and distortion in large sensor networks,” *IEEE Journal Select. Areas Commun.*, vol. 23, no. 4, pp. 745–754, Apr. 2005.

[20] W. U. Bajwa, A. M. Sayeed, and R. Nowak, “Matched source-channel communication for field estimation in wireless sensor networks,” in *Proc. IPSN’05*, Los Angeles, CA, Apr. 2005, pp. 332–339.

[21] W. U. Bajwa, J. Haupt, A. M. Sayeed, and R. Nowak, “Joint source-channel communication for distributed estimation in sensor networks,” *IEEE Trans. Inform. Theory*, vol. 53, no. 10, pp. 3629–3653, Oct. 2007.

[22] M. Gastpar, M. Vetterli, and P. L. Dragotti, “Sensing reality and communicating bits: A dangerous liaison,” *IEEE Signal Processing Mag.*, vol. 23, no. 4, pp. 70–83, July 2006.

[23] A. C. Gilbert and J. Tropp, “Signal recovery from partial information via orthogonal matching pursuit,” *IEEE Trans. Inform. Theory*, to appear.

[24] M. Figueiredo, R. Nowak, and S. Wright, “Gradient projection for sparse reconstruction: Applications to compressed sensing and other inverse problems,” *IEEE Journal Select. Topics in Signal Processing*, to appear.

[25] S.-J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky, “A method for large-scale l1-regularized least squares problems with applications in signal processing and statistics,” *IEEE Journal Select. Topics in Signal Processing*, to appear.

[26] W. Bajwa, J. Haupt, G. Raz, and R. Nowak, “Toeplitz-structured compressed sensing matrices,” in *Proc. SSP’07*, Madison, WI, Aug. 2007, pp. 294–298.

[27] E. Candès and J. Romberg, “Sparsity and incoherence in compressive sampling,” *Inverse Problems*, vol. 23, no. 3, pp. 969–985, 2006.

[28] J. Haupt and R. Nowak, “A generalized restricted isometry property,” University of Wisconsin - Madison, Tech. Rep. ECE-07-1, May 2007.

[29] ——, “Compressive sampling for signal detection,” in *Proc. ICASSP’07*, Honolulu, HI, Apr. 2007.

[30] J. Haupt, R. Castro, R. Nowak, G. Fudge, and A. Yeh, “Compressive sampling for signal classification,” in *Proc. 40th Asilomar Conference on Signal, Systems, and Computers*, Pacific Grove, CA, Oct. 2006, pp. 1430–1434.