Compressed Sensing Reconstruction via Belief Propagation

Shriram Sarvotham, Dror Baron and Richard G. Baraniuk

Department of Electrical and Computer Engineering

Rice University, Houston, TX 77005, USA

July 14, 2006

Abstract

Compressed sensing is an emerging field that enables to reconstruct sparse or compressible signals from a small number of linear projections. We describe a specific measurement scheme using an LDPC-like measurement matrix, which is a real-valued analogue to LDPC techniques over a finite alphabet. We then describe the reconstruction details for mixture Gaussian signals. The technique can be extended to additional compressible signal models.

1 Introduction

In many signal processing applications the focus is often on identifying and estimating a few significant coefficients from a high dimension vector. The wisdom behind this is the ubiquitous compressibility of signals: most of the information contained in a signal resides in a few large coefficients. Traditional sensing and processing first acquires the entire data, only to throw away most of the coefficients and retaining the small number of significant coefficients. Clearly, it is wasteful to sense or compute all of the coefficients when most of it will be discarded at a later stage. This naturally suggests the question: can we sense compressible signals in a compressible way? In other words, can we sense only that part of the signal that will not be thrown away? The ground-breaking work in compressed sensing (CS) pioneered by Candés et al. [1] and Donoho [2] answers the above question in the affirmative. They demonstrate that the information contained in the few significant coefficients can be captured (encoded) by a small number of random linear projections. The original signal can then be reconstructed (decoded) from these random projections using an appropriate decoding scheme.

The initial discovery has led to a vibrant activity in the area of CS research, opening many intriguing questions both in theoretical and practical aspects of CS. Two fundamental questions naturally emerge in compressed sensing. The first question concerns efficiency: what is the minimum number of projections required to capture the information contained in the signal (either losslessly or with a certain fidelity)? Clearly, the number of measurements

needed depends on the signal model as well as the measurement model. The second question relates to algorithmic achievability: can we construct practical CS coding schemes that approach the performance limits? Questions such as these have been addressed in various other contexts. The vision of our work is to leverage the insights from information theory [3] to obtain new CS coding algorithms. In particular, we draw insights from low density parity check (LDPC) codes.

1.1 Information theory

The main problem that information theory deals with is reliable transmission of information over communication channels. A fundamental result by Shannon states that the upper-limit on the rate at which we can send information over a channel is given by the *channel capacity* [4]. Since Shannon's seminal work, several approaches to building practical codes have been proposed. The emphasis has been on imposing *structure* to the codewords; for example, explicit algebraic construction of very good codes were designed for some channels. Although the decoding of such codes has polynomial complexity (and thus practical), most of these codes fared poorly in the asymptotic regime – they achieved arbitrarily small probabilities of decoding errors only by decreasing the information rate to zero. The grand breakthrough occurred quite recently, with the advent of turbo codes [5] and the rediscovery of LDPC codes [6]. These codes belong to a class of linear error correcting codes that use *sparse* parity check matrices and achieve information rates close to the Shannon limit. In addition to their excellent performance, turbo and LDPC codes lend themselves to simple and practical decoding algorithms, thanks to the sparse structure of the parity check matrices.

1.2 Compressed sensing

Consider a discrete-time signal $x \in \mathbb{R}^N$ that has only $K \ll N$ non-zero coefficients. The core tenet of CS is that it is unnecessary to measure all the N values of x; rather, we can recover x from a small number of projections onto an *incoherent* basis [1, 2]. To measure (encode) x, we compute the measurement vector $y \in \mathbb{R}^M$ as M linear projections of x via the matrix-vector multiplication $y = \Phi x$. Our goal is to reconstruct (decode) x – either accurately or approximately – given y and Φ using $M \ll N$ measurements.

Although the recovery of the signal x from the measurements $y = \Phi x$ appears to be a severely ill-posed inverse problem, the strong prior knowledge of *sparsity* in x gives us hope to reconstruct x using $M \ll N$ measurements. In fact the signal recovery can be achieved using optimization by searching for the sparsest signal that agrees with the M observed measurements in y. The key observation is that the signal x is the solution to the ℓ_0 minimization

$$\widehat{x} = \arg\min \|x\|_0 \quad \text{s.t. } y = \Phi x \tag{1}$$

with overwhelming probability as long as we have sufficiently many measurements, where $\|\cdot\|_0$ denotes the ℓ_0 "norm" that counts the number of non-zero elements. Unfortunately, solving the ℓ_0 optimization is known to be an NP-complete problem [7]. In order to recover the signal, the decoder needs to perform combinatorial enumeration of all the $\binom{N}{K}$ possible sparse subspaces.

The practical revelation that supports the CS theory is that it is not necessary to resort to combinatorial search to recover the set of non-zero coefficients of x from the ill-posed inverse problem $y = \Phi x$. A much easier problem yields an equivalent solution. We need only solve for the ℓ_1-sparsest coefficients that agree with the measurements y [1, 2]

$$\widehat{x} = \arg\min \|x\|_1 \quad \text{s.t. } y = \Phi x, \tag{2}$$

as long as Φ satisfies the restricted isometry (RIP) condition [1]. The RIP condition is shown to be satisfied by a measurement strategy using independent and identically distributed (iid) Gaussian entries for Φ. Furthermore, the ℓ_1 optimization problem, also known as *Basis Pursuit* [8], is significantly more approachable and can be solved with linear programming techniques. The decoder based on linear programming requires cK projections for signal reconstruction where $c \approx \log_2(1+N/K)$ [9] and reconstruction complexity is $\Omega(N^3)$ [7, 10, 11].

1.3 Compressed sensing reconstruction algorithms

While linear programming techniques figure prominently in the design of tractable CS decoders, their $\Omega(N^3)$ complexity still renders them impractical for many applications. We often encounter sparse signals with large N; for example, current digital cameras acquire images with the number of pixels N of the order of 10^6 or more. For such applications, the need for faster decoding algorithms is critical. Furthermore, the encoding also has high complexity; encoding with a full Gaussian Φ requires $\Theta(MN)$ computations.[1] This realization has spawned a large number of decoding schemes in the CS research community in search of new measurement strategies and accompanying low-complexity decoders. We briefly review some of the previous work.

At the expense of slightly more measurements, iterative greedy algorithms have been developed to recover the signal x from the measurements y. Examples include the iterative Orthogonal Matching Pursuit (OMP) [12], matching pursuit (MP), and tree matching pursuit (TMP) [13] algorithms. In CS applications, OMP requires $c \approx 2\ln(N)$ [12] to succeed with high probability; decoding complexity is $\Theta(NK^2)$. Unfortunately, $\Theta(NK^2)$ is cubic in N and K, and therefore OMP is also impractical for large K and N.

Donoho et al. recently proposed the Stage-wise Orthogonal Matching Pursuit (StOMP) [14]. StOMP is an enhanced version of OMP where multiple coefficients are resolved at each stage of the greedy algorithm, as opposed to only one in the case of OMP. Moreover, StOMP takes a fixed number of stages while OMP takes many stages to recover the large coefficients of x. The authors show that StOMP with fast operators for Φ (such as permuted FFT's) can recover the signal in $N \log N$ complexity. Therefore StOMP runs much faster than OMP or ℓ_1 minimization and can be used for solving large-scale problems.

While the CS algorithms discussed above use a full Φ (all the entries of Φ are non-zero in general), a class of techniques has emerged that employ sparse Φ and use group testing to decode x. Cormode and Muthukrishnan proposed a fast algorithm based on group testing [15, 16]. Their scheme considers subsets of the signal coefficients in which we expect at most

[1] For two functions $f(n)$ and $g(n)$, $f(n) = O(g(n))$ if $\exists c, n_0 \in \mathbb{R}^+,\ 0 \le f(n) \le cg(n)$ for all $n > n_0$. Similarly, $f(n) = \Omega(g(n))$ if $g(n) = O(f(n))$ and $f(n) = \Theta(g(n))$ if $\exists c_1, c_2, n_0 \in \mathbb{R}^+,\ 0 \le c_1 g(n) \le f(n) \le c_2 g(n)$ for all $n > n_0$.

Table 1: *CS schemes*

Scheme	**Setting**	M_{min}	**Complexity**
ℓ_0 Optimization	noise-free measurements	$K+1$	NP complete
ℓ_1 Optimization	noise-free measurements	$K\log(1+N/K)$	$\Omega(N^3)$
ℓ_1 regularization	noisy measurements	$K\log(1+N/K)$	$\Omega(N^3)$
OMP	noiseless measurements	$2K\log N$	NK^2
StOMP	noiseless measurements	$K\log N$	$N\log N$
Cormode-Muthu	noiseless measurements	$K\log^2 N$	$K\log^2 N$
Chaining Pursuit	noiseless measurements	$K\log^2 N$	$K\log^2 N\log^2 K$
Sudocodes [18]	noiseless measurements	$K\log N$	$K(\log K)(\log N)$
CS-LDPC	noisy measurements	$K\log N$	$N\log N$

one large coefficient to lie. Within this set, they locate the position and value of the large coefficient using a Hamming code construction. Their decoding scheme has $O(K\log^2(N))$ complexity, but they require $M = O(K\log^2(N))$ measurements.

Gilbert et al. [17] propose the Chaining Pursuit (CP) algorithm for reconstructing compressible signals. CP reconstructs with $O(K\log^2(N))$ non-adaptive linear measurements in $O(K\log^2(N)\log^2(K))$ time. Simulations reveal that CP works best for "super-sparse" signals, where the *sparsity rate* $S \triangleq K/N$ is very small [18]. As we increase S, CP requires an enormous number of measurements; for some ranges of S the number of measurements M exceeds the signal length N, which is undesirable.

Table 1 summarizes the number of measurements required and computational complexity of several CS schemes proposed in the literature. Note that the efficiency and complexity depends on the signal class considered as well as the reconstruction scheme. The results suggests that we can reduce the CS coding complexity only at the expense of more measurements.

1.4 Connections between channel coding and CS

The world of error control coding and CS share intriguing connections. For example, a fully Gaussian/Rademacher CS measurement matrix [1, 2] is reminiscent of Shannon's fully random code construction. Although a fully random CS matrix efficiently captures the information content of sparse signals with very few projections, they are not amenable to fast encoding and decoding schemes. On the other hand, the grand success of sparse codes

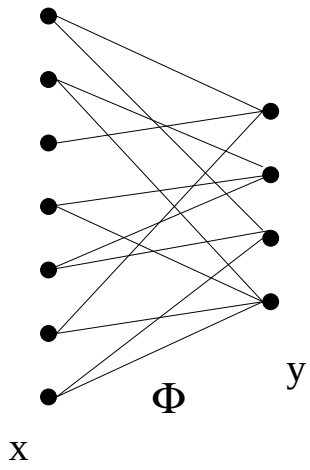

Figure 1: *The CS measurement operator Φ can be represented as a bipartite graph. The nodes on the left correspond to the signal coefficients. The right side nodes correspond to the measurements.*

in error control coding such as LDPC codes strongly suggests the use of sparse compressed sensing matrices. This leading insight inspires the following questions:

- Can sparse CS matrices efficiently encode sparse signals?
- Can we take advantage of the sparse structure to reduce the encoding and decoding complexity?

To address these questions, we overview LDPC codes and describe how we can leverage various insights to CS.

1.5 Lessons from LDPC codes applied to CS

An LDPC code is a block code that has a parity check matrix H that is sparse along every row and every column [6, 19]. The parity check matrix of an LDPC code can be conveniently represented as a bipartite graph (Figure 1). The nodes in the graph are one of two classes: the first class of nodes represent the transmitted bits (called the bit nodes) and the second class of nodes represent the constraints (called check nodes). Although any linear code can be represented by a graph, what makes LDPC codes special is that the bipartite graph is sparse: the number of edges in the graph scales roughly linearly with N, rather than quadratically. The sparsity in the graph representation of LDPC codes simplifies the decoding process: LDPC codes can be decoded by running a low-complexity message-passing algorithm (such as belief-propagation) over the graph.

Message-passing algorithms start with the input consisting of the prior probabilities of the bits. The algorithms use the parity check relationship among the bits to *iteratively* pass messages between the nodes in the graph and extract the posterior probabilities for the codeword bits. Although these algorithms yield accurate posterior probabilities only when

the bipartite graph is cycle free, it turns out that the algorithms compute astonishingly accurate estimates for LDPC decoding in spite of the presence of cycles in the graph.

LDPC codes offer an important insight to the area of CS: namely that *sparse* coding matrices offer great promise in efficiently encoding signals, as well as enabling low complexity algorithms based on the sparse graph structure. We will later deliver on these promises.

1.6 Contributions: Connecting CS decoding to graph decoding algorithms

The key idea underlying this paper is to extend LDPC measurement techniques over a finite alphabet to matrix multiplications over the real numbers. Surprisingly, the usage of similar matrix structures provides superior processing results even in this setting. Despite the significant benefit provided by this extension to real valued signals, the LDPC matrices that we use are not universal in the sense that they do not work for signals that are sparse or approximately sparse in any basis. Although universality can be addressed by modifying the measurement matrix and reconstruction techniques (Section 4), this is only of secondary concern here.

Encoder: We have developed strategies to encode (measure) the signal using sparse $\{0,1\}$ or $\{0,1,-1\}$ LDPC-like matrices. We compute the measurements $y = \Phi x$ using a sparse CS matrix Φ, with the entries of Φ are restricted to $\{0,1,-1\}$. Hence the measurements are just sums and differences of small subsets of the coefficients of x. A sparse Φ can be represented as a sparse bipartite graph (Figure 1). The graph shown in the figure could corresponds to the following Φ:

$$\Phi = \begin{pmatrix} -1 & 0 & 1 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & -1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & -1 & 0 & -1 \\ 0 & -1 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}.$$

The design of Φ (such as column weight, row weight, etc) is based on the properties of the signal as well as the accompanying decoding algorithm. The goal is to construct a decoder (reconstruction algorithm) and Φ structure for which the decoder consumes minimum number of measurements and operates at low complexity. To achieve this goal, we have developed two accompanying low-complexity reconstruction schemes.

Decoder for strictly sparse signals: Our first decoding algorithm appears in [18] and is called *Sudocodes*. This algorithm is based on *graph reduction*, and is tailor-made for strictly sparse signals: signals in $\mathbb{R}^N$ that contain exactly $K \ll N$ non-zero coefficients. The key idea of Sudocodes is that decoding the coefficients of x leads to a series of graph reduction steps. The decoding times of sudo-decoding are comparable to other low-complexity schemes [15–17]. In contrast, Sudocodes require fewer measurements for perfect reconstruction, compared to the other schemes. For details, see [18].

Decoder for approximately sparse signals: The highlight of this paper is a second reconstruction algorithm based on *message passing* over a graph. This algorithm is tailor-made for approximately sparse signals – signals in $\mathbb{R}^N$ that have $K \ll N$ large coefficients

and the remaining coefficients are small but non-zero. Examples of such signals are strictly sparse signals in noise and compressible signals.[2]

The decoding approach is based on message passing over graphs to solve a Bayesian inference problem. Using the two-state mixture Gaussian distribution as a prior model for the signal coefficients, we compute an estimate of x that explains the measurements and best matches the prior. We use belief propagation, similar to the decoding in LDPC codes and turbo codes [5, 6, 19]. Our technique for approximately sparse signals has $O(N\log(N))$ encoding complexity and $O(N\log(N))$ worst case decoding complexity. We also show that only $M = O(K\log(N))$ measurements are needed.

The remainder of this paper is organized as follows. Our CS-LDPC encoder and decoder are described in Sections 2 and 3, respectively. Variations and applications are described in Sections 4 and Section 5, and Section 6 concludes. We have also included some details in an appendix.

2 CS-LDPC encoding

A CS matrix Φ can be represented as a bipartite graph G (Figure 1). The edges of G connect a coefficient node $x(i)$ to a measurement node $y(j)$. The neighbors of a given measurement node is the set of coefficient nodes that are used to generate that measurement. Our techniques rely on the sparse nature of G; and so we choose Φ to be a *sparse* matrix. Thus, a measurement $y(j)$ is computed using a small subset of the coefficients of x. We impose an additional constraint on Φ to enable even simpler encoding: the entries of Φ are restricted to $\{0, 1, -1\}$. Hence the computation of y involves only sums and differences of small subsets of the coefficients of x. The advantage of representing Φ as a graph is that the accompanying decoding procedure can employ graph algorithms over G; these algorithms can either be a graph reduction or a message passing scheme over the edges of G.

In addition to the core structure of the encoding matrix described above, we may introduce other constraints to tailor-make the measurement process to the signal model that generated the input signal. Some of these additional constraints that we use in this paper are listed below.

1. Each row of Φ contains exactly L non-zero entries (1 or -1) placed randomly. The row weight L is chosen based on the properties of the signal that we are trying to encode (such as sparsity), and also the decoding process.

2. In some situations, we fix the column weight for each column of Φ to be a constant R.

3. While sparse Φ is well suited for sparse signals in the canonical "spike" basis, the encoding can be generalized to the case where x is sparse in an arbitrary basis Ψ. In this case, we multiply the encoding matrix Φ with the sparsifying matrix Ψ and use $\Phi\Psi$ to encode x as $y = (\Phi\Psi)x$.

[2] A signal is compressible if its coefficients sorted by magnitude obey a power law; it can be approximated well by a sparse signal.

Encoding complexity: Consider the case where the row weight of Φ is a constant given by L. Each measurement requires $O(L)$ additions/subtractions. Therefore, encoding requires $O(ML) = O(N \log(N))$ computations to generate the M measurements. The above computation assumes that the sparsifying basis $\Psi = I$. For the general case $\Psi \neq I$, the encoding matrix $\Phi\Psi$ may not be sparse. Therefore, the encoding must perform extra computations; this extra cost is $O(N^2)$ in general. Fortunately, in many practical situations Ψ is structured (e.g., Fourier or wavelet bases) and thus amenable to fast computation. Therefore, extending our techniques to such bases is feasible.

3 CS-LDPC decoding of approximately sparse signals

The decoding process for LDPC measurement of strictly sparse signals is provided using our Sudocode approach [18]. The focus here is on decoding *approximately* sparse signals.

3.1 Signal model: approximately sparse signals

We use the two-state Gaussian mixture distribution to model the coefficients of approximately sparse signals [20–22]. This stochastic model for the signal succinctly captures our prior knowledge about its sparsity. We describe the signal model below.

Let $X = [X(1), X(2), ..., X(N)]$ be a random vector in $\mathbb{R}^N$, and let us consider the signal $x = [x(1), x(2), ..., x(N)]$ as an outcome of X. The key observation we exploit is that a sparse signal consists of a small number of large coefficients and a large number of small coefficients. Thus each coefficient of the signal can be associated with a state variable that can take on two values: "high", corresponding to a coefficient of large magnitude and "low", representing a coefficient of small magnitude. Let the state associated with the j'th coefficient be denoted by $q(j)$, where either $q(j) = 1$ (high) or $q(j) = 0$ (low). We view $q(j)$ as an outcome of the state random variable $Q(j)$ that can take values from $\{0, 1\}$. Let $Q = [Q(1), Q(2), ..., Q(N)]$ be the state random vector associated with the signal. The actual state configuration $q = [q(1), q(2), ..., q(N)] \in \{0, 1\}^N$ is one of 2^N possible outcomes of Q. We associate with each possible state of coefficient j a probability density for $X(j)$, resulting in a two state mixture distribution for that coefficient. For the "high" state, we choose a high-variance zero mean Gaussian distribution, and for the "low" state, we choose a low-variance zero-mean Gaussian distribution to model the coefficients. Thus the distribution of $X(j)$ conditioned on $Q(j)$ is given by

$$\begin{aligned} X(j)|Q(j) = 1 &\sim \mathcal{N}(0, \sigma_1^2), \qquad \text{and} \\ X(j)|Q(j) = 0 &\sim \mathcal{N}(0, \sigma_0^2), \end{aligned}$$

where $\sigma_1^2 > \sigma_0^2$. For simplicity, we assume independence between coefficients: the outcome of the state or value of a coefficient does not influence the state or value of another coefficient.[3] Finally, to ensure that we have about K large magnitude coefficients, we choose the probability mass function (pmf) of the state variable $Q(j)$ to be Bernoulli with $P[Q(j) = 1] = S$

[3]The model can be extended to capture the dependencies between coefficients, if any. We leave this for future work.

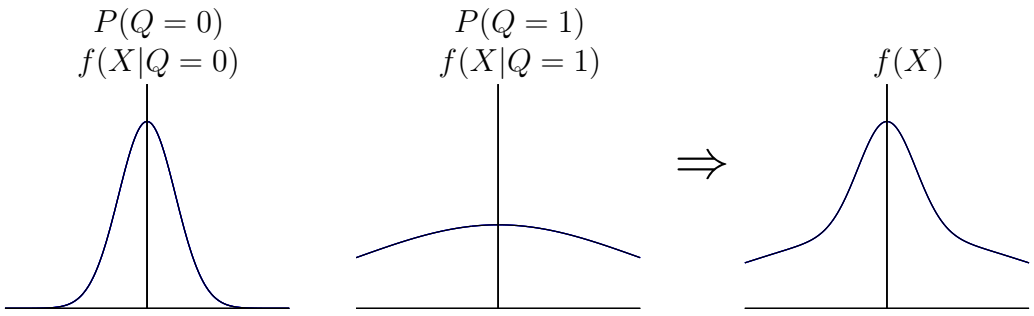

Figure 2: *Illustration of the 2-state mixture Gaussian model for the Random Variable X. The distribution of X conditioned on the state variable Q are depicted. Also shown is the overall non-Gaussian distribution for X. We use this mixture distribution to model the prior for the coefficients of a sparse signal.*

and $P[Q(j) = 0] = 1 - S$, where $S = K/N$ is the sparsity rate (we assume that the sparsity rate is known).

The two-state Gaussian mixture model model is completely characterized by three parameters: the distribution of the state variable (parametrized by the sparsity rate S) and the variances σ_1^2 and σ_0^2 of the Gaussian pdf's corresponding to each state. Figure 2 provides a pictorial illustration of the two state mixture Gaussian model.

Mixture Gaussian models have been successfully employed in image processing and inference problems because they are simple yet effective in modeling real-world signals [20–22]. Theoretical connections have also been made between wavelet coefficient mixture models and the fundamental parameters of Besov spaces – function spaces that have proved invaluable for characterizing real-world images [23]. Furthermore, by increasing the number of states and allowing non-zero means, we can make the fit arbitrarily close for densities with with a finite number of discontinuities [24].

Although we study the use of two state mixture Gaussian distributions for modeling the coefficients of the signal, the decoding techniques proposed in the sequel can be used with other prior distributions as well.

3.2 Decoding via statistical inference

3.2.1 MMSE and MAP estimation for CS

Decoding for approximately sparse signals is a Bayesian inference problem that can be solved efficiently using a message passing algorithm over factor graphs.

The key idea in Bayesian inference is to determine that signal that is consistent with the observed measurements that best matches our signal model. As before, we observe the linear projections $y \in \mathbb{R}^M$ of a sparse signal $x \in \mathbb{R}^N$ given by $y = \Phi x$, where Φ is the compressed sensing matrix of dimension $M \times N$, with $M \ll N$. The goal is to estimate x given Φ and y. Resolving x that satisfy $y = \Phi x$ describes an under-determined set of equations and

has infinitely many solutions. The solutions lie along a hyperplane of minimum dimension $N - M$ [25]. We use Bayesian inference to locate the solution in this hyperplane that best matches our prior signal model. We pose the CS reconstruction as the following inference problems (we consider MAP and MMSE estimates):

$$\begin{aligned} \widehat{x}_{\text{MMSE}} &= \arg \min_{x'} E\|X - x'\|_2^2 \qquad \text{s.t. } y = \Phi x', \qquad \text{and} \\ \widehat{x}_{\text{MAP}} &= \arg \max_{x'} P[X = x'] \qquad \text{s.t. } y = \Phi x', \end{aligned}$$

where the expectation is taken over the prior distribution P of X. The MMSE estimate can also be expressed as the conditional mean, given by $\widehat{x}_{\text{MMSE}} = E[X|Y = y]$, where Y is the random vector in $\mathbb{R}^M$ corresponding to the measurement. In the sequel, we first present a scheme to determine the exact MMSE estimate of the signal (Section 3.2.2). This scheme has exponential complexity, and hence is impractical for large N. However, the sparse structure of Φ enables us to use low-complexity message-passing schemes to estimate x approximately; this procedure is described in Section 3.2.3.

3.2.2 Exact solution to CS statistical inference

To compute the MAP and MMSE estimates of the signal, let us first consider the simple case in which we know the state configuration q of the signal. In this setting, the MMSE estimate of x can be computed using the pseudo-inverse of Φ after applying the covariance matrix of the associated state configuration (as described below). The covariance matrix Σ_q associated with a state configuration q is given by $\Sigma_q = E[XX^T|Q = q]$. The covariance matrix is diagonal (because the coefficients are independent), with the j'th diagonal entry equal to σ_0^2 if the state $q(j) = 0$, or σ_1^2 if $q(j) = 1$. We have the following Theorem:

Theorem 1 *Given the measurements $y = \Phi x$, then the MMSE and the MAP estimate of x conditioned on knowing the state configuration q is given by*

$$\widehat{x}_{\text{MAP},q} = \widehat{x}_{\text{MMSE},q} = \Sigma_q \Phi^T \left(\Phi \Sigma_q \Phi^T\right)^{-1} y,$$

where Σ_q is the covariance of X conditioned on knowing the state configuration q.

Proof sketch: Conditioned on knowing the state configuration q, the distribution of X is a multivariate Gaussian with covariance Σ_q. Therefore,

$$P[X = x|Q = q] = \frac{1}{(2\pi)^{N/2} \det(\Sigma_q)^{1/2}} e^{-\frac{1}{2} x^T \Sigma_q^{-1} x}.$$

The probability $P(x)$ is maximized when $x^T \Sigma_q^{-1} x$ is minimum. Consider the MAP estimate $\widehat{x}_{\text{MAP},q}$: this is given by $\widehat{x}_{\text{MAP},q} = \arg\min_{x'} x'^T \Sigma_q^{-1} x'$ such that $y = \Phi x'$. Now consider change of variables, by setting $\theta = \Sigma_q^{-1/2} x'$. In this setting, we have $\theta_{\text{MAP}} = \arg\min_\theta \theta^T \theta$ such that $y = \Phi \Sigma_q^{1/2} \theta$. This is a simple least squares problem and the solution can be computed using the pseudo-inverse: $\theta_{\text{MAP}} = (\Phi \Sigma_q^{1/2})^+ y = \Sigma_q^{1/2} \Phi^T \left(\Phi \Sigma_q \Phi^T\right)^{-1} y$. Because $\widehat{x}_{\text{MAP},q} = \Sigma_q^{1/2} \theta_{\text{MAP}}$, we have proved the result for $\widehat{x}_{\text{MAP},q}$. Finally, the MMSE and MAP estimates are identical for multivariate Gaussian random variables, and this proves the Theorem. □

We now present a scheme with exponential complexity that computes $\widehat{x}_{\text{MMSE}}$ when the states are unknown. We use this solution as a yardstick to compare the approximate solutions we propose later. To determine $\widehat{x}_{\text{MMSE}}$ accurately, we compute the conditional MMSE estimate for each of the 2^N state configurations using Theorem 1. The following Theorem relates the overall MMSE estimate to the conditional MMSE estimates.

Theorem 2 *The MMSE estimate of x when the state configuration is unknown is given by*

$$\widehat{x}_{\text{MMSE}} = \sum_{q \in [0,1]^N} P[Y = y | Q = q] \cdot P[Q = q] \cdot \widehat{x}_{\text{MMSE},q}.$$

Proof sketch: The expected value of the MMSE error obeys the equalities

$$\begin{aligned}
\widehat{x}_{\text{MMSE}} &= E\left[X|Y = y\right] \\
&= \sum_{q \in [0,1]^N} E[X, Q = q | Y = y] \\
&= \sum_{q \in [0,1]^N} P[Q = q | Y = y] \cdot E[X | Q = q, Y = y] \\
&= \sum_{q \in [0,1]^N} P[Q = q | Y = y] \cdot \widehat{x}_{\text{MMSE},q} \\
&= \sum_{q \in [0,1]^N} P[Y = y | Q = q] \cdot P[Q = q] \cdot \widehat{x}_{\text{MMSE},q},
\end{aligned}$$

which completes the proof. □

The signal estimate can be interpreted geometrically. As a simple example, consider a 2D signal ($N = 2$) where each coefficient is modeled by a mixture Gaussian distribution. Figure 3 shows a sample sketch of contours of equi-probable values of x. This enables to visualize the MAP and MMSE estimates of x.

It is well known that exact inference in arbitrary graphical models is an NP hard problem [26]. However, sparse matrices Φ lend themselves to easy approximate inference using message passing algorithms such as BP. In the sequel, we present a message-passing approach for CS reconstruction, where we compute the approximate marginals for $[X(j)|Y]$ and thus evaluate the MMSE or MAP estimate for each coefficient.

3.2.3 Approximate solution to CS statistical inference via message passing

We now employ belief propagation (BP), an efficient scheme to solve inference problems by message passing over graphical models [19, 27–32]. We employ BP by message passing over factor graphs. Factor graphs enable fast algorithms to compute global functions of many variables by exploiting the way in which the global function factors into a product of simpler local functions, each of which depends on a subset of the variables [33].

The factor graph shown in Figure 4 captures the relationship between the signal coefficients, their states, and the observed CS measurements. The factor graph contains two

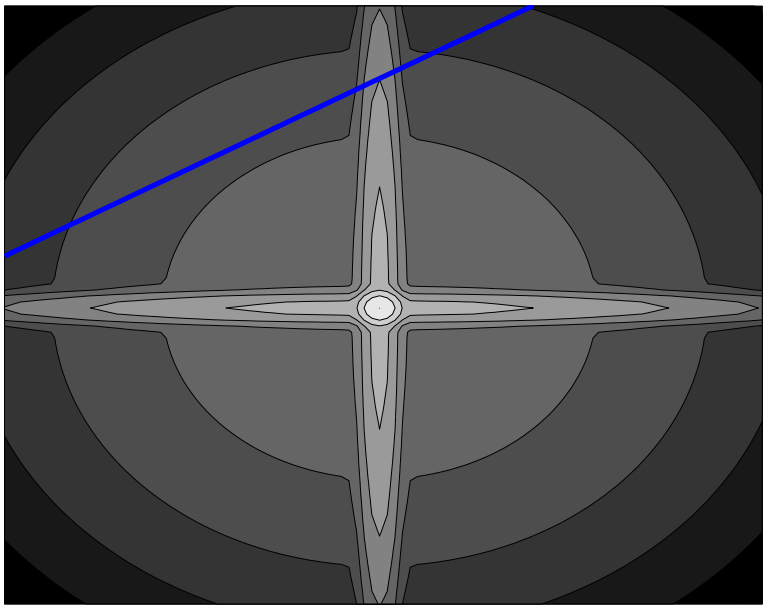

Figure 3: *Equi-probable surfaces using the mixture Gaussian model for the coefficients. The line represents the set of x's that satisfy $\Phi x = y$. As we expand the equi-probable surface, it touches the line at the MAP estimate.*

types of vertices: variable nodes (black) and constraint nodes (white). The factor graph is bipartite: all edges connect a variable node to a constraint node. A constraint node encapsulates the dependencies (constraints) that its neighbors (variable nodes) are subjected to. The factor graph has three types of variable nodes: states $Q(j)$, coefficients $X(j)$ and the measurements $Y(i)$. Furthermore it has three types of constraint nodes. The first type – prior constraint nodes – impose the Bernoulli prior on the state variables. The second type of constraint nodes connect the state variables and the coefficient variables. These constraints impose the conditional distribution of the coefficient value given the state. The third type of constraint nodes connect one measurement variable to a set of coefficient variables that are involved in computing that measurement. In the CS inference problem, the observed variables are the outcomes of the measurement variables Y. We employ BP to (approximately) infer the probability distributions (beliefs) of the coefficient and the state variables conditioned on the measurements. In other words, BP provides the mechanism to infer the marginal distributions $P[X(j)|Y = y]$ and $P[Q(j)|Y = y]$. Knowing these distributions, one can extract either the MAP estimate (the value of $x(j)$ that maximizes $P[X(j)|Y = y]$) or the MMSE estimate (the mean of $P[X(j)|Y = y]$) for each coefficient.

An overview of BP is provided in Appendix A. Let us describe some particularly interesting details:

1. There are only 2 types of message processing in CS reconstruction using BP: multiplication of beliefs (at the variable nodes) and convolution (as the constraint node).

2. We need a strategy to encode the beliefs. We propose 2 strategies:

   - Approximating the continuous distribution by a mixture Gaussian with a given

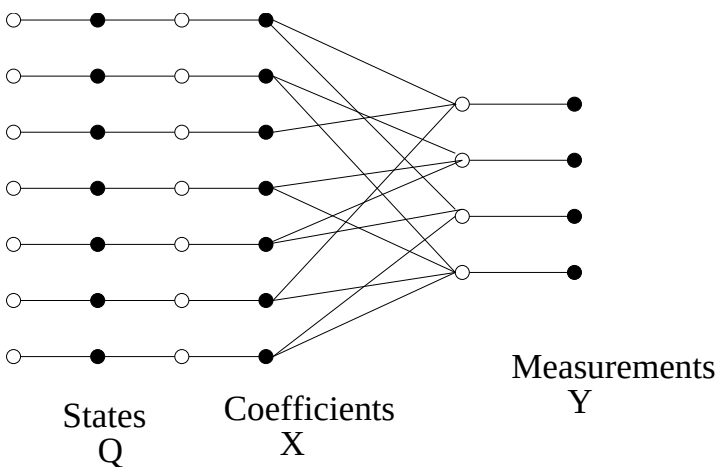

Figure 4: *Factor graph depicting the relationship between the variables involved in our CS problem. The variable nodes are depicted in black and the constraint nodes in white.*

number of components. We use the parameters of the mixture Gaussian as the messages.

- We sample the pdf's uniformly and use the samples as the messages. We present results for both schemes and discuss the pros and cons for each.

We could use alternate schemes to encode continuous distributions such as particle filtering, importance sampling and Monte Carlo methods [34]. These directions are left for future work.

Mixture Gaussian parameters as messages: In this method, we approximate a distribution by a mixture Gaussian with a maximum number of components. We then use the mixture Gaussian parameters as messages.

For both multiplication and convolution, the resulting number of components in the mixture is multiplicative. To keep the message representation tractable, employ model order reduction on mixture Gaussian distributions. We use the Iterative Pairwise Replacement Algorithm (IPRA) scheme for model order reduction [35].

The advantage of using mixture Gaussian parameters to encode continuous distributions is that the messages are short and hence do not consume large amounts of memory. However, this encoding scheme could be too limiting. The scheme is tailor mode to work for mixture Gaussian priors. In particular, this scheme cannot be extended to priors such as ℓ_p compressible signals. Also, the model order reduction – a necessary step to keep the representation tractable – introduces errors in the messages, that can affect the quality of the solution as well as impact the convergence of BP [36]. Furthermore, the model reduction algorithms (such as IPRA) used to limit the number of components in the mixture can be computationally expensive.

Samples of the pdf as messages: An alternate approach to encode the beliefs is to sample the pdf and send the samples as the messages. In this scheme, multiplication of pdf's correspond to point-wise multiplication of messages. Convolution of the pdf's can be

computed efficiently in the frequency domain.[4]

The primary advantage of using pdf samples as messages is that it can be applied to a larger class of prior distributions for the coefficients. Also, both multiplication and convolution of the pdf's can be computed efficiently. However, the drawback of this scheme is the large memory requirement: we require finer sampling of the pdf for more accurate reconstruction. As a rule of thumb, we sample the pdf's with a spacing less than σ_0, the standard deviation of the noise level in the signal. Furthermore, the mere process of sampling the pdf introduces errors in the messages that can impact accuracy and convergence of BP [36].

Stabilizing BP using damping: We employ damping using message damped belief propagation (MDBP) [37] to stabilize BP in the face of loopy graphs and inaccurate belief representations. We refer the reader to Appendix A for the reasons to use damping.

3.3 Number of measurements

The number of measurements required for reconstruction of approximately sparse signals with our decoding technique is $O(N\log(N))$ using $L = O(S^{-1})$ where $S = K/N$.

3.4 Decoding complexity

The complexity of our decoding technique for approximately sparse signals is $O(N\text{polylog}(N))$ when we use sampled pdf as messages, or $O(m^2\frac{N}{S}\log(S^{-1})\log N)$ when we use mixture Gaussian parameters as messages, where m is the maximal model order (details below).

First consider the implementation of BP with sampled pdf. Let each message be a vector of p samples. During each iteration, we perform 2 sets of operations: multiplication on the coefficient nodes and convolution on the constraint nodes. For a given coefficient node, an outgoing message is given by (4), which can be re-written in the following form to reduce complexity:

$$\begin{aligned}\mu_{v\longrightarrow c}(v) &= \prod_{u\in n(v)\setminus\{c\}} \mu_{u\longrightarrow v}(v) \\ &= \frac{\prod_{u\in n(v)}\mu_{u\longrightarrow v}(v)}{\mu_{c\longrightarrow v}(v)},\end{aligned} \tag{3}$$

as long as the condition that the denominator contains no zeros in the beliefs is satisfied. This condition is satisfies in our setting because a mixture Gaussian distribution has non-zero pdf in the real number domain. The modified form of message computation given in Equation (3) makes computation simple, because the numerator can be computed once and can be reused for all the message outgoing from the variable node v. In a similar manner,

[4]This approach of computing the FFT of the message to enable fast convolution has been used in LDPC decoding over $GF(2^q)$ using BP [19].

the messages outgoing from a constraint node (Equation (5)), can be re-written as:

$$\begin{aligned}\mu_{c\longrightarrow v}(v) &= \sum_{\sim\{v\}}\left(c(n(c))\prod_{w\in n(c)\setminus\{v\}}\mu_{w\longrightarrow c}(w)\right)\\ &= \sum_{\sim\{v\}}\left(c(n(c))\frac{\prod_{w\in n(c)}\mu_{w\longrightarrow c}(w)}{\mu_{v\longrightarrow c}(v)}\right).\end{aligned}$$

The computational complexity for the message processing at a variable node for one iteration is $O(Rp)$, because we are multiplying $R+1$ vectors of length p. For all N nodes, the combined complexity per iteration is $O(NRp)$. Now consider a constraint node associated with the measurements. We perform the convolution operation in the DFT domain, so the computational cost per node is $O(Lp + Lp\log p) = O(Lp\log p)$, to account for the FFT and the multiplications. Therefore, the combined complexity per iteration for all the M constraint nodes is $MLp\log p$. Thus the total cost per iteration is $O(NRp + MLp\log p) = O(MLp\log p)$, because $NR = ML$. For $\log N$ iterations, we get the overall complexity of $O(MLp\log p\log N)$. As a rule of thumb, we choose $p = \sigma_1/\sigma_0$ and hence the overall complexity can be expressed as $O(ML\frac{\sigma_1}{\sigma_0}\log\frac{\sigma_1}{\sigma_0}\log N)$, which is $O(N\text{polylog}(N))$.

Now let us consider the BP implementation where the messages are mixture Gaussian parameters. In this scenario, we cannot use the modified Equations (3) and (4) because it is impossible to undo the effect of multiplication. Furthermore, we perform component-reduction for products and convolutions of mixtures (using schemes like IPRA [35]) to make the representation tractable. The component-reduction makes it impossible to undo the effect of multiplication. Let m be the maximum model-order that we deal with. The complexity per coefficient node per iteration is given by $O(m^2R^2)$. For all N nodes per iteration, the complexity is $O(m^2R^2N)$. Similarly, the complexity for all M constraint nodes associated with the measurement is given by $O(m^2L^2M)$. Thus the overall complexity for $\log N$ iterations is $O(m^2LM(R+L)\log N)$. Substituting $L = O(S^{-1}) = O(N/K)$, $R = O(\log(N/K))$, and $M = O(K\log(N/K))$, the overall complexity is $O(m^2\frac{N}{S}\log(S^{-1})\log N)$.

3.5 Storage requirements

The bulk of the storage is for message representations for each edge. The number of edges is given by $ML = NR = O(N\log(N/K))$. For implementation of BP with pdf samples, the message length is p, and so the storage requirement is $O(MLp)$. For implementation of BP with mixture Gaussian parameters, the message length is m, and so the storage requirement is $O(MLm) = O(mN\log(N/K))$.

3.6 Properties of BP decoding for CS-LDPC

We now briefly describe several properties of our BP decoding approach for CS-LDPC measurements:

1. **Progressive reconstruction**: Whatever the value of M, we can always find the posterior probabilities.

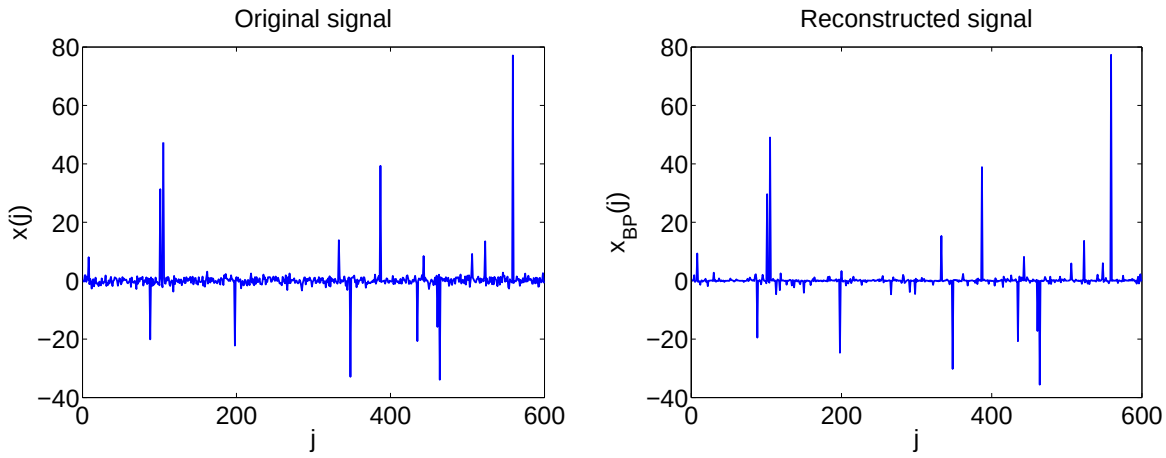

Figure 5: *Example reconstruction for* $N = 600$, $K = 18$, $M = 180$, $L = 15$.

2. **Information scalability**: If we are interested only in the states of the coefficients but not their value, i.e., estimating the support set, we would need less measurements.

3. **Robustness to noise**: We can incorporate noisy measurements into our model. This could add an additional component to the Gaussian mixtures of our current signal model.

3.7 Numerical results

We study the performance of BP for CS reconstruction for several levels of sparsity and σ_1/σ_0 ratios. Figure 5 shows a sample reconstruction of a sparse signal using BP.

Figure 6 shows sample simulation results depicting the reconstruction quality for $N = 6000$, $\sigma_1/\sigma_0 = 30$ for different values of K and L. Note that as we increase L, we require less measurements to obtain the same accuracy. However, there is an optimal value of L (L_{OPT}) beyond which there is the additional gain in performance is marginal. Furthermore, values of $L > L_{\text{OPT}}$ gives rise to divergence in BP, even with damping.

4 Variations and enhancements

While we present our techniques for sparse signals in the canonical "spike" basis, they can be generalized to the case where x is sparse in an arbitrary basis Ψ. In this case, we multiply the LDPC matrix Φ with the sparsifying matrix Ψ and use $\Phi\Psi$ to encode x as $y = (\Phi\Psi)x$. The decoder uses y to determine the signal represented in the sparsifying basis (namely Ψx), and x is then recovered by multiplying by Ψ^T. The encoding matrix $\Phi\Psi$ may not be sparse. Therefore, the encoding and decoding must perform extra computations; this extra cost is $O(N^2)$ in general. Fortunately, in many practical situations Ψ is structured (e.g., Fourier or wavelet bases) and thus amenable to fast computation. Therefore, extending our techniques to such bases is feasible.

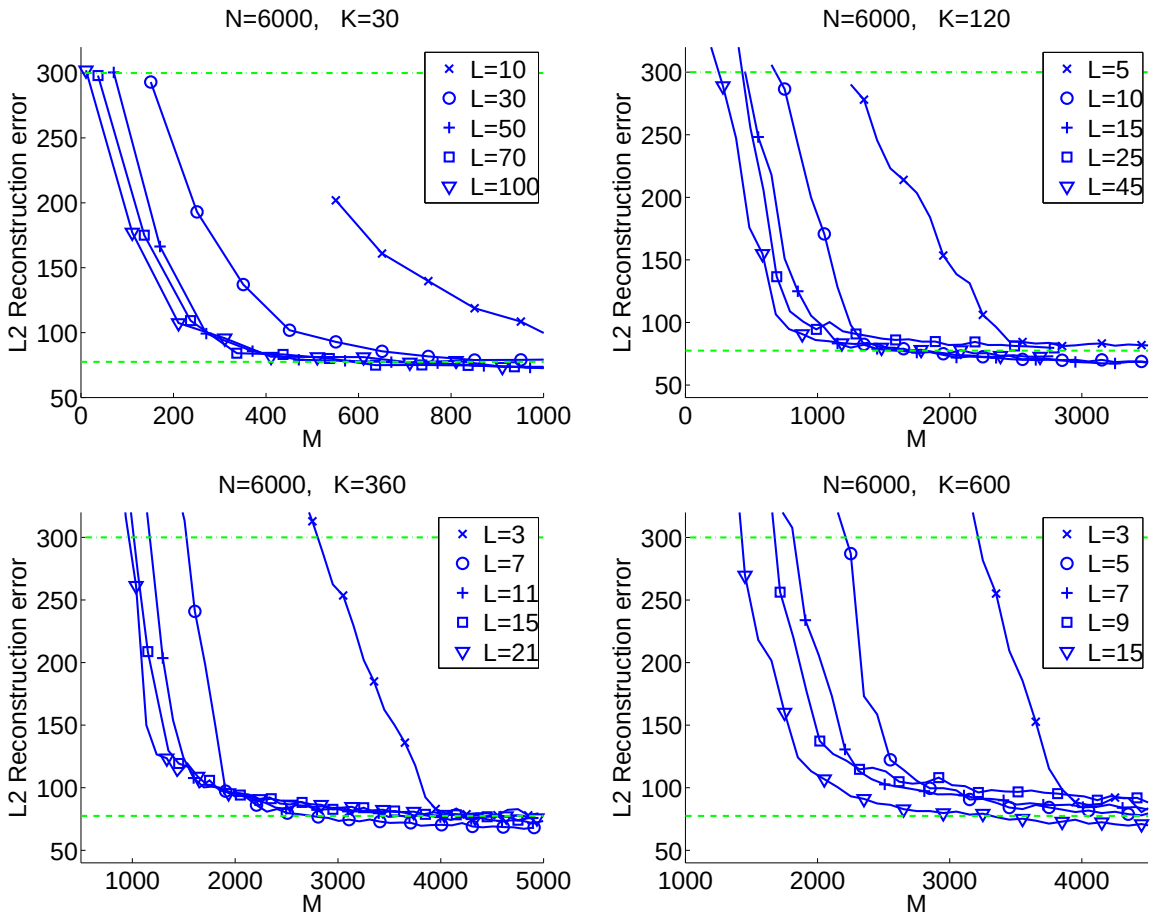

Figure 6: *Reconstruction versus the number of measurements, for different values of L using belief propagation.*

Exploiting statistical dependencies: In many signal representations, the coefficients of the representation are not statistically independent but have correlations. For example, natural images that are sparsely represented in a wavelet basis have significant coefficients that can be arranged on a tree structure, and there is strong correlation between the magnitudes of coefficients that are arranged on parent and child nodes. It is possible to decode signals from fewer measurements via an extended algorithm that allocates different probabilities of different coefficients being non-zero. For example, the wavelet coefficients of the low frequency components have a higher probability of appearing in the sparse representation than coefficients for high-frequency components [13]. The LDPC measurement technique can compensate for these different probabilities by allocating different probabilities that these coefficients are measured in any given row.

Feedback: Feedback from the decoder to the encoder can allow the measurements to be adapted according to the coefficients that have been reconstructed thus far. Consider the Sudocode scheme for strictly sparse signals as an example [18]. Notions reminiscent of closed-loop iterative doping [38] can be utilized here. The Sudocode encoder knows what it has transmitted. Consequently, the encoder can track the decoder's progress and send the

decoder "hints" regarding problematic signal coefficients. These hints will cause many other unknown coefficient values to be resolved.

5 Applications

Compressive signal acquisition: The obvious application of our ideas is to sense signals in an implicitly compressive manner using a reduced number of measurements. This approach can be especially appealing on the encoding side, because the analog measurement process can be run much slower than in a traditional signal acquisition system.

Partial reconstruction: Our decoding scheme resolves the signal several coefficients at a time. This property can be used in applications that look for information in parts of the signal, where revelation of the entire signal is not necessary. For example, problems in detection do not require complete signal revelation [39].

Lossy channels: In some applications, measurements may be lost because of transmissions over faulty channels. In such a setup, one can stream measurements until the decoder has sufficiently many to reconstruct the signal. The decoder can notify the encoder that the reconstruction is complete.

p2p and distributed file storage applications: Our techniques can be used in distributed settings where multiple encoders send measurements to a single reconstruction center. Consider peer-to-peer (p2p) applications for example. A plurality of p2p sources can all stream digital content such as video, audio, images, etc. These data are typically in compressed form and thus strictly sparse. Otherwise, we enforce strict sparsity by thresholding the transform coefficients. When an individual user wishes to download some content, these sources transmit LDPC measurements towards the user. Once the user has received enough measurements, the original signal can be reconstructed. This strategy does not require complicated synchronization of the transmissions arriving from the large number of servers. *All that is required for reconstruction is that enough measurements arrive at the decoder, no matter from which transmitter.* The decoding algorithm must also know the seeds of all the pseudo-random number generators in all the encoders. This is reminiscent of a *digital fountain* [40] but we use real valued coefficients and strictly sparse signals.

6 Conclusions

The extension of LDPC matrices from a finite alphabet setting to measurement of real-valued inputs is very promising, and leads to effective encoding and decoding schemes. In a previous paper our emphasis was reconstruction of strictly sparse signals [18]. Here we employed belief propagation techniques to reconstruct approximately sparse signals. A comparison of our results to several CS schemes proposed in the literature appears in Table 1. Our techniques can be extended to sensing and reconstructing signals that are sparse in other bases, additional signal models, and also handle noise.

Acknowledgments

Thanks to David Scott, Danny Sorensen, Yin Zhang, Marco Duarte, Michael Wakin, Mark Davenport, Jason Laska, Matthew Moravec and Elaine Hale for informative and inspiring conversations on sparsity and CS. Special thanks to Ramesh Neelamani, Alexandre de Baynast, and Predrag Radosavljevic for providing helpful hints on implementing BP.

Appendix A Overview of belief propagation

BP solves inference problems by iteratively exchanging messages over the edges of the factor graph. BP computes approximately the marginal distribution of all the variables involved in the factor graph, conditioned on the observed data. Messages are propagated over the edges of the factor graph; the messages encode the probability distribution for the variable associated with that edge. We review the basic message passing and message processing operations involved in BP.

Consider a variable node v in a factor graph and one of its neighbors, a constraint node c. Let $\mu_{v \longrightarrow c}(v)$ denote the message sent from v to c in the operation of BP, and let $\mu_{c \longrightarrow v}(v)$ denote the message sent from c to v. Let $n(v)$ denote the set of neighbors of v, and $n(c)$ the set of neighbors of c.

Belief propagation, also called the sum-product algorithm, operates according to the following simple update rules: the message (belief) from a node v on an edge e is the product of all the messages received at v on edges *other* than e. The message from node c to v is computed in a similar manner, but the constraint associated with c is applied to the product of messages and the result is marginalized for the variable associated with e. In other words, the message computations performed by belief propagation is given by

$$\mu_{v \longrightarrow c}(v) = \prod_{u \in n(v) \setminus \{c\}} \mu_{u \longrightarrow v}(v) \qquad \text{and} \tag{4}$$

$$\mu_{c \longrightarrow v}(v) = \sum_{\sim \{v\}} \left(c(n(c)) \prod_{w \in n(c) \setminus \{v\}} \mu_{w \longrightarrow c}(w) \right), \tag{5}$$

where $c(n(c))$ is the constraint on the set of variable nodes $n(c)$ (here the outer c represents the constraint and the inner c represents the node). The notation $\sim \{v\}$ represents the set of all nodes neighboring c except v, and $n(v) \setminus \{c\}$ is the set of neighbors to v excluding c.

BP enables the computation of the marginal distribution of all the variables, conditioned on the observed data. The marginal distribution for a given variable node is obtained by the product of all the most recent incoming messages along the edges connecting to that node:

$$\text{pdf}(v) = \prod_{u \in n(v)} \mu_{u \longrightarrow v}(v). \tag{6}$$

Protecting BP against loopy graphs and message errors: The results of BP converge to the exact conditional distribution (function summaries) in the ideal situation

where the following two conditions are met:
1) The factor graph is cycle-free, and
2) Messages are processed and propagated without errors.
In CS reconstruction with BP as described above, both these conditions are violated. In general, the factor graph is loopy. Also, our message propagation for CS involves message quantization because exact message representation is computationally intractable. These non-idealities may lead BP to converge to inexact function summaries, or more critically, lead BP to diverge [36, 41, 42]. We investigate the impact of these non-idealities and suggest damping strategies to protect BP.

BP may be used in factor graphs with cycles simply by following the same message update rules. In this scenario, the algorithm has no natural termination, with messages passed multiple times on a given edge. Furthermore, the result of BP operating in a factor graph with cycles cannot in general be interpreted as exact marginal distributions (even if the iterations converge). Despite this, the approximate nature of BP for graph with cycles is an acceptable price to pay for performing efficient inference. Indeed, BP is used in numerous applications in which the underlying factor graph *does* have cycles [41]. Examples of such applications include decoding of turbo codes [5] and LDPC codes [6], where BP achieves astonishing performance (within a fraction of a decibel of the Shannon limit on a Gaussian channel) [43]. The excellent performance of BP in loopy graphs motivates its use in CS reconstruction algorithms.

Furthermore, BP for CS has to deal with message quantization because exact message representation is computationally intractable. We incur message errors either by approximation by model order reduction of mixture Gaussian distributions, or by sampling the pdf. The message errors can propagate and can lead BP to diverge [42].

To protect BP in the face of loopy-graphs and message errors, we propose the use of *damping* schemes.

Damping techniques for BP: We provide a brief overview of techniques in the literature to combat divergence in the iterations of BP. Yuille proposed the Convex Concave Procedure – a class of discrete iterative algorithms that are provably convergent alternatives to BP [44]. The main idea is to obtain a discrete iterative algorithms by decomposing a cost function into a concave and a convex part. The crucial trick, is to bound the possibly concave part and solve the remaining convex problem. Heskes et al. [45] proposed the double-loop algorithm that is provably convergent for the minimization of Bethe and Kikuchi free energies [30] that represent the cost function of BP. Pretti and Pelizzola proposed a new propagation algorithm for the minimization of the cost function (Bethe free energy) for a generic lattice model with pair interactions [46]. The algorithm is shown to be more stable than belief propagation, as it reaches a fixed point also for highly frustrated systems and faster than the provably convergent double loop algorithms. More recently, Pretti [37] proposed a modified version of BP with an over-relaxed BP dynamics, where, at each step of the algorithm, the evaluation of beliefs (messages) is taken to be a weighted average between the old estimate and the new estimate. The weighted average could either be applied to the messages (resulting in the message damped BP, or MDBP) or to the estimate of the probability distribution of the variables (probability damped BP, or PDBP). The author relates MDBP and PDBP to the double-loop algorithm of Heskes et al [45]; in particular, MDBP

and PDBP are limiting cases of the double-loop algorithm. The connection between MDBP and PDBP to the double-loop algorithm sheds light about the way MDBP and PDBP work, and also provides directions for further enhancements.

References

[1] E. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Trans. Inform. Theory*, vol. 52, no. 2, pp. 489–509, Feb. 2006.

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

[3] T. M. Cover and J. A. Thomas, "Elements of information theory," *John Wiley and Sons, New York*, 1991.

[4] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, pp. 379–423,623–656, 1948.

[5] C. Berrou, A. Glavieux, and P.Thitimajshima, "Near Shannon limit error correcting coding and decoding: Turbo codes," *Proc. 1993 IEEE Int. Conf. Communications*, pp. 1064–1070, May 1993.

[6] R. G. Gallager, "Low-density parity-check codes," *IEEE Trans. Inform. Theory*, vol. 8, pp. 21–28, Jan. 1962.

[7] E. Candès and T. Tao, "Error correction via linear programming," *Found. of Comp. Math.*, 2005, Submitted.

[8] S. Chen, D. Donoho, and M. Saunders, "Atomic decomposition by basis pursuit," *SIAM J. on Sci. Comp.*, vol. 20, no. 1, pp. 33–61, 1998.

[9] D. Baron, M. B. Wakin, M. F. Duarte, S. Sarvotham, and R. G. Baraniuk, "Distributed compressed sensing," Nov. 2005, Submitted.

[10] D. Donoho and J. Tanner, "Neighborliness of randomly projected simplices in high dimensions," Mar. 2005, Preprint.

[11] D. Donoho, "High-dimensional centrally symmetric polytopes with neighborliness proportional to dimension," Jan. 2005, Preprint.

[12] J. Tropp and A. C. Gilbert, "Signal recovery from partial information via orthogonal matching pursuit," Apr. 2005, Preprint.

[13] M. F. Duarte, M. B. Wakin, and R. G. Baraniuk, "Fast reconstruction of piecewise smooth signals from random projections," in *Proc. SPARS05*, Rennes, France, Nov. 2005.

[14] D. L. Donoho, Y. Tsaig, I. Drori, and J-C Starck, "Sparse soltion of underdetermined linear equations by stagewise orthogonal matching pursuit," Mar. 2006, Preprint.

[15] G. Cormode and S. Muthukrishnan, "Towards an algorithmic theory of compressed sensing," *DIMACS Technical Report TR 2005-25*, 2005.

[16] G. Cormode and S. Muthukrishnan, "Combinatorial algorithms for compressed sensing," *DIMACS Technical Report TR 2005-40*, 2005.

[17] A. C. Gilbert, M. J. Strauss, J. Tropp, and R. Vershynin, "Algorithmic linear dimension reduction in the ℓ_1 norm for sparse vectors," Apr. 2006, Submitted.

[18] S. Sarvotham, D. Baron, and R. G. Baraniuk, "Sudocodes – fast measurement and reconstruction of sparse signals," in *Proc. International Symposium Information Theory (ISIT2006)*, Seattle, WA, July 2006.

[19] D. J. C. MacKay, "Information Theory, Inference and Learning Algorithms," *Cambridge University Press*, 2002, ISBN 0521642981.

[20] J.-C Pesquet, H. Krim, and E. Hamman, "Bayesian Approach to Best Basis Selection," *IEEE 1996 Int. Conf. Acoustics, Speech, Signal Processing (ICASSP)*, pp. 2634–2637, 1996.

[21] H. Chipman, E. Kolaczyk, and R. McCulloch, "Adaptive Bayesian Wavelet Shrinkage," *J. Amer. Stat. Assoc.*, vol. 92, 1997.

[22] M. Crouse, R. Nowak, and R. Baraniuk, "Wavelet-Based Statistical Signal Processing using Hidden Markov Models," *IEEE Transactions on Signal Processing*, vol. 46, no. 4, pp. 886–902, April 1998.

[23] F. Abramovich, T. Sapatinas, and B. W. Silverman, "Wavelet Thresholding via a Bayesian Approach," *Dept. Math., Univ. Bristol, Bristol, U.K., Tech. Rep.*, Nov. 1996.

[24] H. W. Sorenson and D. L. Alspach, "Recursive Bayesian Estimation using Gaussian sums," *Automatica*, vol. 7, pp. 465–479, 1971.

[25] D. Baron, M. B. Wakin, M. F. Duarte, S. Sarvotham, and R. G. Baraniuk, "Distributed Compressed Sensing," 2005, Submitted.

[26] G. Cooper, "The Computational Complexity of Probabilistic Inference using Bayesian Belief Networks," *Artificial Intelligence*, vol. 42, pp. 393–405, 1990.

[27] J. Pearl, "Probablistic Reasoning in Intelligent Systems: Networks of Plausible Inference," *Morgan-Kaufmann*, 1988.

[28] F. V. Jensen, "An Introduction to Bayesian Networks," *Springer-Verlag*, 1996.

[29] B. J. Frey, "Graphical Models for Machine Learning and Digital Communication," *MIT press*, 1998.

[30] J. S. Yedidia, W. T. Freeman, and Y. Weiss, "Understanding Belief Propagation and its Generalizations," *Mitsubishi Tech. Rep. TR2001-022*, Jan. 2002, http://www.merl.com/publications/TR2001-022.

[31] F. V. Jensen, "Bayesian Networks and Decision Graphs," *Springer-Verlag*, May 2002.

[32] R. G. Cowell, A. P. Dawid, S. L. Lauritzen, and D. J. Spiegelhalter, "Probabilistic Networks and Expert Systems," *Springer-Verlag*, 2003, ISBN 0387987673.

[33] F. R. Kschischang, B. J. Frey, and H-A. Loeliger, "Factor Graphs and the Sum-Product Algorithm," *IEEE Trans. Inform. Theory*, vol. 47, no. 2, pp. 498–519, Feb. 2001.

[34] "Sequential Monte Carlo Methods (Particle Filtering) Homepage," http://www-sigproc.eng.cam.ac.uk/smc.

[35] D. W. Scott and W. F. Szewczyk, "From Kernels to Mixtures," *Technometrics*, vol. 43, pp. 323–335, Aug. 2001.

[36] E. Sudderth, A. Ihler, W. Freeman, and A. S. Willsky, "Nonparametric Belief Propagation," *MIT LIDS Tech. Rep. 2551*, Oct. 2002.

[37] M. Pretti, "A Message-Passing Algorithm with Damping," *J. Stat. Mech.*, Nov. 2005.

[38] G. Caire, S. Shamai, and S. Verdu, *Noiseless data compression with low-density parity-check codes*, Advances in network information theory, Ed: Piyush Gupta, Gerhard Kramer, and Adriaan J. van Wijnggaardenl, Vol 66, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Sep 2004.

[39] M. F. Duarte, M. A. Davenport, M. B. Wakin, and R. G. Baraniuk, "Sparse signal detection from incoherent projections," in *IEEE 2006 Int. Conf. Acoustics, Speech, Signal Processing (ICASSP)*, May 2006, To appear.

[40] J. W. Byers, M. Luby, and M. Mitzenmacher, "A digital fountain approach to asynchronous reliable multicast," *IEEE J. Select. Areas Commun.*, vol. 20, no. 8, October 2002.

[41] B. J. Frey and D. J. C. MacKay, "A Revolution: Belief Propagation in Graph with Cycles," *Adv. in Neural Information Processing Systems, M. Jordan, M. S. Kearns and S. A. Solla (Eds.)*, vol. 10, 1998.

[42] A. Ihler, J. Fisher, and A. S. Willsky, "Loopy Belief Propagation: Convergence and Effects of Message Errors," *J. of Machine Learning Research*, vol. 6, pp. 905–936, May 2005.

[43] D. J. C. MacKay, "Good error-correcting codes based on very sparse matrices," *IEEE Trans. Inform. Theory*, vol. 45, pp. 399–431, Mar. 1999.

[44] A. L. Yuille, "CCCP Algorithms to Minimize the Bethe and Kikuchi Free Energies: Convergent Alternatives to Belief Propagation," *Neural Comput.*, , no. 14, pp. 1691–1722, Jul. 2002.

[45] T. Heskes, K. Albers, and B. Kappen, "Approximate Inference and Constrained Optimization," *Proc. Uncertainty in AI*, Aug. 2003.

[46] M. Pretti and A. Pelizzola, "Stable Propagation Algorithm for the Minimization of the Bethe Free Energy," *J. Phys. A: Math. Gen.*, Nov. 2003.