TURBO COMPRESSED SENSING USING MESSAGE PASSING DE-QUANTIZATION

Amin Movahed, Mark C. Reed, Neda Aboutorab

School of Eng. and Information Tech., University of New South Wales, Canberra, Australia

ABSTRACT

This paper proposes a new technique to concatenate 1-bit compressed sensing with a convolutional channel encoder for transmission of sparse signals over a memoryless AWGN channel. At the reconstruction part, an iterative decoder, referred to as turbo-CS decoder, has been proposed. At the turbo-CS decoder, the sparse signal is decoded through iterations between an a posteriori probability decoder and a soft-in/soft-out 1-bit compressed sensing decoder. By numerical experiments, we show that the turbo-CS decoder outperforms the state-of-the-art algorithms for 1-bit compressed sensing reconstruction in the presence of AWGN channel by more than 10 dB in terms of signal reconstruction performance.

Index Terms— 1-bit compressed sensing, iterative decoding, message passing de-quantization

1. INTRODUCTION

Compressed sensing (CS) enables signal acquisition through a few number of non-adaptive linear measurements by exploiting the sparsity in the signals [1, 2]. In practice, many types of signals are sparse (i.e., most of their elements are zero or near to zero) or can be approximated by sparse signals [3–5]. In CS, the number of required linear measurements to ensure exact signal reconstruction is less than the number of the elements in the original signal vector. Therefore, it can be studied as a compression method for sparse signals.

Typically, in communication applications, compressed measurements need to be quantized for further transmission. Furthermore, the transmission channel adds noise to the quantized measurements which makes the signal reconstruction process challenging. To alleviate the effect of the channel noise, a channel encoding system can be applied.

In this work, we consider sparse signal transmission over an AWGN channel. We propose an iterative decoding method which we refer to as turbo-CS decoder. At the transmitter, we apply 1-bit CS [6, 7] and a convolutional channel encoder as the outer and the inner component encoders, respectively.

At the turbo-CS decoder, *a posteriori probability* (APP) decoder and *soft-in/soft-out* (SISO) 1-bit CS decoder are applied. The iterative decoding approach of turbo-CS is the same as in turbo-codes where two decoders at the receiver exchange *a posteriori /a priori* information through a number of iterations to improve the decoding process [8–10].

The main contribution of this paper is to design a SISO 1-bit CS decoder using the message passing technique. There are two main challenges in the design of SISO 1-bit CS decoder. Firstly, the input of 1-bit CS decoders is in the form of bit while in SISO 1-bit CS decoder the input should be in the form of probabilities (soft-input). In addition, the reconstructed signal at the output of 1-bit CS decoders is in the form of real value elements with zero-mean Gaussian distribution. Thus, generating updated bit probabilities (soft-output) based on these values is not straightforward.

The SISO 1-bit decoder that we introduce in this paper is based on *message passing de-quantization* (MPDQ) algorithm in [11]. The MPDQ algorithm applies Gaussian approximation of loopy belief propagation to estimate the sparse signals from 1-bit CS measurements. Here, we significantly modify MPDQ to accept soft-input and to provide soft-output. The proposed algorithm is referred to as SISO 1-bit MPDQ.

The proposed turbo-CS decoder in this paper is dramatic improvement to the method first introduced by the authors in [12]. Here we instead use a Bayesian CS decoder and derive for the first time the non-obvious way to interface the CS decoder to the APP decoder. The SISO 1-bit CS decoders in [12, 13] are based on heuristically chosen linear/non-linear mapping functions to provide a priori information, which is not necessarily bit probability, to the APP decoder. In SISO 1-bit MPDQ proposed in this paper, the soft-output is bit probability that enables the APP decoder to decode more effectively in the next iteration of turbo-CS decoding.

Through numerical simulations we show that the turbo-CS decoder outperforms the most efficient 1-bit CS decoders, e.g. R1-BCS [14], in terms of signal reconstruction by more than 10 dB in the presence of AWGN channel with a much lower complexity.

2. SYSTEM MODEL

Assume a sparse signal vector $\mathbf{x} \in \mathbb{R}^N$ where there are only K non-zero elements in the signal. In classic CS sampling method, each measurement is obtained by an inner product of $\mathbf{x}$ with $\boldsymbol{\phi}_i \in \mathbb{R}^N$ where $\boldsymbol{\phi}_i$ is the ith row of the measuring matrix $\boldsymbol{\Phi}$. In other words, we have $y_i = \langle \boldsymbol{\phi}_i, \mathbf{x} \rangle$ for $i = 1, \ldots, M$. Therefore, the measurement vector $\mathbf{y}$ is obtained from $\mathbf{y} = \boldsymbol{\Phi}\mathbf{x}$.

It is shown that matrix $\boldsymbol{\Phi}$ satisfies the *restricted isometry property* which guarantees exact signal reconstruction with

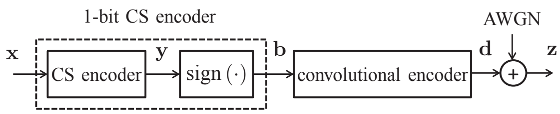


Fig. 1: Turbo-CS encoding system and the channel model

high probability when the elements of $\mathbf{\Phi}$ independently and identically follow a Gaussian distribution [15].

In quantized CS, for further storage and transmission purposes, the measurements are quantized to a number of alphabets. 1-bit CS refers to a special case where each obtained measurement is quantized by a 1-bit (two-level) quantizer [16]. In fact, 1-bit quantizer is a simple scalar sign function over the CS measurements. We denote the 1-bit scalar quantization function by $\mathcal{Q} : \mathbb{R} \to \{-1, +1\}$. We denote the 1-bit CS measurements by $\mathbf{b} \in \{-1, +1\}^M$ and we have

$$\mathcal{Q}(\mathbf{y}) = \text{sign}(\mathbf{\Phi x}) = \mathbf{b}. \tag{1}$$

In most practical applications, the sparsity level of $\mathbf{x}$ is stochastic and unknown to the reconstruction part. In this work, we assume that the elements of $\mathbf{x}$ are independent and identically distributed (i.i.d) and generated from the following Bernoulli-Gaussian distribution:

$$p(x) = \rho\sqrt{\frac{\rho}{2\pi}}e^{-\frac{\rho x^2}{2}} + (1-\rho)\,\delta(x), \tag{2}$$

where ρ is the sparsity ratio defining the density of non-zero elements in the signal.

Since CS reduces the dimension of the signal, it can be generally considered as a compression method with rate N/M. In particular, since the 1-bit CS outputs are in the form of bits, 1-bit CS is a proper choice of CS encoder for further channel encoding. As illustrated in Fig. 1, in turbo-CS encoding scheme, the obtained 1-bit CS measurements, $\mathbf{b}$, are encoded through a channel encoder. We focus on a convolutional encoder with rate M/P as a channel encoder. We denote the BPSK modulated encoded block at the output of convolutional encoder by $\mathbf{d} \in \{-1, +1\}^P$. The encoded bits then pass through a memoryless AWGN channel with noise variance $N_0 = \sigma_c^2$. Thus, we have

$$z_i = d_i + n_i \text{ for } i = 1, \ldots, P \tag{3}$$

where $n_i \sim \mathcal{N}\left(0, \sigma_c^2\right)$. We denote the channel output vector (received data) by $\mathbf{z}$.

3. TURBO-CS DECODER

In this section, we propose an iterative decoder for the mentioned transmission system in Section 2 which we refer to as turbo-CS decoder. The turbo-CS decoder comprises of an APP decoder as the inner component decoder and a 1-bit CS decoder as the outer component decoder.

3.1. A posteriori probability decoder

An APP decoder is a SISO decoder [17]. The inputs of an APP decoder are the received noisy signal ($\mathbf{z}$) and the a priori probabilities of the bits which are denoted by $p^{[\text{apri}]}(b_i)$ for $i = 1, \ldots, M$. The a priori bit probabilities in turbo-decoding are usually provided by the outer constituent decoder (in our case the 1-bit CS decoder).

APP decoder applies BCJR algorithm which generates APP of the states and transitions of a discrete time finite-state Markov process [18]. The output of APP decoder is in the form of a posteriori bit probabilities which are denoted by $p^{[\text{apos}]}(b_i)$ for $i = 1, \ldots, M$.

At the turbo-CS decoder, the a posteriori bit probabilities from APP decoder are given to a SISO inner component to update the bit probabilities and provide a priori bit probabilities to be given to APP decoder for the next turbo-CS decoding iteration. Therefore, the outer constituent decoder in turbo-CS should be in the form of SISO 1-bit CS decoder.

3.2. Soft-in/soft-out 1-bit CS decoder formulation

In this section, we derive the Bayesian formulation of SISO 1-bit CS decoding problem. For simplicity and as the first step, we assume that there is no transmission error in the channel. Therefore, we can omit the convolutional encoder in Fig. 1 and we have $\mathbf{z} = \mathbf{b}$ where $\mathbf{b}$ is obtained from (1). Hence, the conditional probability density function of the signal $\mathbf{x}$ given the received $\mathbf{b}$ is

$$\begin{aligned} p(\mathbf{x}|\mathbf{b}) &\propto p(\mathbf{b}|\mathbf{y})\,p(\mathbf{x}) \\ &\propto \prod_{i=1}^{M} p(b_i|y_i) \prod_{i=1}^{N} p(x_i), \end{aligned} \tag{4}$$

where $\propto$ denotes equality after normalization to unity. The element-wise conditional probability in (4) is obtained from

$$p(b_i|y_i) = \begin{cases} 1, & \text{if } y_i \in Q^{-1}(b_i) \\ 0, & y_i \notin Q^{-1}(b_i), \end{cases} \tag{5}$$

$$Q^{-1}(b) = \begin{cases} [0, +\infty), & \text{if } b = +1 \\ (-\infty, 0), & b = -1. \end{cases} \tag{6}$$

The distribution in (4) describes the complete statistical characterization of the decoding problem. The *minimum mean square error* (MMSE) estimator of $\mathbf{x}$ is obtained from

$$\hat{\mathbf{x}}_{\text{MMSE}}(\mathbf{b}) = \mathbb{E}(\mathbf{x}|\mathbf{b}). \tag{7}$$

However, as discussed in Section 2, turbo-CS encoded bits are contaminated with channel noise and after channel decoding the APP decoder provides probability of bits, describing the uncertainty in the decoded bits. The probability of getting bit block $\mathbf{b}$ at the output of APP decoder is

$$p^{[\text{apos}]}(\mathbf{b}) = \prod_{i=1}^{M} p^{[\text{apos}]}(b_i), \tag{8}$$

where $p^{[\text{apos}]}(b_i)$ for each element of $\mathbf{b}$ is obtained from the output of the APP decoder. Therefore, the decoding problem in the SISO 1-bit CS decoder is the estimation of $\mathbf{x}$ given the bit probabilities in (8). By obtaining the MMSE estimate for each $\mathbf{b}$ from (7) and calculating the probability of $\mathbf{b}$ from (8), we can apply the *law of total expectation* to estimate signal $\mathbf{x}$. Hence,

$$\mathbb{E}\left(\hat{\mathbf{x}}_{\text{MMSE}}(\mathbf{b})\right) = \sum_{\text{all } \mathbf{b}} \mathbb{E}\left(\mathbf{x}|\mathbf{b}\right) p^{[\text{apos}]}\left(\mathbf{b}\right) = \mathbb{E}\left(\mathbf{x}\right) = \hat{\mathbf{x}}, \quad (9)$$

where $\hat{\mathbf{x}}$ denotes the estimate of original signal $\mathbf{x}$ via the SISO 1-bit decoder.

Recently, Kamilov *et al.* introduced message passing de-quantization (MPDQ) algorithm to solve (7) [11].

In the following section, we propose SISO 1-bit MPDQ to estimate the original signal $\mathbf{x}$ to $\hat{\mathbf{x}}$ in (9).

3.3. Soft-in/soft-out 1-bit message passing de-quantization

In this section, we introduce SISO 1-bit MPDQ as a SISO 1-bit CS decoder component in turbo-CS decoder. SISO 1-bit MPDQ is a modified version of MPDQ. The main two modification steps are as follows: 1) We modify the non-linear factor update function in MPDQ to accept the bit probabilities, $p^{[\text{apos}]}(b)$, as input. 2) We extend MPDQ to provide updated bit probabilities at the output (as a priori bit probabilities for the APP decoder in the next iteration). In the following, we explain SISO 1-bit MPDQ:

Step1) Initialization: We set the initial values $\hat{\mathbf{x}}(0) = \mathbb{E}\left(\mathbf{x}\right) = \mathbf{0}$, $\mathbf{v}^x(0) = \text{var}\left(\mathbf{x}\right) = \mathbf{1}$, $\hat{\mathbf{s}}(0) = \mathbf{0}$ where expected and variance vectors are set with respect to prior distribution of $\mathbf{x}$ in (2). The number inside the parenthesis denotes the number of corresponding iteration and $\mathbf{0}$ and $\mathbf{1}$ denote all-zero and all-one vectors, respectively.

Step 2) Linear factor update functions:

$$\mathbf{v}^u\left(t\right) = \left(\mathbf{\Phi} \bullet \mathbf{\Phi}\right) \mathbf{v}^x\left(t-1\right), \quad (10)$$

$$\hat{\mathbf{u}}\left(t\right) = \mathbf{\Phi}\hat{\mathbf{x}}\left(t-1\right) - \mathbf{v}^u\left(t\right) \bullet \hat{\mathbf{s}}\left(t-1\right). \quad (11)$$

Step 3) Modified non-linear factor update functions:

$$\hat{s}_i\left(t\right) = \mathcal{E}_{\text{F}}\left(p^{[\text{apos}]}\left(b_i\right), \hat{u}_i\left(t\right), v_i^u\left(t\right)\right), \quad (12)$$

$$v_i^s\left(t\right) = \mathcal{V}_{\text{F}}\left(p^{[\text{apos}]}\left(b_i\right), \hat{u}_i\left(t\right), v_i^u\left(t\right)\right), \quad (13)$$

where $\bullet$ denotes element-wise multiplication. $\hat{u}_i$ and v_i^u denote the ith elements of $\hat{\mathbf{u}}$ and $\mathbf{v}^u$, respectively. $\mathcal{E}_{\text{F}}$ and $\mathcal{V}_{\text{F}}$ are functions over scalar values:

$$\mathcal{E}_{\text{F}}\left(p(b), \hat{u}, v^u\right) = \frac{1}{v^u}\left(\mathbb{E}\left(z\right) - \hat{u}\right), \quad (14)$$

$$\mathcal{V}_{\text{F}}\left(p(b), \hat{u}, v^u\right) = \frac{1}{v^u}\left(1 - \frac{\text{var}\left(z\right)}{v^u}\right). \quad (15)$$

The above scalar functions, which are the modified versions of factor update functions in [11], consider all the quantization symbols with different probabilities as input. Therefore, this step allows the input of MPDQ to be in the form of bit probabilities (soft-input).

From the *law of total expectation* and the *law of total variance* we have

$$\mathbb{E}\left(z\right) = \sum_{b=-1,+1} p\left(b\right) \mathbb{E}\left(z|z \in Q^{-1}(b)\right), \quad (16)$$

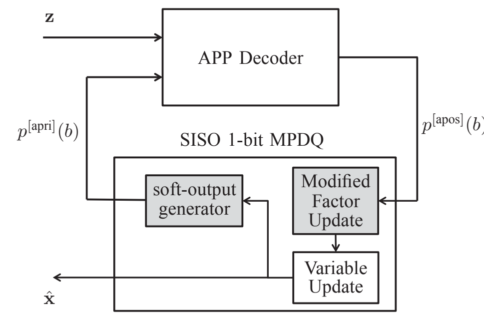


Fig. 2: The turbo-CS decoder: the proposed modules (Step 3 and Step 7) are shown by the dark color.

$$\begin{aligned}\text{var}\left(z\right) &= \mathbb{E}_b\left(\text{var}\left(z|b\right)\right) + \text{var}_b\left(\mathbb{E}\left(z|b\right)\right) = \quad (17)\\ &= \mathbb{E}_b\left(\text{var}\left(z|b\right)\right) + \mathbb{E}_b\left(\mathbb{E}\left(z|b\right)^2\right) - \mathbb{E}\left(z\right)^2,\end{aligned}$$

$$\mathbb{E}_b\left(\text{var}\left(z|b\right)\right) = \sum_{b=-1,+1} p\left(b\right) \text{var}\left(z|z \in Q^{-1}(b)\right), \quad (18)$$

$$\mathbb{E}_b\left(\mathbb{E}\left(z|b\right)^2\right) = \sum_{b=-1,+1} p\left(b\right) \mathbb{E}\left(z|z \in Q^{-1}(b)\right)^2. \quad (19)$$

The conditional expectation and variance in (16), (18) and (19) are with respect to a priori distribution $z \sim \mathcal{N}\left(\hat{u}, v^u\right)$.

Step 4) Linear variable update functions:

$$\mathbf{v}^r\left(t\right) = \left(\left(\mathbf{\Phi} \bullet \mathbf{\Phi}\right)^T \mathbf{v}^s\left(t\right)\right)^{-1}, \quad (20)$$

$$\hat{\mathbf{r}}\left(t\right) = \hat{\mathbf{x}}\left(t-1\right) + \mathbf{v}^r\left(t\right) \bullet \left(\mathbf{\Phi}^T\hat{\mathbf{s}}\left(t\right)\right), \quad (21)$$

where $\hat{\mathbf{s}}$ and $\mathbf{v}^s$ are vector representation of (12) and (13).

Step 5) Non-linear variable update functions:

$$\hat{x}_i\left(t\right) = \mathcal{E}_{\text{V}}\left(\hat{r}_i\left(t\right), v_i^r\left(t\right)\right), \quad (22)$$

$$v_i^x\left(t\right) = \mathcal{V}_{\text{V}}\left(\hat{r}_i\left(t\right), v_i^r\left(t\right)\right), \quad (23)$$

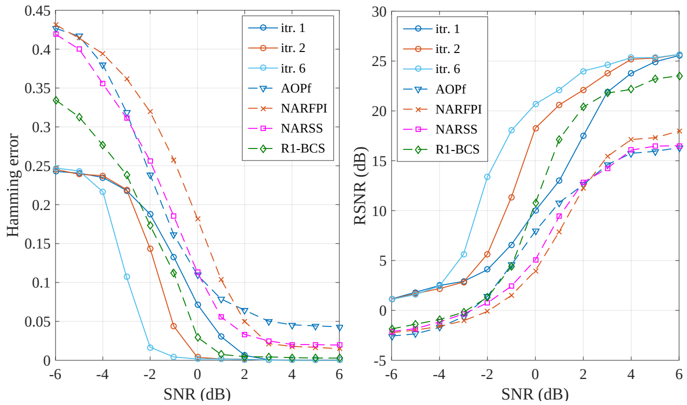
where $\hat{r}_i$ and v_i^r denote the ith elements of $\hat{\mathbf{r}}$ and $\mathbf{v}^r$, respectively, and the above functions are defined over scalar values. We have $\mathcal{E}_{\text{V}}\left(\hat{r}, v^r\right) = \mathbb{E}\left(x|\hat{r}\right)$ and $\mathcal{V}_{\text{V}}\left(\hat{r}, v^r\right) = \text{var}\left(x|\hat{r}\right)$. The above expected value and variance are with respect to $\hat{r} = x + w$ where $w \sim \mathcal{N}(0, v^r)$.

Step 6) Termination of iterations: The iteration number increments by one and the algorithm proceeds to Step 2 until convergence. Through the above updating steps the original signal $\mathbf{x}$ is estimated with $\hat{\mathbf{x}}$.

Step 7) Soft-output generator: In order to update a posteriori bit probabilities, $p^{[\text{apos}]}(b)$, we need to find the distribution of the estimate of $\mathbf{y}$. We apply mean and variance of $\mathbf{x}$ in (22) and (23). Since $\mathbf{y}$ is a linear combination of $\mathbf{x}$, the estimate of original $\mathbf{y}$ is obtained from $\hat{\mathbf{y}} = \mathbb{E}\left(\mathbf{y}\right) = \mathbf{\Phi}\hat{\mathbf{x}}$ and $\mathbf{v}^y = \text{var}\left(\mathbf{y}\right) = \left(\mathbf{\Phi} \bullet \mathbf{\Phi}\right) \mathbf{v}^x$.

From Central limit theorem, the estimated y_i at the output of SISO 1-bit MPDQ has a Gaussian distribution with mean $\hat{y}_i$ and variance v_i^y for $i = 1, \cdots, M$. Hence, we can update the probability of each corresponding bit at the output of SISO 1-bit CS decoder from

$$p^{[\text{apri}]}\left(b_i\right) = \text{Q}\left(\frac{-b_i\hat{y}_i}{\sqrt{v_i^y}}\right), \quad (24)$$



(a) (b)

Fig. 3: Comparison of the turbo-CS decoder reconstruction performance with the other 1-bit CS reconstruction algorithms in terms of Hamming errors (a), and RSNR (b).

where $\mathrm{Q}(x) = \frac{1}{\sqrt{2\pi}} \int_x^{\infty} \exp\left(\frac{-u^2}{2}\right) du$.

Remark: the modifications in SISO 1-bit MPDQ are in Step 3 and Step 7; The rest of the algorithm is derived from MPDQ. The reader is referred to [11] and [19] for more details.

As shown in Fig. 2, in each turbo-CS decoder iteration (itr.), APP decoder provides a posteriori bit probabilities, $p^{[\text{apos}]}(b)$, by processing a priori bit probabilities, $p^{[\text{apri}]}(b)$ and the received signal $\mathbf{z}$. Then, the a posteriori bit probabilities are given to SISO 1-bit MPDQ as soft-input. SISO 1-bit MPDQ estimates the original signal and updates the bit probabilities as soft-output. The output bit probabilities are given to APP decoder as updated a priori information for the next turbo-CS iteration. In the first iteration of turbo-CS decoder, the a priori bit probabilities are initialized with 0.5, i.e., each bit has an equal chance of being a 1 or 0.

4. PERFORMANCE OF THE TURBO-CS DECODER

In this section, we compare the performance of the turbo-CS decoder with the most recent 1-bit CS algorithms designed for signal reconstruction in the presence of noise (AOP-f [20], NARFPI [21], NARSS [22] and R1-BCS [14]) through numerical experiments. Here, we assume that signal $\mathbf{x}$ is generated based on the distribution in (2). We set $N = 1000$ and $\rho = 0.01$. The measuring matrix $\mathbf{\Phi}$ has i.i.d zero-mean Gaussian entries with variance $1/M$ where $M = 500$.

As a channel encoder, we use a recursive systematic rate 1/3 convolutional encoder with polynomial coefficients $\mathbf{G} = \left(1, \frac{37}{23}, \frac{33}{23}\right)_8$. The coded bits then pass through a memoryless AWGN channel. We define *signal to noise ratio* (SNR) as the channel noise measure by $\mathrm{SNR} = \frac{\mathrm{E}_b}{\mathrm{N}_0} = \frac{1}{\sigma_c^2}$, where E_b is the average power of a bit at the output of 1-bit CS encoder.

At the receiver, AOP-f, NARFPI and NARSS require knowledge of the channel noise in terms of bit flips, L. We provide average of L to these decoders from a table that maps each SNR to L and is calculated through simulation. In addition, we input ρN as sparsity level of the signal to the AOP-f decoder. For AOP-f, NARFPI, NARSS and R1-BCS, the bits at the receiver are provided by applying a simple comparator function over the output bit probabilities of the APP decoder (i.e., bits with probabilities greater than 0.5 are mapped to $+1$ and the rest of the bits are mapped to -1). Other setup parameters of these algorithms are the same as in their references. The turbo-CS decoder is applied to reconstruct the signal through 6 turbo-CS iterations. The number of inner iterations in SISO 1-bit MPDQ is set to $t_{\max} = 100$.

Table 1: Comparison of complexity of different 1-bit CS algorithms in terms of running time (in seconds) on a PC with intel Core i7, 3.40 GHz processor and 8G RAM.

Algorithm	Turbo CS	AOPf	NARFPI	NARSS	R1-BCS
Running time (s)	0.18 per itr.	0.14	1.21	0.06	63

We measure the performance of the signal reconstruction by *received signal to noise ratio* (RSNR) and Hamming error. Former is defined by $\mathrm{RSNR} = \frac{\mathbb{E}\left(\|\mathbf{x}\|_2^2\right)}{\mathbb{E}\left(\|\hat{\mathbf{x}}-\mathbf{x}\|_2^2\right)}$ where $\|\mathbf{x}\|_2$ denotes the ℓ_2-norm of vector $\mathbf{x}$. In order to have a fair comparison, we normalize the sparse signals before 1-bit CS encoding to have unit energy (i.e., $\|\mathbf{x}\|_2 = 1$). The same normalization is done after signal reconstruction at the output of the decoders[1]. In addition, we define Hamming errors by $\|\mathrm{sign}(\mathbf{\Phi}\mathbf{x}) - \mathrm{sign}(\mathbf{\Phi}\hat{\mathbf{x}})\|_0 / M$, where $\|\cdot\|_0$ denotes ℓ_0-norm (i.e., number of non-zero elements in the argument).

The averaged reconstruction performance of the 1-bit CS decoders over 1000 Monte Carlo realizations is illustrated in Fig. 3. As it is shown, the turbo-CS decoder outperforms the other algorithms after two iterations (itr. 2) both in terms of Hamming errors (Fig. 3-a) and RSNR (Fig. 3-b). The turbo-CS decoder outperforms R1-BCS by more than 10 dB in RSNR after 6 iterations (itr. 6) at SNR$= 0$ (Fig. 3-b).

In Table 1 the complexity of the decoders is shown in terms of running time in seconds. It can be seen that the turbo-CS decoder is significantly less complex than R1-BCS, that has the closest reconstruction performance to turbo-CS.

5. CONCLUSION

We considered transmission of sparse signals over an AWGN channel by serially concatenating the 1-bit-CS encoder and the convolutional channel encoder. A turbo-CS decoder is proposed, where a SISO 1-bit MPDQ and an APP decoder are applied. We introduced the SISO 1-bit MPDQ decoder which accepts soft-values at the input and provides updated soft-values at the output. Through numerical simulations, we showed that the turbo-CS decoder outperforms the most efficient recent 1-bit CS algorithms in terms of RSNR (by more than 10 dB) and with a lower complexity when the measurements are contaminated with AWGN.

[1]NARFPI, NARSS and AOP-f require the sparse signal to be on unit ℓ_2-ball to reconstruct the signal efficiently.

6. REFERENCES

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

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

[3] J. Bobin, J.-L. Starck, and R. Ottensamer, "Compressed sensing in astronomy," *IEEE Trans. Signal Process.*, vol. 2, no. 5, pp. 718–726, 2008.

[4] M. Lustig, D. Donoho, and J. M. Pauly, "Sparse MRI: The application of compressed sensing for rapid MR imaging," *Magnetic resonance in medicine*, vol. 58, no. 6, pp. 1182–1195, 2007.

[5] J. Haupt, W. U. Bajwa, M. Rabbat, and R. Nowak, "Compressed sensing for networked data," *IEEE Signal Process. Mag.*, vol. 25, no. 2, pp. 92–101, 2008.

[6] P. T. Boufounos and R. G. Baraniuk, "1-bit compressive sensing," in *Proc. Annual Conf. Inf. Sciences Syst. (CISS)*, Princeton, NJ, Mar. 2008, pp. 16–21.

[7] L. Jacques, J. N. Laska, P. T. Boufounos, and R. G. Baraniuk, "Robust 1-bit compressive sensing via binary stable embeddings of sparse vectors," *IEEE Trans. Inf. Theory*, vol. 59, no. 4, pp. 2082–2102, Apr. 2013.

[8] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes. 1," in *Proc. IEEE Int. Conf. Commun. (ICC)*, vol. 2, Geneva, Switzerland, May 1993, pp. 1064–1070.

[9] C. Berrou and A. Glavieux, "Near optimum error correcting coding and decoding: Turbo-codes," *IEEE Trans. Commun.*, vol. 44, no. 10, pp. 1261–1271, 1996.

[10] J. Hagenauer, "Source-controlled channel decoding," *IEEE Trans. Commun.*, vol. 43, no. 9, pp. 2449–2457, 1995.

[11] U. S. Kamilov, V. K. Goyal, and S. Rangan, "Message-passing de-quantization with applications to compressed sensing," *IEEE Trans. Signal Process.*, vol. 60, no. 12, pp. 6270–6281, 2012.

[12] A. Movahed and M. C. Reed, "Iterative detection for compressive sensing: Turbo CS," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Sydney, Australia, Jun. 2014, pp. 4518–4523.

[13] A. Movahed, M. C. Reed, and N. Aboutorab, "Hyperbolic mapping method for turbo compressive sensing," in *IEEE 2016 Australian Commun. Theory Workshop (AusCTW)*, to appear.

[14] F. Li, J. Fang, H. Li, and L. Huang, "Robust one-bit bayesian compressed sensing with sign-flip errors," *IEEE Signal Process. Lett.*, vol. 22, no. 7, pp. 857–861, 2015.

[15] E. J. Candès, "The restricted isometry property and its implications for compressed sensing," *Comptes Rendus Mathematique*, vol. 346, no. 9, pp. 589–592, May 2008.

[16] J. N. Laska and R. G. Baraniuk, "Regime change: Bit-depth versus measurement-rate in compressive sensing," *IEEE Trans. Signal Process.*, vol. 60, no. 7, pp. 3496–3505, Jul. 2012.

[17] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "A soft-input soft-output app module for iterative decoding of concatenated codes," *IEEE Commun. Lett.*, vol. 1, no. 1, pp. 22–24, 1997.

[18] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inf. Theory*, vol. 20, no. 2, pp. 284–287, Mar. 1974.

[19] S. Rangan, "Generalized approximate message passing for estimation with random linear mixing," *arXiv preprint arXiv:1010.5141*, 2010.

[20] M. Yan, Y. Yang, and S. Osher, "Robust 1-bit compressive sensing using adaptive outlier pursuit," *IEEE Trans. Signal Process.*, vol. 60, no. 7, pp. 3868–3875, 2012.

[21] A. Movahed, A. Panahi, and G. Durisi, "A robust RFPI-based 1-bit compressive sensing reconstruction algorithm," in *Proc. IEEE Inf. Theory Workshop (ITW)*, Lausanne, Switzerland, Sep. 2012, pp. 567–571.

[22] A. Movahed, A. Panahi, and M. C. Reed, "Recovering signals with variable sparsity levels from the noisy 1-bit compressive measurements," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Florence, Italy, May 2014, pp. 6454–6458.