

An Improved Reconstruction Algorithm for Compressive Sensing for Sparse Binary Signals using Progressive l_q Reweighting and Bounding

Jiangtao Wen, Zhuoyuan Chen, Yuxing Han, John Villasenor, and Shiqiang Yang

Abstract—An algorithm for reconstructing a sparse binary signal using compressive sensing is presented. In contrast with most previous approaches which are based on the l_q norm, with q fixed, the method described here utilizes progressive lowering of q and bounding of the reconstructed signal to the range $[0,1]$ as the iterations progress. This enables faster and more robust reconstruction at a reduced computational complexity.

Index Terms—compressive sensing, sparse binary signals, bounding, l_p reweighting

I. INTRODUCTION

Compressive Sensing (CS) has been receiving significant recent attention due to the potential to reconstruct a sparse signal from a small number of observations. For example, the pioneering papers of Donoho [1], Candès [2], [3], [4], [5], [6], [7], Romberg [3], [6], [8] and Tao [2], [3], [6] laid the groundwork for CS, and proved [4] that it is possible to recover a sparse signal sampled at sub-Nyquist rates with very high probability.

A signal is called “ k -sparse” if it contains k non-zero elements. The reconstruction problem can be represented mathematically through

$$\min \|x\|_0 \quad s.t. \quad Ax = y \quad (1)$$

where A is an $m \times n$ matrix (the “sensing matrix”) with $m < n$, $x \in R^n$, $y \in R^m$. As the direct solution is NP-hard, the problem is often approximated by

$$\min \|x\|_q \quad s.t. \quad Ax = y \quad (2)$$

where q is often taken to be 1.

To better approximate the formulation in equation (1), [9] presents an iterative scheme based on reweighting the values in a computation described in terms of the l_1 norm in order to approximate the l_0 norm:

$$\min \sum_{i=1}^n w_i |x_i| \quad s.t. \quad Ax = y \quad (3)$$

where x_i is the i th element of the solution, $i = 1 \dots n$. For each iteration, $w_i = \frac{1}{|\tilde{x}_i| + \epsilon}$, where \tilde{x}_i is the i th element of the solution found in the previous iteration, and ϵ is a

Jiangtao Wen, Zhuoyuan Chen and Shiqiang Yang are with Tsinghua University, Beijing, China (email: jtwen@tsinghua.edu.cn, chen-zhuoyuan07@mails.tsinghua.edu.cn, yangshq@tsinghua.edu.cn)

Yuxing Han and John D. Villasenor are with the University of California, Los Angeles, CA 90095-1594 (email: ericahan@ee.ucla.edu, villa@ee.ucla.edu)

small positive number used to improve the robustness of the numerical implementation. Experiments described in [9] with both 1-dimensional and 2-dimensional signals showed that reweighting scheme leads to better results than traditional l_1 based algorithm recovery.

Addressing equation (2) using the l_2 -norm enables a closed form solution. If the l_1 -norm is used, the solution is no longer closed form, but remains convex. For values of q in the range $0 \leq q < 1$ equation (2) becomes non-convex and the optimization is not guaranteed to converge. Despite these challenges, recent work by Foucart and Lai [10], Davies and Bribonval [11], and Chartrand and Staneva [12] described approaches in the regime $0 \leq q < 1$ and showed that proper choice of q within this range can lead to computationally practical solutions providing improved reconstruction.

II. ALGORITHM DESCRIPTION

In this paper, we describe an improved reconstruction algorithm for compressive sensing of sparse binary signals using $0 \leq q < 1$. As in [9] and [12], we utilize reweighting. However, in contrast to [9] and [12], we modify the value of q to become progressively lower as the algorithm progresses. In addition, the knowledge that the original signal was binary is utilized to bound the reconstructed signal to the range $[0, 1]$ after each iteration. By combining the progressive lowering of q with this bounding operation, the quality of the reconstruction can be significantly improved, and the computational complexity reduced.

The core of the algorithm consists of two nested loops. The inner loop successively modifies the ϵ value used in the reweighting, while the outer loop modifies the norm q . For each q, ϵ pair, the following minimization is performed

$$\min \sum_{i=1}^n \frac{|x_i|}{|\tilde{x}_i|^{1-q} + \epsilon} \quad s.t. \quad Ax = y \quad (4)$$

where \tilde{x}_i is the i -th element of the solution found in the previous iteration as noted earlier, and ϵ is used in equation (4) to handle with the elements of \tilde{x}_i that are zero. To avoid letting the values of ϵ distort the target function and the optimization, ϵ should usually be small. Because both \tilde{x}_i and ϵ are constants in the subsequent iteration, equation (4) remains a linear programming problem that can be solved using mature techniques such as interior point methods [13].

The difference between equation (4) and (3) is that in equation (4), the term $|\tilde{x}_i|$ is raised to the power of $1 - q$

TABLE I

SNR EVOLUTION AS THE ALGORITHM PROGRESSES. THE ORIGINAL SIGNAL IS A BINARY SIGNAL OF LENGTH $n=8192$ WITH SPARSNESS $k=1150$ AND $m = 2800$ NOISELET COEFFICIENTS. $\epsilon_r = \epsilon_0/\alpha^{(r-1)}$, WITH $\epsilon_0=1/2$. AN SNR OF $+\infty$ INDICATES THAT COMPLETE CONVERGENCE HAS BEEN ACHIEVED WITH THE RECONSTRUCTED SIGNAL EQUAL TO THE ORIGINAL SIGNAL.

	ϵ_1	ϵ_2	ϵ_3	ϵ_4	ϵ_5
$q = 0.8$	11.31	11.48	11.68	12.06	12.55
$q = 0.6$	12.46	12.60	12.88	13.40	14.14
$q = 0.4$	14.43	15.16	$+\infty$	$+\infty$	$+\infty$
$q = 0.2$	$+\infty$	$+\infty$	$+\infty$	$+\infty$	$+\infty$
$q = 0$	$+\infty$	$+\infty$	$+\infty$	$+\infty$	$+\infty$

in the weight for $|x_i|$ in the summation in equation (4), as a result, as \tilde{x} approaches the solution x^* , and ϵ approaches 0, the target function of equation (4) approaches the l_q norm of x^* , as opposed to the l_0 norm of x^* in equation (3).

The algorithm is initialized using $q = q_0$ and $\epsilon = \epsilon_1$. For each q , the optimization problem of equation (4) is subjected to a first round of iterative processing containing I iterations. After this and each subsequent round of processing, bounding of the values in the reconstructed signal is performed by replacing all values exceeding 1 by 1, and replacing all negative values by 0. ϵ is then set to ϵ_1/α , and a second round is performed, with bounding to the range $[0,1]$ performed again in each iteration. This process continues through a total of R rounds, with the r th round utilizing $\epsilon_r = \epsilon_1/\alpha^{(r-1)}$. After R rounds are completed, q is reduced by Δq , ϵ is reinitialized to ϵ_1 , and the process repeats through $q = 0$. In the experiments described in the next section, the algorithm parameters were set at $I=15$, $q_0 = 0.8$, $\epsilon_1 = 1/2$, $R=5$, $\Delta q = 0.2$, and $\alpha = 3$. It is possible to repeat the above until $q = 0$, though as discussed in the next section, convergence is usually reached relatively early in the process. This approach contrasts with earlier efforts both in the mechanisms for modifying ϵ and q as well as in the use of bounding.

In each iteration we use the interior point method to find a solution that satisfies the Karush-Kuhn-Tucker conditions, which are the necessary and sufficient conditions for convex optimality [13]. As mentioned above, the algorithm is divided into rounds, each corresponding (in the results presented here) to 15 iterations using the same ϵ value. Each iteration is an iteration in a standard interior point method (also called the barrier method) for solving the linear programming problem of equation (4) using the current q and ϵ values. Since the objective function is twice-differentiable the algorithm uses the Newton method to find an update Δx to the current solution x , while making sure that the new solution Δx remains in the interior of the feasible set and is a better solution to equation (4). Each iteration involves similar computational complexity to an iteration in [9] and [12]. As discussed below, however, since the algorithm of the present paper converges more quickly, it requires fewer overall iterations and thus lower complexity.

TABLE II

SNR EVOLUTION WITH $k = 950$, $m = 2800$ AND $n = 8192$. WE COMPARE ALGORITHM OF THE PRESENT PAPER WITH THE ALGORITHMS IN [9], [12] WITH $l_{0,2}$ AND l_{0+} WITHOUT BOUNDING.

	ϵ_1	ϵ_2	ϵ_3	ϵ_4	ϵ_5
$q = 0.8$	15.27	26.38	$+\infty$	$+\infty$	$+\infty$
$q = 0.6$	$+\infty$	$+\infty$	$+\infty$	$+\infty$	$+\infty$
...					
fix $q = 0$					
[9], [12]	13.26	13.20	13.13	13.13	13.13
fix $q = 0.2$					
[9], [12]	13.18	13.09	13.02	13.00	12.99

III. EXPERIMENTAL RESULTS

To explore the performance of the algorithm, we tested 100 different randomly generated realizations of sparse binary signals with dimensionality $n = 8192$, sparseness $k = 1150$, and with $m = 2800$ noiselet samples. In 95 out of the 100 cases, full convergence was achieved and the original signal was successfully constructed with no errors. In 5 of the cases, the algorithm did not converge. Table I shows the SNR evolution as the algorithm progresses for a representative converging realization; the same trend was observed for all the other converging realizations. In the table, the SNR is defined as $10 \log(P_S/MSE)$, where P_S is the power of the peak input signal value (1 in our case) and MSE is the mean square error between the input and the reconstruction. An SNR of $+\infty$ indicates that complete convergence has been achieved with the reconstructed signal equal to the original signal. Since the decoder does not have the original signal with respect to which to calculate the SNR, to avoid getting “trapped” in a locally optimum, the decoder can in theory complete the double loop over q and ϵ values. However, in practice, the iteration for the optimization of equation (4) after perfect reconstruction is usually several orders of magnitude faster than the computations before the input is correctly recovered.

In experiments conducted using Matlab on an Intel Pentium D dual-core processor running at 2.80 GHz and 1.0 GMB memory, for input binary signals with $k = 1150$, the typical time to perform one iteration was about 1.1 seconds, and the average total time for all iterations to achieve perfect recovery was approximately 300 seconds. Although in theory, a local minimum that is close to the global optimal solution to equation (4) may also cause the iteration speed to decrease significantly, in practice the decoder may use the information about the iteration speed as an exit criterion for the algorithm. As implemented here, the algorithm has a worst case number of iterations encountered when five different q values are used (starting at $q = 0.8$ and decrementing by steps of $\Delta q = 0.2$) with 75 iterations for each q . In many cases where the algorithm presented here converges, the algorithms in [9] and [12], do not, even after hundreds of rounds of iterations.

As is clear from Table I, the SNR increases with each step until reaching full convergence, which for this example occurs after the 3th round of $q = 0.4$. The lowering of q as the algorithm progresses is a key aspect enabling convergence; this contrasts for example with the approach in [12], which

TABLE III

AN EXAMPLE OF SNR EVOLUTION AT $k = 935$, $m = 2800$ AND $n = 8192$. BOTH ALGORITHMS SUCCEEDED IN RECOVERING THE ORIGINAL SIGNAL. IN [9] AND [12], THE q -NORM IS USED WITHOUT BOUNDING

	15	30	45	60	75
this paper	90.39	$+\infty$	$+\infty$	$+\infty$	$+\infty$
fix $q = 0$					
[9], [12]	14.20	14.79	15.59	$+\infty$	$+\infty$
fix $q = 0.2$					
[9], [12]	14.15	14.73	15.71	$+\infty$	$+\infty$

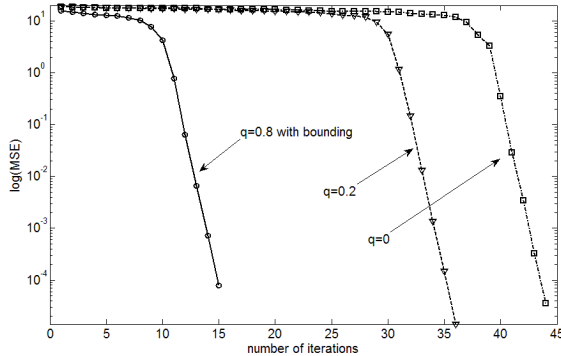


Fig. 1. Typical algorithm convergence as a function of iteration. The vertical axis gives the MSE between the original and reconstructed signal. Curves for both the present algorithm (curve marked with circles) as well as the method of [9] and [12]. (curves marked using triangles and squares) are shown.

is based on a fixed q . If q is not adjusted and the number of rounds at a fixed q is simply increased, rounding errors and other factors can impede convergence, as was observed in [9]. With a fixed q , [9] and [12] achieving convergence requires increasing the number of samples m or reducing the sparseness by increasing the number k of non-zero elements of the input.

An example using $k = 950$, $m = 2800$, and $n = 8192$ is shown in Table II. In this example, the present algorithm achieves convergence after two rounds, while the approach in [9] and [12] does not converge. More generally, for all realizations examined for which [9] and [12] converges, the present algorithm does as well, although the converse does not hold true. When both the algorithm of the present paper and that of [9] and [12] are able to successfully reconstruct a signal, it is of significance to compare the rate of convergence. Table III examines the SNR evolution for a representative case with $k = 930$, $m = 2800$ and $n = 8192$. For the algorithm of this paper, convergence is achieved after a total of 20 iterations over two rounds, while that in [9] and [12] converges after a total of 60 iterations over four rounds. It is also instructive to show the convergence speed in terms of mean square error (MSE) as a function of iteration number. A typical example is shown in Fig. 1, which shows curves for both the present algorithm as well as the method from [9] and [12].

We next explore what happens if two of the critical methods introduced here - the bounding to $[0,1]$, and the reduction of q - are not utilized. Table IV is partitioned into two halves.

TABLE IV

SNR EVOLUTION USING THE SAME SIGNAL AS IN TABLE II, WITH $k = 950$, $m = 2800$ AND $n = 8192$. THE TOP PARTITION GIVES THE ALGORITHM AS DESCRIBED IN THIS PAPER (PROVIDING THE SAME SNR VALUES AS THE TOP ROW OF TABLE II). THE BOTTOM PARTITION SHOWS THE LACK OF CONVERGENCE WHEN THE ALGORITHM IS IMPLEMENTED WITHOUT BOUNDING.

	ϵ_1	ϵ_2	ϵ_3	ϵ_4	ϵ_5
with bounding					
$q = 0.8$	15.27	26.38	$+\infty$	$+\infty$	$+\infty$
...					
without bounding					
$q = 0.8$	13.14	13.10	13.05	13.01	13.00
$q = 0.6$	13.08	13.03	13.00	12.99	12.97
$q = 0.4$	13.02	12.94	12.90	12.89	12.87
$q = 0.2$	12.93	12.84	12.79	12.77	12.77
$q = 0.0$	12.82	12.70	12.66	12.66	12.66

TABLE V

SNR EVOLUTION WITH $k = 1150$, $m = 2800$ AND $n = 8192$. THE TOP PARTITION GIVES THE ALGORITHM AS DESCRIBED IN THIS PAPER. THE BOTTOM PARTITION SHOWS THE LACK OF CONVERGENCE WHEN THE ALGORITHM IS IMPLEMENTED WITHOUT SUCCESSIVE REDUCTION OF q .

	ϵ_1	ϵ_2	ϵ_3	ϵ_4	ϵ_5
reduced q					
$q = 0.8$	11.31	11.48	11.68	12.06	12.55
$q = 0.6$	12.46	12.60	12.88	13.40	14.14
$q = 0.4$	14.43	15.16	$+\infty$	$+\infty$	$+\infty$
...					
fix $q = 0.2$					
	11.19	11.24	11.34	11.47	11.60
	11.73	11.86	11.92	11.94	11.96
	11.98	11.94	11.81	11.61	11.67
...					
fix $q = 0$					
	11.20	11.26	11.34	11.43	11.60
	11.75	11.81	11.93	11.96	11.96
	11.97	11.82	11.96	12.08	12.01
...					

In the upper half, the algorithm is implemented normally. In the lower half, the bounding step is removed, though q is successively reduced in the normal manner. As is clear from the table, the absence of bounding prevents convergence. Table V explores the impact of successive reduction of q . Again, the table is partitioned into two halves. In the upper half the algorithm is operated normally, while in the lower half q is fixed at 0 or 0.2. This illustrates the value of the q reduction in the algorithm described in this paper.

Table VI examines the case where instead of bounding after each round, the reconstructed values are subject to a "hard decision" such that they are quantized to 0 or 1. As might be expected, this discarding of "soft" information contained in the values between 0 and 1 significantly damages convergence. The top half of Table VI shows the algorithm when operated with normal bounding, and the bottom half shows the case where quantization to 0 or 1 is performed after each round.

In addition, we also compared the performance of bounding with a selective quantization scheme in which quantization is carried out for values above 0.9 and below 0.1 (i.e. the

TABLE VI

AN EXAMPLE AT $k = 1000$, $m = 2800$ AND $n = 8192$. THE TOP PARTITION OF THE TABLE SHOWS THE ALGORITHM AS DESCRIBED ABOVE; THE BOTTOM PARTITION SHOWS THE INFERIOR CONVERGENCE BEHAVIOR IF THE SIGNAL IS QUANTIZED TO 0 AND 1 AFTER EACH ITERATION.

	ϵ_1	ϵ_2	ϵ_3	ϵ_4	ϵ_5
this paper					
$q = 0.8$	13.76	14.73	74.64	$+\infty$	$+\infty$
...					
binary quantization					
$q = 0.8$	12.21	12.35	12.39	12.40	12.41
$q = 0.6$	12.53	12.57	12.67	12.66	12.66
$q = 0.4$	12.64	12.67	12.70	12.70	12.70
$q = 0.2$	12.73	12.71	12.73	12.72	12.72
$q = 0.0$	12.72	12.72	12.71	12.70	12.70

values between 0.1 and 0.9 are not processed). Although for input signals with smaller values of k the selective quantization method converges faster than bounding, for larger values of k , bounding is more likely to recover the input than the selective quantization method. None of the algorithms we tested were able to recover signals when the number of samples m is smaller than $2k$, where k is the the sparseness of the input signal in any significant manner.

IV. ANALYSIS AND CONCLUSIONS

This paper has described an improved algorithm for reconstructing sparse binary signals using compressive sensing. The algorithm is based on the reweighted l_q norm optimization algorithm of [9], but with the important additional operation of bounding in each round of the interior-point method iteration, and progressive reduction of q . Experimental results confirm that the algorithm performs well both in terms of the ability to recover an input signal as well as in terms of speed. We also found that both the progressive reduction and the bounding are integral to the improvement in performance. We have also examined the performance of the algorithm when the bounding operation is replaced by quantization of floating point numbers to 0 or 1, as well as selectively quantizing a subset of the floating point numbers. Experimental results show that both alternatives worsen the performance of the algorithm. Future work includes extending this approach to Gaussian distributed, as opposed to binary inputs.

REFERENCES

- [1] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM Review*, vol. 43, no. 1, pp. 129–159, 2001.
- [2] E. Candès and T. Tao, "Decoding by linear programming," *Information Theory, IEEE Transactions on*, vol. 51, no. 12, pp. 4203–4215, Dec. 2005.
- [3] E. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: exact signal reconstruction from highly incomplete frequency information," *Information Theory, IEEE Transactions on*, vol. 52, no. 2, pp. 489–509, Feb. 2006.
- [4] Candès, "The restricted isometry property and its implications for compressed sensing," *Comptes rendus-Mathématique*, Feb. 2008.
- [5] E. J. Candès and J. Romberg, "Sparsity and incoherence in compressive sampling," *Inverse Problems*, vol. 23, no. 3, pp. 969–985, June 2007.
- [6] E. J. Candès, J. K. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," *Communications on Pure and Applied Mathematics*, vol. 59, no. 8, pp. 1207–1223, 2006.

- [7] E. J. Candès and M. B. Wakin, "An introduction to compressive sampling [a sensing/sampling paradigm that goes against the common knowledge in data acquisition]," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 21–30, March 2008.
- [8] J. Romberg, "Imaging via compressive sampling [introduction to compressive sampling and recovery via convex programming]," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 14–20, March 2008.
- [9] E. J. Candès, "Enhancing sparsity by reweighted l_1 minimization," *Journal of Fourier Analysis and Applications*, vol. 14, no. 5-6, pp. 877–905, Dec. 2008.
- [10] S. Foucart and M. Lai, "Sparsest solutions of underdetermined linear systems via l_q -minimization for $0 < q \leq 1$," *Applied and Computational Harmonic Analysis*, vol. 26, no. 3, pp. 395–407, May 2009.
- [11] M. E. Davies and R. Gribonval, "Restricted isometry constants where l_p sparse recovery can fail for $0 < q \leq 1$," *Information Theory, IEEE Transactions on*, 2009.
- [12] R. Chartrand and V. Staneva, "Restricted isometry properties and non-convex compressive sensing," *Inverse Problem*, vol. 24, no. 3, June 2008.
- [13] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, March 2004.