Zero-point attracting projection algorithm for sequential compressive sensing

Yang You, Jian Jin, Wei Duan, Ningning Liu, Yuantao Gu,* and Jian Yang

Abstract

Sequential Compressive Sensing, which may be widely used in sensing devices, is a popular topic of recent research. This paper proposes an online recovery algorithm for sparse approximation of sequential compressive sensing. Several techniques including warm start, fast iteration, and variable step size are adopted in the proposed algorithm to improve its online performance. Finally, numerical simulations demonstrate its better performance than the relative art.

Keywords: Compressive sensing, sparse signal recovery, sequential, online algorithm, zero-point attracting projection

1   Introduction

Compressive sensing (CS) [1, 2] is a recently proposed concept that enables sampling below Nyquist rate, without (or with little) sacrificing reconstruction quality. Based on exploiting the signal sparsity in typical domains, CS methods can be used in the sensing devices, such as MR imaging [3] and AD conversion [4], where the devices have a high cost of acquiring each additional sample or a high requirement on time. Therefore, as the sparsity level is often not known a priori, it can be very challenging to use CS in practical sensing hardware.

Sequential compressive sensing [5] can effectively deal with the above problems. Sequential CS considers a scenario where the observations can be obtained in sequence, and computations with observations are performed to decide whether these samples are enough. Consequently, it is allowed to recover the signal either exactly or to a given tolerance from the smallest possible number of observations. There have been several recovery algorithms for sequential CS. Asif [6] solved the problem by homotopy method. Garrigues [7] discussed the Lasso problem with sequential observations.

This work extends a recent proposed zero-point attracting projection (ZAP) algorithm [8] to the scenario of sequential CS. ZAP employs an approximate l_0 norm as the sparsity constraint and updates in the solution space. Comparing with the existing algorithms, it needs fewer measurements and lower computation complexity. Therefore the new algorithm can provide a much more appropriate solution for practical sensing devices, which is validated by numerical simulations.

2 Background

2.1 Compressive sensing

Suppose $\mathbf{x}$ is an unknown sparse signal, which is N-length but has only K nonzero entries, where $K \ll N$. In their ice-breaking contributions, Candes et al suggested to measure $\mathbf{x}$ with under-determined observations, i.e. $\mathbf{y} = \mathbf{A}\mathbf{x}$, where $\mathbf{A}$ consists of random entries and has much fewer rows than columns. They also proved that l_0 norm or l_1 norm constraint optimization can successfully recover the unknown signal with overwhelming probability,

$$\hat{\mathbf{x}} = \arg \min_{\mathbf{x}} \|\mathbf{x}\|_{0 \text{ or } 1}, \text{ subject to } \mathbf{y} = \mathbf{A}\mathbf{x}. \tag{1}$$

There are many methods proposed to solve (1), of which concerned in this work is ZAP.

2.2 Zero-point attracting projection

ZAP iteratively searches the sparsest result in solution space. The recursion starts from the least square optimal solution, $\mathbf{x}(0) = \mathbf{A}^\dagger \mathbf{y}$, where $\mathbf{A}^\dagger = \mathbf{A}^T (\mathbf{A}\mathbf{A}^T)^{-1}$ denotes the Pseudo-inverse matrix of $\mathbf{A}$. In the nth iteration, the solution is first updated along the negative gradient direction of a sparse penalty,

$$\tilde{\mathbf{x}}(n+1) = \mathbf{x}(n) - \kappa \cdot \nabla g(\mathbf{x}(n)), \tag{2}$$

where $g(\cdot)$ denotes a sparse constraint function and κ denotes the step size. In the reference, an approximate l_0 norm is employed and the corresponding ith entry of $\nabla g(\cdot)$ is

$$\{\nabla g\}_i(\mathbf{x}) = \begin{cases} \alpha \cdot \text{sgn}(x_i) - \alpha^2 x_i & |x_i| \leq \frac{1}{\alpha}; \\ 0 & \text{elsewhere}, \end{cases} \tag{3}$$

where α is a controlling parameter and it is readily recognized that the penalty tends to l_0 norm as α approaches to infinity. Then $\tilde{\mathbf{x}}(n+1)$ is projected back to the solution space to satisfy the observation constraint,

$$\mathbf{x}(n+1) = \mathbf{P}\tilde{\mathbf{x}}(n+1) + \mathbf{Q}, \tag{4}$$

where $\mathbf{P} = \mathbf{I} - \mathbf{A}^\dagger \mathbf{A}$ is defined as projection matrix and $\mathbf{Q} = \mathbf{A}^\dagger \mathbf{y}$. Equation (2) appears that an attractor locates at the zero-point is pulling the iterative solution to be sparser, as explains the first part of the algorithm's name. The last part comes from (4), which means that $\tilde{\mathbf{x}}(n+1)$ is projected back to the solution space.

2.3 Sequential compressive sensing

Imagining a scenario that the samples are measured in realtime. At time m, an m-length measurement vector $\mathbf{y}_m = \mathbf{A}_m\mathbf{x}$ is collected and utilized to solve the sparsest solution by (1). If the available measurements are not enough to recover the original sparse signal, a new sample $y_{m+1} = \mathbf{a}_{m+1}^{\mathrm{T}}\mathbf{x}$ is generated at time $m+1$, where $\mathbf{a}_{m+1}$ denotes the sampling weight vector. Thus the problem becomes solving $\mathbf{y}_{m+1} = \mathbf{A}_{m+1}\mathbf{x}$, where

$$\mathbf{y}_{m+1} = \begin{bmatrix} \mathbf{y}_m \\ y_{m+1} \end{bmatrix}, \qquad \mathbf{A}_{m+1} = \begin{bmatrix} \mathbf{A}_m \\ \mathbf{a}_{m+1}^{\mathrm{T}} \end{bmatrix}.$$

Obviously, it is a waste of resources if the recovery algorithm is re-initialized without the utilization of earlier estimate, i.e. the available result at time m. Consequently, the basic aim of sequential compressive signal reconstruction is to find an effective method of refining $\hat{\mathbf{x}}_{m+1}$ based on the information of $\hat{\mathbf{x}}_m$.

3 Online ZAP for sequential compressive sensing

For conciseness, the detailed iteration procedure of online ZAP is provided in Tab.1. It can be seen that online ZAP has two recursions. The inner iteration is to update the solution by ZAP with the given measurements. The outer iteration is for sequential input. In order to improve the performance, several techniques are used in online ZAP and they are discussed in the following subsections.

Table 1: The Procedure of Online ZAP

Input: $\alpha, \kappa_0, \eta_1, \eta_2, T, Q$;

Initialize online ZAP: $\mathbf{x}_0(0) = 0, m = 0, n_0 = 0$.

Repeat: (for time instant m);

 for the new $\mathbf{a}_{m+1}$ and y_{m+1}, calculate $\mathbf{\Gamma}_{m+1}$ by (5)

  and then produce $\mathbf{P}_{m+1}$ and $\mathbf{Q}_{m+1}$;

 decrease step size by $\kappa_{m+1} = \eta_1\kappa_m$;

 $m = m + 1$;

 Initialize ZAP: $n = 0, \mathbf{x}_m(0) = \mathbf{x}_{m-1}(n_{m-1}), \kappa = \kappa_m$;

 Repeat: (for the nth iteration of ZAP);

  Update $\tilde{\mathbf{x}}_m(n+1)$ with the zero attraction by (2) and (3);

  Project $\mathbf{x}_m(n+1)$ back to the solution space by (4);

  if $\|\mathbf{x}_m(n+1)\|_1 > \|\mathbf{x}_m(n)\|_1$ and $\kappa > \kappa_m/Q$,

   decrease the step size by $\kappa = \eta_2\kappa$;

  $n = n + 1$;

 Until: $n > T$ or $\kappa < \kappa_m/Q$.

 $n_m = n - 1$;

Until: online ZAP stop criterion is satisfied.

3.1 Warm Start

ZAP works in an iterative way to produce a sparse solution via recursion in the solution space. In the online scenario, the previous estimate can be used to initialize the incoming iteration, i.e. $\mathbf{x}_{m+1}(0) = \mathbf{x}_m(n_m)$, where n_m denotes the maximum iteration number at time m.

3.2 Fast Iteration

The Pseudo-inverse matrix $\mathbf{A}^\dagger$ plays an important role in the recursion of (4). Considering the high computational cost of matrix inverse operation, $\mathbf{A}^\dagger$ is generally prepared before iterations. However, in the online scenario, $\mathbf{A}_m$ becomes time-dependent and $\mathbf{A}_m^\dagger = \mathbf{A}_m^{\mathrm{T}} \left(\mathbf{A}_m \mathbf{A}_m^{\mathrm{T}} \right)^{-1}$ need to be recalculated in each time instant. In order to reduce the complexity, the Pseudo-inverse matrix is updated iteratively.

Define $\mathbf{\Gamma}_m = \left(\mathbf{A}_m \mathbf{A}_m^{\mathrm{T}} \right)^{-1}$, which is already available after time m. Consequently, as the new sample is arriving, using basic algebra one has the recursion

$$
\begin{aligned}
\mathbf{\Gamma}_{m+1} &= \left[\begin{bmatrix} \mathbf{A}_m \\ \mathbf{a}_{m+1}^{\mathrm{T}} \end{bmatrix} \begin{bmatrix} \mathbf{A}_m^{\mathrm{T}} & \mathbf{a}_{m+1} \end{bmatrix} \right]^{-1} = \begin{bmatrix} \mathbf{\Gamma}_m^{-1} & \boldsymbol{\alpha}_m \\ \boldsymbol{\alpha}_m^{\mathrm{T}} & \beta_m \end{bmatrix}^{-1} \\
&= \begin{bmatrix} \mathbf{\Gamma}_m + \theta_m \mathbf{\Gamma}_m \boldsymbol{\alpha}_m \boldsymbol{\alpha}_m^{\mathrm{T}} \mathbf{\Gamma}_m & -\theta_m \mathbf{\Gamma}_m \boldsymbol{\alpha}_m \\ -\theta_m \boldsymbol{\alpha}_m^{\mathrm{T}} \mathbf{\Gamma}_m & \theta_m \end{bmatrix},
\end{aligned}
\tag{5}
$$

where

$$
\boldsymbol{\alpha}_m = \mathbf{A}_m \mathbf{a}_{m+1}, \qquad \beta_m = \mathbf{a}_{m+1}^{\mathrm{T}} \mathbf{a}_{m+1}, \qquad \theta_m = \frac{1}{\beta_m - \boldsymbol{\alpha}_m^{\mathrm{T}} \mathbf{\Gamma}_m \boldsymbol{\alpha}_m}.
$$

3.3 Variable step size

As the step size in gradient descent iterations, the parameter κ controls a tradeoff between the speed of convergence and the accuracy of the solution. In order to improve the performances of the proposed algorithm, the idea of variable step size is taken into consideration. The control scheme is rather direct: κ is initialized to be a large value after new sample arrived, and reduced by a factor as long as the iteration is convergent. The reduction is repeated several times until κ is sufficiently small. Since the algorithm has two recursions, we employ η_1 and η_2 to denote the decreasing speed of outer and inner iteration, respectively. In addition, κ is no longer decreased when the step size is rather small.

3.4 Stop Rules

There are two kinds of recursions requiring stop rules in the online ZAP algorithm. Firstly, after the mth sample arrived, $\mathbf{x}_m(n)$ iterates with n to produce the best estimate based on the m measurements. The inner iteration should stop after the algorithm reaches steady state, which means the sparsity penalty starts increasing. Consequently, the inner
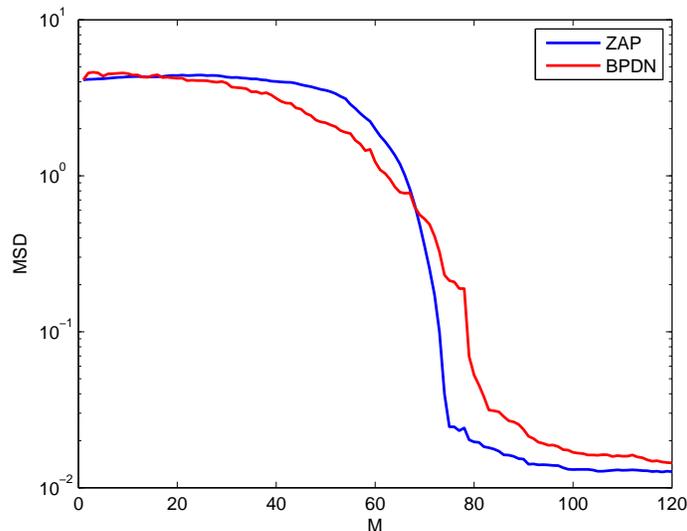
Figure 1: The reconstruction MSD versus the number of measurements M.

recursion stops (a) when the number of reductions of κ reaches one-Qth of its initial value or (b) when the number of iterations reaches the bound T.

Secondly, as soon as the sparse signal is successfully reconstructed, the following samples are no longer necessary and the sensing procedure stops. Therefore, the outer recursion stops when the estimate error is below a particular value ε.

4   Experiment and Discussion

Computer simulation is presented in this section to verify the performance of the proposed algorithm compared with typical sequential CS reconstruction algorithm for solving BPDN problem [6], whose MATLAB code can be downloaded from the website [9]. In the following experiment, the entries of each row of **A** are independently generated from normal distribution. The locations of K nonzero coefficients of sparse signal **x** are randomly chosen with uniform distribution $[1, N]$. The corresponding nonzero coefficients are Gaussian with mean zero and unit variance. The system parameters are $N = 256$ and $K = 20$. The number of measurements M increases form 1 to 120. The parameters for BPDN are set as the recommended values by the author. The parameters for online ZAP are $\alpha = 1$, $\kappa_0 = 0.02$, $\eta_1 = 0.99$, $\eta_2 = 0.8$, $T = 50$, $Q = 2000$. The simulation is repeated ten times, then Mean Square Derivation (MSD) between the original signal and reconstruction signal as well as the average running time calculated.

Figure 1 shows MSD curve according to M. As can be seen, the performance of ZAP is better than that of BPDN. When the sparse signal is recovered successfully, the number of measurements BPDN needs is larger than 80, while the number ZAP algorithm needs is less
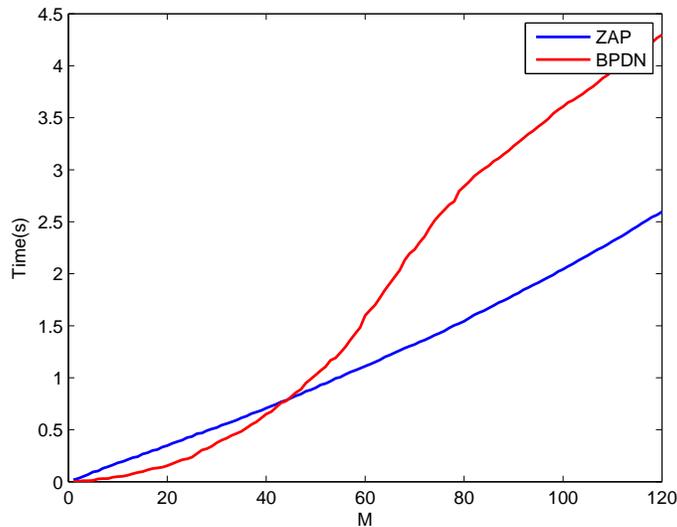
Figure 2: The CPU running time versus the number of measurements M.

than 80. Figure 2 demonstrates the CPU running time as M increases. Again, ZAP has the better performance. The CPU time of BPDN is twice than that of ZAP for successful recovery (according to Fig.1, here M is chosen as 80 for comparison).

5   Conclusion

We have introduced in this paper a new online signal reconstruction algorithm for sequential compressive sensing. The proposed algorithm extends ZAP to sequential scenario. And in order to improve the performance, some methods, including the warm start and variable step size, are adopted. The final experiment indicates that the proposed algorithm needs less measurements and less CPU time than the reference algorithm.

References

[1] D. L. Donoho, "Compressed sensing," *IEEE Trans. on Information Theory*, 52(4), pp.1289-1306, April 2006.

[2] E. Candes, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Trans. Inform. Theory*, 52:489-509, 2006.

[3] M. Lustig, D. Donoho, and J. Pauly, "Sparse MRI: The application of compressed sensing for rapid MR imaging," *Magnetic Resonance in Medicine*, 58(6) pp. 1182 - 1195, December 2007.

[4] M. Mishali, Y. C. Eldar, and J. A. Tropp, "Efficient sampling of sparse wideband analog signals," *CCIT Report #705* , October 2008.

[5] D. Malioutov, S. Sanghavi, and A. Willsky, "Compressed sensing with sequential observations," *ICASSP*, pp. 3357-3360, April 2008.

[6] M. S. Asif and J. Romberg, "Streaming measurements in compressive sensing: ℓ_1 filtering," *42nd ACSSC*, 26-29, Oct. 2008.

[7] P. Garrigues and L. Ghaoui, "An homotopy algorithm for the lasso with online observations," *NISP conference*, 2008.

[8] J. Jin, Y. Gu, and S. Mei, "A stochastic gradient approach on compressive sensing signal reconstruction based on adaptive filtering framework," *IEEE Journal of Selected topics in Signal Processing*, 4(2), pp.409-420, 2010.

[9] http://users.ece.gatech.edu/~sasif/homotopy/