Compressed Sensing in Internet of Things

Seminar Report

by

Nagma Samreen Khan

153079030

under the guidance of

Prof. Kumar Appaiah

Department of Electrical Engineering

Indian Institute of Technology, Bombay

April 2016

Abstract

Internet of Things is the emerging technology of this era and involves quite a lot of data which is being sensed and transmitted over the internet. In most cases the sensed data is sparse, so we need efficient sampling techniques which exploits this property and samples and transmits much less data i.e. if $x \in \mathbb{R}^n$ is the quantity being originally sampled then sample $y \in \mathbb{R}^m$ where $m << n$. Also efficient reconstruction algorithms are needed to reconstruct the original quantity with small error at the receiver's end.

1 Introduction

1.1 Internet of Things

Internet of Things or commonly referred to as IoT defines a paradigm in which 'everyday' devices can be connected together in a network and thus can communicate with each other over the internet. Moreover the connected devices are equipped with 'identifying', 'sensing' and 'processing' [1] capabilites which enable them to achieve some objective or complete some assigned task. Nowadays IoT is widely used in applications like healthcare, utilities, transport, etc. The revolution in the field of IoT has enabled these devices to have interaction capabilites too i.e. they can be controlled, actuated and commanded. [2].

The components of IoT broadly are Hardware, Software and Architecture. Hardware refers to the component/s or the device/s which constitue the system which has been designed to achieve some objective, Software refers roughly to the support which enables the devices in the system to exchange and process data inspite of their heterogeniety and Architecture provides the specifications so that information transfer and processing as well as networking of devices can be done in a standard fashion.

IoT has made its mark in healthcare by providing remote monitoring of patients health, in industries by providing efficient tracking of goods, in 'smart homes' by monitoring and controlling resource consumption. And it holds the promise of 'smart cities' where pollution level to traffic will be monitored and controlled [1] [2].

1.2 Compressed Sensing

With increase in the number of connected devices around us, the amount of data transmitted over the internet will naturally increase. Also with increase in the number of sensors, large amounts of data will be generated which will have to be processed, stored and transmitted over the internet in a timely fashion. So consumption of resources by the entire system will increase. This is where Compressed or Compressive Sensing comes in.

The simplest way to define Compressed Sensing is that we measure and transmit much less information compared to what would have been actually measured by the sensors and that can also be reconstructed with a small error at the receiver's end. This opens up a world of possibilities as it leads to saving time and resources and thus data transmission can be done much faster. The only prerequisite is that the data has to *sparse* or *compressible* in some transform domain. These concepts are defined mathematically in the next section.

2 Mathematical Background to Compressed Sensing

A vector $x \in \mathbb{R}^n$ is said to be *k-sparse* if there are k non-zero coefficients in x. And the vector $x \in \mathbb{R}^n$ is said to be *compressible* if it is well-approximated by its k-term *sparse* approximation z and approximation error is small. Also *support* of a vector x i.e. $supp(x)$ is defined as the set of indices for which the value of x is non-zero [3].

In this discussion we are assuming that x is *sparse* or *compressible* in its original domain and no transform is needed. We can make this assumption without loss of generality because any signal in $\mathbb{R}^n$ can be represented in terms on a basis matrix (assumed orthonormal) [4]. Also whenever a vector is mentioned as *k-sparse* then either it actually is or it is *compressible* and can be well approximated by its *k-sparse* representation.

Now suppose $x \in \mathbb{R}^n$ is the original data measured by the sensors and is known to be *k-sparse* then we don't measure x directly but instead measure $y \in \mathbb{R}^m$, $m << n$ where,

$$y = \phi x \tag{1}$$

where ϕ is $m \times n$ matrix, called the *measurement* matrix, and is required to satisfy the *restricted isometry property*(RIP) [5] and also it is required to be *incoherent* with the basis matrix ψ i.e. the rows of ϕ cannot sparsely represent the columns of ψ [4]. Satisfying the RIP property simply means that the distance (here l_2 norm) between two *k-sparse* vectors is preserved during transformation with matrix ϕ.

It has been shown that a $m \times n$ matrix ϕ having iid entries from a Gaussian distribution possesess *RIP* with high probability if $m \geq c \ \ log(\frac{n}{k})$, where c is small constant [4].

3 Recovery Algorithms

Given a measurement vector $y \in \mathbb{R}^m$, which may be noisy, reconstructing the original *k-sparse* vector $x \in \mathbb{R}^n$ requires searching over all $\binom{n}{k}$ subspaces. The challenge for the recovery algorithms is to identify which k columns of ϕ span the subspace in which y lies [6].

3.1 CoSaMP

In this discussion, one of the popular reconstruction algorithm that is CoSaMP will be discussed, which belongs to the class of Greedy Algorithms. The number of measurements needed for proper signal reconstruction is $m = O(klog(\frac{n}{k}))$ [5].

3.1.1 Notation

Before giving the mathematical background of CoSaMP , certain notations have to be introduced [5]. For $x \in \mathbb{R}^n$ and k a positive integer, x_k implies retaining only the k largest values of x and setting the rest equal to zero. Given the set $T \subseteq 1, 2, ...n$, ϕ_T refers to the column submatrix of ϕ where columns are chosen as per the values in set T. For the same set T, $x|_T$ is the restriction of x to the set T, i.e.

$$x|_T = \begin{cases} x_i, & \text{if } i \in T \\ 0, & otherwise. \end{cases} \tag{2}$$

And pseudo-inverse of a matrix A is defined as $A^\dagger = (A^*A)^{-1}A^*$.

3.1.2 Algorithm Description

The CoSaMP algorithm steps are 1)Forming the signal proxy and then pruning it to its k largest components; 2)Support merger; 3) Estimation using least squares; 4)Pruning the estimate to retain only the largest k entries; 5)Updating the residue. Here pruning refers to restricting x to x_k and residual refers to the part of the signal which has not been estimated yet.

The pseudo-algorithm is described in Algorithm 1 [5].

Algorithm 1 CoSaMP algorithm

Input: $\phi \in \mathbb{R}^{n\times m}$(measurement matrix), $y \in \mathbb{R}^m$(measurements), k(sparsity number)

Output: $x \in \mathbb{R}^n$(recovered signal)

$x^0 \leftarrow 0$ ▷ Initialisation
$y^r \leftarrow y$
$i \leftarrow 0$

repeat
  $i \leftarrow i + 1$
  $x^{temp} \leftarrow \phi^* y^r$ ▷ Forming signal proxy
  $\Omega \leftarrow supp(x_{temp\ 2k})$ ▷ Restricting the proxy to its 2k largest component
  $T \leftarrow \Omega \cup supp(x^{i-1})$ ▷ Support Merger
  $b|_T \leftarrow \phi_T^\dagger y$ ▷ Estimating signal using least squares on the merged support set
  $x^i \leftarrow b_k$
  $y^r \leftarrow y - \phi x^k$ ▷ Calculating residual
until $x^i - x^{i-1} \leq \epsilon$ ▷ For some small ϵ
return $x = \phi^\dagger y$ ▷ Returning the best estimate when the algorithm converges

3.2 Dynamic Group Sparsity

It might happen in certain cases that the k nonzero coefficients of x are not randomly distributed but clustered into some groups, but this grouping keeps on changing and thus is called *Dynamic Group Sparsity* or DGS.

Definition 3.1. A vector $x \in \mathbb{R}^n$ is defined as *dynamic group sparse* where x is either *k-sparse* or can be well approximated by its *k-sparse* approximation and the k non-zero coefficients of x are grouped into $q \in \{1, 2, ...k\}$ groups.

In DGS, we don't need to search over all $\binom{n}{k}$ subspaces, so the number of measurements required decreases to $m = O(k + qlog(\frac{n}{q}))$ [7] [8]. This is quite an improvement compared to CoSaMP .

The DGS algorithm is similar to CoSaMP algorithm described in Section 3.1.2, except steps 1 and 4. In DGS, pruning is done by using DGS approximation pruning which is described in Algorithm 2 [7]. For DGS approximation pruning, the neighbours of the signal as well as the weights assigned to those neighbours need to be specified.

Algorithm 2 DGS approximation algorithm

Input: $x \in \mathbb{R}^n$ (vector to be pruned), k (sparsity number), τ (no. of neighbours of x), $N_x \in \mathbb{R}^{n\times\tau}$ (values of x's neighbours), $w \in \mathbb{R}^{n\times\tau}$ (weights assigned to neighbours).

Output: $supp(x,k)$ (Support set of pruned signal)

for $i = 1,..,n$ **do**
$z(i) = x^2(i) + \sum_{t=1}^{\tau} w^2(i,t)N_x^2(i,t)$
▷ Calculating total contribution of a value and it's neighbours
end for

$\Omega \in \mathbb{R}^n \leftarrow z_k$ ▷ Pruning z to it's k largest components
return $supp(x,k) \leftarrow \Omega$

---

4 Verification of Recovery Algorithms through Simulation

4.1 Implementation

Both the CoSaMP and DGS algorithm have been implemented for verification purpose and to understand how number of measurements affect the convergence of algorithm and the amount of error.

For test purpose, 1D measurement set having values ± 1 was generated randomly with $n = 512$, $k = 64$ and $q = 4$. Measurement matrix ϕ was generated choosing iid values from Gaussian distribution and then it was row normalized to have unit magnitude, τ was taken as 2, the neighbours of the signal generated were taken as τ neighbouring values i.e. neighbours of $x(i)$ were $x(i-1)$ and $x(i+1)$ and the weights w was set as 0.5 for all neighbours [7].

First CoSaMP was tried on it and then DGS. Results are decribed in the next section.

4.2 Discussion

CoSaMP needs $m = O(klog(\frac{n}{k}))$ for successful signal recovery, $m = (factor + 1)k$ was used where $factor = log(\frac{n}{k})$ and the algorithm converged in about 6 iterations, as per the *halting critereon* described in Algorithm 1.

Whereas for DGS, the number of measurements needed for convergence were $m = factor\ k$ and it took about 12 iterations to converge. Error is of the same order as in CoSaMP . The generated dataset x is shown in Fig.1 and the error with CoSaMP and DGS is shown in Fig.2 and Fig.3 respectively.

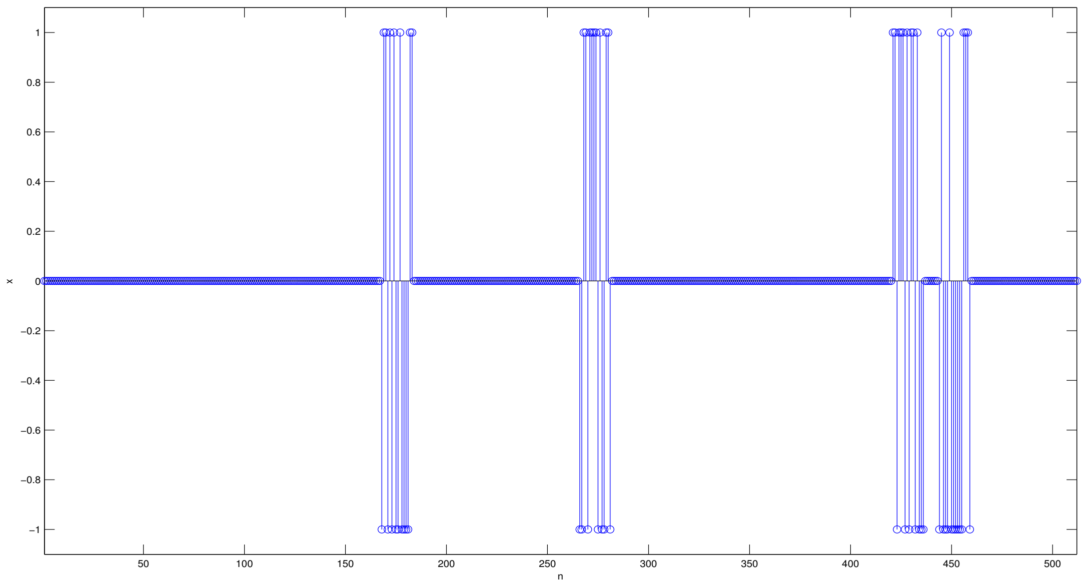

Figure 1: Generated data set x

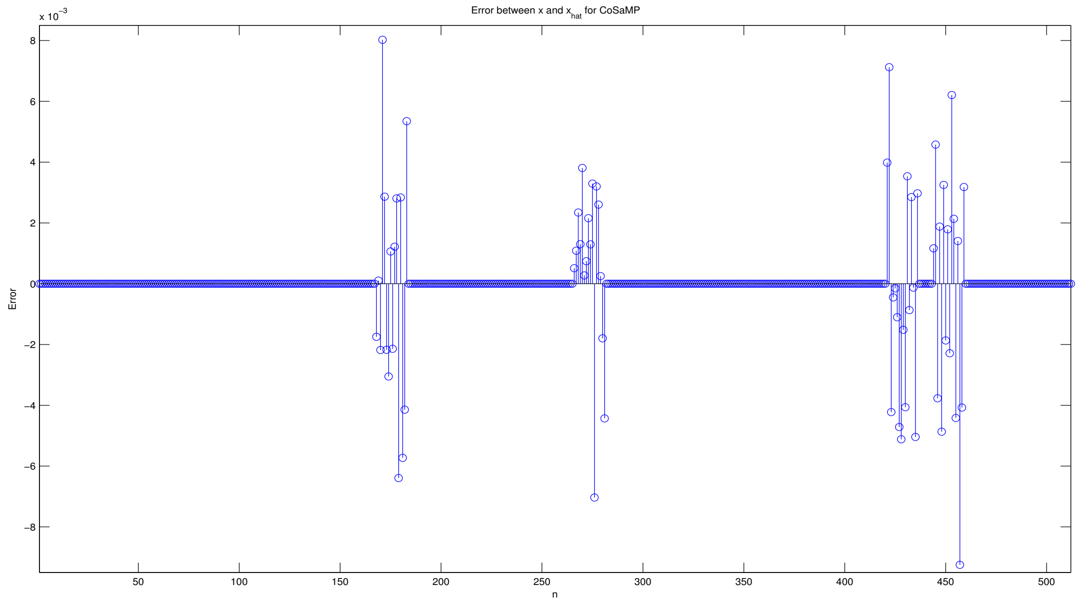

Figure 2: Error between x and x_{hat} with CoSaMP

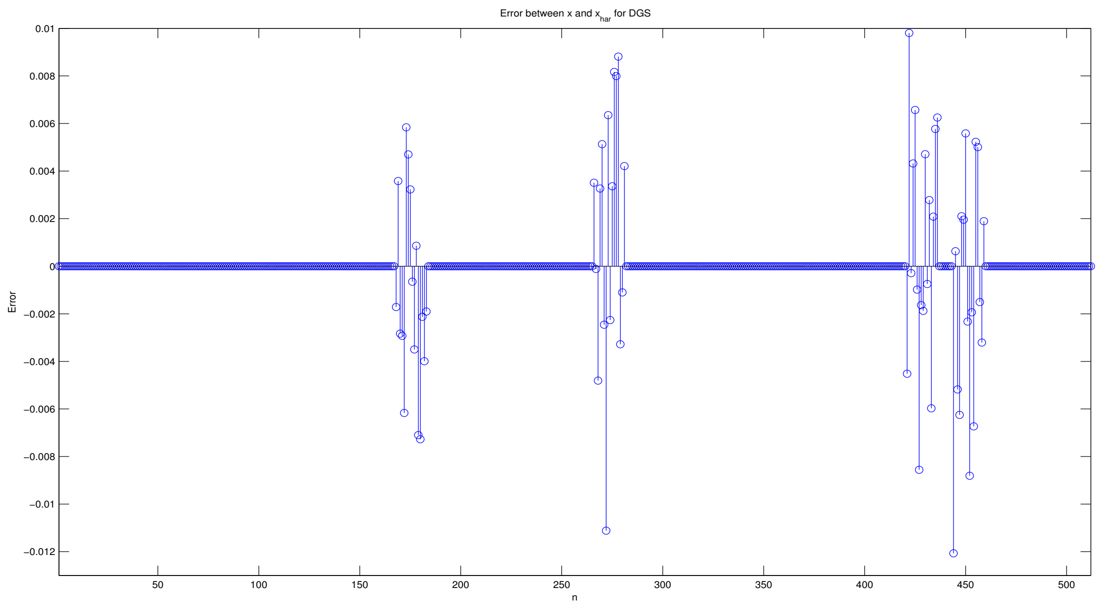

Figure 3: Error between x and x_{hat} with DGS

If CoSaMP is used with $m < (factor + 1)k$, then the algorithm doesn't converge.

5 Conclusion

In the coming years, IoT is going to dominate all spheres of our life and so amount of data generated will increase even further and with that there will be a dire need for algorithms which require fewer and fewer measurements and still provide signal reconstruction with small error.

CoSaMP has been around for quite a long time and works equally well for all kind of data sets irrespective of clustering or not. But for data with group clustering, DGS gives similar performance as CoSaMP but with fewer number of measurements.

DGS can be further improved to use lesser measurements and still provide proper reconstruction by using dynamic weights w, which can be obtained by solving a least squares problem [8].

Overall, more algorithms which utilize the inherent properties of the data, like group clustering in this case, need to be developed.

References

[1] A. Whitmore, A. Agarwal, and L. D. Xu. The internet of things-a survey of topics and trends. *Information Systems Frontiers*, 17(2):261–274, April 2015.

[2] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami. Internet of things (iot): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, 29:1645–1660, February 2013.

[3] S. Foucart and H. Rauhut. *A Mathematical Introduction to Compressive Sensing*, chapter 1. Applied and Numerical Harmonic Analysis. Springer, New York, 2013.

[4] R. G. Baraniuk. Compressive sensing. *IEEE Signal Processing Magazine*, 24(4):118–121, July 2007.

[5] D. Needell and J. A. Tropp. Cosamp: Iterative signal recovery from incomplete and inaccurate samples. *Applied and Computational Harmonic Analysis*, 26(3):301–321, May 2009.

[6] W. Dai and O. Milenkovic. Subspace pursuit for compressive sensing signal reconstruction. *IEEE Transactions on Information Theory*, 55(5):2230–2249, May 2009.

[7] J. Huang, D. Huang, and D. Metaxas. Learning with dynamic group sparsity. In *IEEE 12th International Conference on Computer Vision*, pages 64–71, Kyoto, 29 September–2 October 2009.

[8] S. Li, L. D. Xu, and X. Wang. Compressed sensing signal and data acquisition in wireless sensor networks and internet of things. *IEEE Transactions on Industrial Informatics*, 9(4):2177–2186, November 2013.