# Distributed Approximate Message Passing for Sparse Signal Recovery

Puxiao Han, Ruixin Niu, Mengqi Ren
Dept. of Electrical and Computer Engineering
Virginia Commonwealth University
Richmond, VA, 23284, U.S.A.
Email: {hanp, rniu, renm}@vcu.edu

Yonina C. Eldar
Dept. of Electrical Engineering
Technion-Israel Institute of Technology
Haifa, 32000, Israel
Email: yonina@ee.technion.ac.il

***Abstract*—In this paper, an efficient distributed approximate message passing (AMP) algorithm, named distributed AMP (DAMP), is developed for compressed sensing (CS) signal recovery in sensor networks with the sparsity $K$ unknown. In the proposed DAMP, distributed sensors do not have to use or know the entire global sensing matrix, and the burden of computation and storage for each sensor is reduced. To reduce communications among the sensors, a new data query algorithm, called global computation for AMP (GCAMP), is proposed. The proposed GCAMP based DAMP approach has exactly the same recovery solution as the centralized AMP algorithm. The performance of the DAMP approach is evaluated in terms of the communication cost saved by using the GCAMP. For the purpose of comparison, thresholding algorithm (TA), a well known distributed Top-K algorithm, is modified so that it also leads to the same recovery solution as the centralized AMP. Numerical results demonstrate that the GCAMP based DAMP outperforms the Modified TA based DAMP, and reduces the communication cost significantly.**

***Index Terms*—Compressed sensing, distributed AMP, sensor networks.**

## I. Introduction

Compressed sensing (CS) has wide applications in various areas of signal processing [1]. Due to the curse of dimensionality, it can be highly demanding to perform CS on a single processor. Further, distributed processing has the potential to reduce communications among distributed sensors. Hence, distributed CS (DCS) in sensor networks has become an interesting topic. A general DCS system contains two parts: (1) the local computation performed at each sensor, and (2) the global computation to obtain the estimate of the original sparse signal after sensors exchange the results of local computation.

Several distributed approaches based on various CS recovery algorithms were proposed. In [2], a distributed subspace pursuit (DiSP) algorithm was developed to recover joint sparse signals. In DiSP, each sensor needs to store the global sensing matrix, and local computation at each sensor involves optimization and matrix inversion. The computation and memory burden may become very challenging for the sensors in large-scale problems. Further, in DiSP the sparsity $K$ is assumed to be known, which may not be the case in many applications. In [3], an algorithm named D-ADMM based on basis pursuit (BP) was proposed, in which the sensors do not have to store the entire global sensing matrix. However, each sensor still needs to solve an optimization problem to get a recovery per iteration, and broadcasts it to its neighbors, which may induce high communication cost since the recovery in the first few iterations is not sparse.

Focusing on these problems, a DCS algorithm based on iterative hard thresholding (IHT) named D-IHT was proposed in [4], [5]. In the local computation of the D-IHT framework, each sensor just performs very simple operations such as matrix transpose, addition and multiplication. In the global computation, thresholding algorithm (TA) [6] has been applied, which is a popular method to solve the distributed Top-K problem in the field of database querying, to reduce the amount of messages sent between sensors. Nevertheless, in the D-IHT, the sparsity $K$ was also assumed to be known.

In this paper, we propose a distributed algorithm based on approximate message passing (AMP) [7], which does not require the knowledge of the signal's sparsity level, and has a linear convergence rate [7], [8]. For the proposed distributed AMP (DAMP) approach, we do not assume any prior knowledge of the global sensing matrix, and the distributed sensors do not have to store the entire global sensing matrix. In the local computation, each sensor only performs simple matrix operations. In the global computation per iteration, we propose a new algorithm, Global Computation for AMP (GCAMP), to reduce the amount of data transmitted in the sensor network. To the best of our knowledge, the proposed approach is the first distributed AMP algorithm.

We use $v(k)$ to denote the $k$-th component of the vector $v$, $[\cdot]^T$ to denote the transpose a matrix or vector, and $\|\cdot\|_0$ to denote the number of non-zero components of a vector.

## II. DAMP System

### A. The Centralized AMP

A task of CS is to recover a $K$-sparse signal $s_0 \in \mathbb{R}^N$ from its measurement $y = As_0 + n$, where $A \in \mathbb{R}^{M\times N}$ is the sensing matrix and $n$ is additive noise. We consider solving the problem:

$$\min_x \frac{1}{2}||y - Ax||_2^2 + \lambda||x||_1 \quad (1)$$

where $\lambda > 0$ is a regularization parameter. AMP is a good solution to (1) [7] without knowing $K$ and $\lambda$. Starting from

978-1-4799-7088-9/14/$31.00 ©2014 IEEE

$x_0 = 0$ and $z_0 = y$, it recursively obtains a new estimate $x_{t+1}$ of $s_0$ as follows:

$$x_{t+1} = \eta_t(x_t + A^T z_t; \tau\sigma_t) \tag{2}$$

$$z_{t+1} = y - Ax_{t+1} + \frac{||x_{t+1}||_0}{M} z_t \tag{3}$$

where $\sigma_t^2 = \frac{||z_t||_2^2}{M}$ [9], $\tau$ is a tunable parameter, and $\eta_t(x;\beta)$ for a vector $x$ returns another vector $u$ of the same dimensionality such that (s.t.)

$$u(k) = \text{sgn}(x(k)) \max(|x(k)| - \beta, 0), \ \forall k. \tag{4}$$

Note that if $x$ is a scalar, then it can be viewed as a one-dimensional vector and the definition of $\eta_t(x;\beta)$ still holds. The optimal value of $\tau$ depends on $\kappa = \frac{M}{N}$ and $\rho = \frac{K}{M}$ [9]. Since $K$ is unknown, a tuning procedure is needed to find a value for $\tau$ which is very close to the optimum.

### *B. The Distributed Framework of AMP*

Let us consider a sensor network with $P$ distributed sensors. Each sensor $p$ ($p = 1, \cdots, P$) takes a measurement of $s_0$ as $y^p = A^p s_0 + n^p$ and $A = \left[(A^1)^T, \cdots, (A^P)^T\right]^T$. Then, for each $p$, (2) and (3) can be re-written as:

$$x_{t+1} = \eta_t\left(x_t + \Sigma_{p=1}^P (A^p)^T z_t^p; \tau\sigma_t\right) \tag{5}$$

$$z_{t+1}^p = y^p - A^p x_{t+1} + \frac{||x_{t+1}||_0}{M} z_t^p \tag{6}$$

Let us introduce an intermediate matrix $W_t = \left[w_t^1, \ldots, w_t^P\right]$ with each column computed by the corresponding sensor as:

$$w_t^p = \begin{cases} x_t + (A^p)^T z_t^p, & p = 1 \\ (A^p)^T z_t^p, & \text{otherwise (o.w.)} \end{cases} \tag{7}$$

This matrix is similar to that in [4]. We can then write (5) as

$$x_{t+1} = \eta_t\left(\Sigma_{p=1}^P w_t^p; \tau\sigma_t\right) \tag{8}$$

DAMP can be divided into two parts: local computation of $z_t^p$ and $w_t^p (p = 1, \cdots, P)$, and global computation of $x_{t+1}$ and $\sigma_{t+1}$, in which transmission of data between sensors is required. For the latter, a natural approach is to send all the data in $w_t^p (p = 2, \cdots, P)$ to sensor 1, which induces a high communication cost when $N$ is large. We next show how to reduce the communication cost, while maintaining the same recovery solution as the centralized AMP.

### *C. GCAMP Algorithm*

According to (8), $x_{t+1}(n) = 0$ if $|\Sigma_{p=1}^P w_t^p(n)| \le \beta = \tau\sigma_t$. Therefore, we only need to know all the $n$'s and the corresponding $w_t^p(n)$'s s.t. $|\Sigma_{p=1}^P w_t^p(n)| > \beta$ in the global computation. This is similar to the Top-K problem in distributed database querying, which is to find the $K$ largest components of $\Sigma_{p=1}^P w_t^p$. There are two efficient approaches solving the Top-K problem: TA [6], which has been modified and applied in the global computation in D-IHT, and three-phase uniform threshold (TPUT) algorithm [10]. Our problem is different from the Top-K problem since we do not know how many components of $\Sigma_{p=1}^P w_t^p$ have magnitudes larger than $\beta$. Hence, we cannot directly apply TA or TPUT. Nevertheless, they do provide some insight on how to design the communication algorithm for DAMP. The key idea of the proposed GCAMP algorithm, as shown in Table I, is trying to get an upper bound for $|\Sigma_{p=1}^P w_t^p(n)|$. This is easier to accomplish by using the idea of TPUT than TA. Therefore, the GCAMP is more related to TPUT. To make a comparison, we also modify TA so that it can be used for global computation in DAMP in Section II-E, and show that GCAMP saves much more communication cost than the Modified TA in Section III-C.

TABLE I
GCAMP ALGORITHM

**Input** $w_t^1, \cdots, w_t^P$, $\beta = \tau\sigma_t$;

---

**Step I** Set $T = \beta\theta/(P-1)$, where $\theta \in (0,1)$ is a tuned parameter;
for sensor $p$ = 2:$P$
  denote $R_p = \{n : |w_t^p(n)| > T\}$;
  send all $(n, w_t^p(n))$ pairs for $n \in R_p$ to sensor 1;
endfor
**Step II** for sensor 1, define $I_S(x) := 1$ if $x \in S$ and $I_S(x) := 0$ o.w.;
for $n$ = 1:$N$
  get $S_n := \{p = 2, \cdots, P : I_{R_p}(n) = 1\}$ with cardinality $m_n$;
  Compute $U(n) = |w_t^1(n) + \Sigma_{p \in S_n} w_t^p(n)| + (P - 1 - m_n)T$;
  if $U(n) > \beta$ and $m_n < P - 1$
    broadcast the index $n$ to other sensors;
  endif
endfor
**Step III** denote $F = \{n : U(n) > \beta, m_n < P - 1\}$;
for sensor $p$ = 2:$P$
  send all $(n, w_t^p(n))$ pairs for $n \in F \backslash R_p$ to sensor 1;
endfor
**Step IV** for sensor 1, initialize $x_{t+1} = 0$;
for $n \in V := \{n : U(n) > \beta\}$
  Update $x_{t+1}(n) = \eta_t\left(\Sigma_{p=1}^P w_t^p(n); \beta\right)$ by (4);
endfor

---

**Output** $x_{t+1}$

It is easy to show that $U(n)$ is an upper bound of $\left|\Sigma_{p=1}^P w_t^p(n)\right|$ for all $n$, and $x_{t+1}$ which the GCAMP algorithm obtains is exactly the same as that obtained by the original centralized AMP algorithm. Interested readers are referred to [11] for the proof.

In Fig. 1, an example is provided to illustrate how GCAMP works, in which each sensor $p$ already sorts $w_t^p(n)$ in descending order of magnitudes, and stores the data in the form of $(n,\ w_t^p(n))$ pairs ($p = 1, \cdots, 3, n = 1, \cdots, 10$). Suppose $\beta = 20$ and $\theta = 0.8$, since we have $P = 3$ sensors, we get $T = \beta\theta/(P-1) = 8$. In step I, sensors 2 to $P$ send all $(n,\ w_t^p(n))$ pairs with $|w_t^p(n)| > T$ to sensor 1. In step II, sensor 1 receives the data, computes upper bounds $U(n)$ for $n = 1, \cdots, 10$ and obtains $F = V = \{4, 6, 7\}$. Then sensor 1 broadcasts indices in $n \in F$. In step III, sensor 2 sends $w_t^2(4)$ and $w_t^2(7)$, and sensor 3 sends $w_t^3(4)$ and $w_t^3(6)$ to sensor 1. Finally, in step IV, sensor 1 computes $x_{t+1}(n)$ for $n \in V$ by (7), and outputs the non-zero components of $x_{t+1}$. Overall, in this example, only 9 data points are sent from other sensors to sensor 1, and the total number of messages is 12 (9 data points plus 3 broadcast requests).

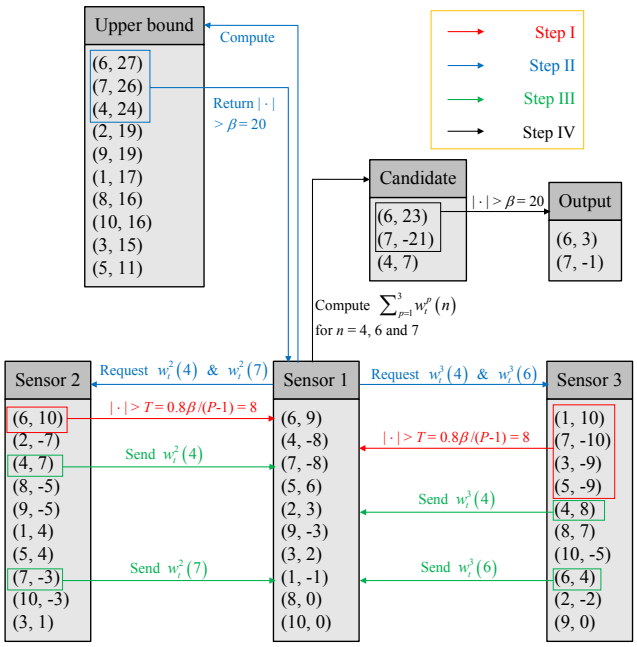


Fig. 1. An example of GCAMP algorithm

### D. Tuning of $\tau$ Values

With the GCAMP algorithm, DAMP can be developed. We adopt the tuning framework in [12] to find the optimal value for $\tau$. First, a descending list of candidate values of $\tau$, $\{\tau\}_{\ell=1}^{L} := [\tau_{\max},\ \tau_{\max} - \Delta\tau, \cdots, \tau_{\max} - (L-1)\Delta\tau]$ is generated. Then, for each candidate $\tau_\ell$, we run iterations in (5) and (6) until $x_t$ and $\sigma_t$ converge to $x_\ell^*$ and $\sigma_\ell^*$, and use them as the initial estimates for the iterations using the next candidate $\tau_{\ell+1}$. We repeat this process until $\sigma_\ell^*$ is not decreasing, and get the optimal $\tau$ value as well as the final estimate of $s_0$. For choosing the maximum candidate value, i.e., $\tau_{\max}$, we propose a different approach from [12] which may save more computational cost. Denote $\tilde{x}_t := x_t + A^T z_t = \Sigma_{p=1}^{P} w_t^p$. According to [13], as $N \to \infty$, asymptotically each component of $\tilde{x}_t - s_0$ is independent and identically distributed (i.i.d.) random variable, following a $\mathcal{N}(0, \sigma_t^2)$ distribution. Define $\gamma$ s.t. $\frac{1}{\sqrt{2\pi}} \int_{\gamma}^{+\infty} \exp(-\frac{t^2}{2}) dt = \frac{\alpha}{2}$, where $\alpha$ is a small number. If $|\tilde{x}_t(n)| > \gamma\sigma_t$, then we can reject the hypothesis $s_0(n) = 0$ at $1-\alpha$ level of significance. Therefore, we can let $\tau_{\max} = \gamma$. For example, we can set $\alpha = 0.0027$ and $\tau_{\max} = \gamma = 3$.

Note that in every iteration involving (5) and (6), after GCAMP returns $x_{t+1}$, sensor 1 broadcasts non-zero components of $x_{t+1}$ as well as their indices. In DAMP, we tune the optimal $\tau$ value in a descending order, which implies a larger threshold $\beta = \tau\sigma_t$ in the beginning. Therefore, different from [3], we have a sparse estimate $x_{t+1}$ even at the first few iterations. Hence, the communication cost for broadcasting $x_{t+1}$ is negligible compared with that of GCAMP. Once knowing $x_{t+1}$, each local sensor can obtain $z_{t+1}^p$ using (6) and $\sigma_{t+1}^p = ||z_{t+1}^p||_2$ $(p = 1, \cdots, P)$. Next, each sensor $p \geq 2$ just sends a scalar $\sigma_{t+1}^p$ to sensor 1, which needs $P-1$ messages. Then, sensor 1 computes $\sigma_{t+1} = \sqrt{\Sigma_{p=1}^{P} (\sigma_{t+1}^p)^2 / M}$, updates $\beta$ and $T$, and broadcasts the scalar $T$ to other sensors. Overall, GCAMP incurs most of the communication cost in DAMP. We will verify this in Section III-C.

### E. Comparison of GCAMP and Modified TA

TA [6] is another popular algorithm for solving Top-K problems. As discussed in Section II-C, the original TA can only be applied when $K$ is known. Therefore, we propose a modified TA algorithm as in Table II, and let it be a control algorithm for GCAMP.

TABLE II
MODIFIED TA ALGORITHM

**Input** $w_t^1, \cdots, w_t^P$, $\beta = \tau\sigma_t$;

---

**Initialization** $x_{t+1} = 0$, `count`$= 0$;
for sensor $p$ = 1:$P$
  sort components of $w_t^p$ in descending order of magnitudes;
  define the sorted vector as $s_t^p$ and $I_t^p(n) := \ell$ s.t. $w_t^p(\ell) = s_t^p(n)$;
  mark all $(I_t^p(n), s_t^p(n))$ pairs as "unsent";
endfor
while `TRUE`
  for $p$ = 1:$P$
    find the first $(I_t^p(n), s_t^p(n))$ pair marked "unsent" from top;
    set $u_p = s_t^p(n)$, broadcast $(I_t^p(n), u_p)$ to other sensors;
    mark $(I_t^p(n), s_t^p(n))$ as "sent";
    for sensor $q \neq p$
      store $u_p$ and send $(I_t^p(n), w_t^q(I_t^p(n)))$ to sensor $p$;
      mark $(I_t^p(n), w_t^q(I_t^p(n)))$ as "sent";
    endfor
    update $x_{t+1}(I_t^p(n)) = \eta_t(\Sigma_{p=1}^{P} w_t^p(I_t^p(n)); \beta)$;
    `count`=`count`+1;
    if `count`$\geq P$ and $\Sigma_{p=1}^{P}|u_p| \leq \beta$, or if `count`$\geq N$
      set $N_s =$ `count`, the algorithm terminates;
    endif
  endfor
endwhile

---

**Output** $x_{t+1}$

---

It is easy to show that in each iteration, the Modified TA algorithm also gives exactly the same $x_{t+1}$ as that of the original AMP algorithm. The proof is available in [11].

*Number of Messages:* For a set, denote $|\cdot|$ as its cardinality. For GCAMP, the total number of messages is $\Sigma_{p=2}^{P}|R_p| + |F| + \Sigma_{p=2}^{P}|F \backslash R_p|$. For Modified TA, in each for-loop iteration inside the while-loop, there are 1 broadcasting message from some sensor to others and $P-1$ incoming messages. Clearly, the number of for-loop iterations inside the while-loop is $N_s$ in Table II, so the total number of messages is $PN_s$. It is easy to check that, for the data set in Fig. 1, Modified TA needs $PN_s = 3 \times 9 = 27$ messages, more than twice that required by the GCAMP.

## III. NUMERICAL RESULTS

### A. Performance Measures

Since we have proved that the DAMP algorithm has exactly the same solution as the centralized AMP, and the recovery accuracy and convergence of AMP has been well studied in the literature, it is not necessary to evaluate them again in the paper. Instead, as DAMP is a distributed algorithm, it is important to evaluate the communication cost saved by using GCAMP. So we use the number of messages transmitted as the

performance measure, which is widely used in literature [6], [10]. We compare the number of messages used in GCAMP to that in Modified TA. Considering the approach of sending all the data to sensor 1, which has a total number of messages $N(P-1)$, we define the normalized message number as

$$\mu_M = \frac{\text{number of messages in computing } x_{t+1}}{N(P-1)} \quad (9)$$

This leads to $\mu_M = \frac{\Sigma_{p=1}^{P}|R_p|+|F|+\Sigma_{p=1}^{P}|F\backslash R_p|}{N(P-1)}$ for GCAMP and $\mu_M = \frac{N_s P}{N(P-1)}$ for Modified TA.

### B. Simulation Setup

Our focus is not to investigate large-scale problems, but to develop distributed algorithms and evaluate their efficiency in reducing communication costs. Nevertheless, we still use a considerably large $N = 5000$, and choose $\kappa$ from $[0.1,\ 0.5]$, $\rho$ from $[0.1,\ 0.3]$, which leads to $M = N\kappa$ in $[500,\ 2500]$ and $K = M\rho$ in [50, 750]. The number of sensors $P$ is within $[5,\ 50]$. The sensing matrix $A$ with i.i.d. entries $\sim \mathcal{N}(0, \frac{1}{M})$ is partitioned into $P$ parts with each sensor having a $(M/P) \times N$ submatrix. Each component of $s_0$ is i.i.d. drawn from $f_X(x) = \kappa\rho G(x) + (1-\kappa\rho)\delta(x)$ , where $G(x)$ is the probability density function (pdf) of the standard Gaussian distribution and $\delta(x)$ is the Dirac delta function. The measurements of $s_0$ are corrupted by an additive noise $n \sim \mathcal{N}(0, \sigma^2 I_M)$ and $\sigma$ is the standard deviation with a value in [0.01, 0.1]. The parameter $\theta$ in GCAMP is set to 0.8. Regarding the tuning procedure for optimal $\tau$ values, we make a candidate list for $\tau$ of length 11, starting from 3 with a step size $-0.2$; for each candidate, the convergence criterion is $|\sigma_t - \sigma_{t-1}| < 0.01\sigma_{t-1}$. $\bar{\mu}_M$ is defined as $\mu_M$ averaged over iterations based on 100 Monte-Carlo runs.

### C. Performance Evaluation

*1) Comparison between GCAMP and Modified TA:* We evaluate $\bar{\mu}_M$ in different settings with various combinations of $\sigma$, $P$, $\rho$, and $\kappa$, and the numerical results are provided in Tables III, IV and V. In the tables, the former entry in each pair inside the parentheses denotes $\bar{\mu}_M$ for GCAMP, and the latter denotes that for Modified TA. It is clear that in each case, GCAMP outperforms Modified TA significantly. Modified TA always uses more than $N(P-1)$ messages except for the case $P = 5$, while the GCAMP can save from 22.7% to 48.2% of the messages. Fig. 2 gives the cumulative distributions of $\mu_M$ in each iteration for GCAMP and Modified TA in four different scenarios. It is clear that in each scenario, Modified TA uses more than $N(P-1)$ messages in at least 33.4 % of the total iterations; while GCAMP never uses more than $0.91N(P-1)$ messages in any iteration, and among more than 95% of the total iterations, it just uses $[40\%, 80\%] \times N(P-1)$ messages, that is, it can save $20\% \sim 60\%$ of the messages with probability at least 95%.

*2) Communication cost for computing $z_{t+1}^p$ and $\sigma_{t+1}$:* We investigate the ratio between the number of messages used for obtaining $z_{t+1}^p$ and $\sigma_{t+1}$ after the GCAMP returns $x_{t+1}$, and that used for GCAMP computing $x_{t+1}$ over all parameter settings above, and the largest value we observe is 7.25 %, which means that GCAMP incurs most of the communication cost in DAMP.

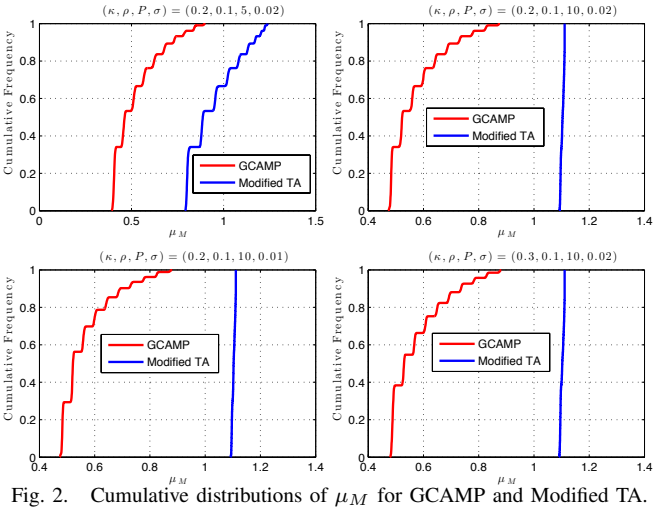


Fig. 2. Cumulative distributions of $\mu_M$ for GCAMP and Modified TA.

TABLE III
$\bar{\mu}_M$ FOR GCAMP AND MODIFIED TA ($\sigma = 0.02$, $P = 10$)

| | $\kappa = 0.1$ | 0.2 | 0.3 | 0.4 | 0.5 |
|---|---|---|---|---|---|
| $\rho$=0.10 | (0.547, 1.101) | (0.567, 1.103) | (0.573, 1.103) | (0.587, 1.103) | (0.589, 1.103) |
| 0.20 | (0.659, 1.108) | (0.667, 1.108) | (0.672, 1.108) | (0.691, 1.109) | (0.684, 1.108) |
| 0.30 | (0.632, 1.108) | (0.690, 1.109) | (0.737, 1.109) | (0.751, 1.110) | (0.755, 1.110) |

TABLE IV
$\bar{\mu}_M$ FOR GCAMP AND MODIFIED TA ($\kappa = 0.2$, $\rho = 0.1$, $P = 10$)

| $\sigma = 0.01$ | 0.03 | 0.05 | 0.07 | 0.09 |
|---|---|---|---|---|
| (0.564, 1.103) | (0.574, 1.104) | (0.582, 1.104) | (0.589, 1.104) | (0.592, 1.105) |

TABLE V
$\bar{\mu}_M$ FOR GCAMP AND MODIFIED TA ($\kappa = 0.2$, $\rho = 0.1$, $\sigma = 0.02$)

| $P = 5$ | 10 | 15 | 20 | 25 |
|---|---|---|---|---|
| (0.518, 0.941) | (0.567, 1.103) | (0.623, 1.071) | (0.664, 1.053) | (0.694, 1.042) |
| $P = 30$ | 35 | 40 | 45 | 50 |
| (0.717, 1.034) | (0.735, 1.029) | (0.751, 1.026) | (0.763, 1.023) | (0.773, 1.020) |

## IV. Conclusion

Without assuming the knowledge of the signal's sparsity level, the DAMP approach has been developed for performing distributed compressed sensing in sensor networks, consisting of a series of local and global computations. We proposed the GCAMP in the stage of global computation to reduce the number of messages per iteration, and the DAMP based on GCAMP has exactly the same solution as the centralized AMP. For comparison purposes, we modified the TA algorithm so that it can be used in DAMP, also with exactly the same solution as that of the centralized AMP. Numerical results demonstrated that GCAMP based DAMP outperforms Modified TA based DAMP significantly, and is very efficient in reducing communication costs.

## REFERENCES


[1] M. F. Duarte and Y. C. Eldar, "Structured compressed sensing: From theory to applications," *IEEE Trans. Sig. Proc.*, vol. 59, pp. 4053–4085, September 2011.

[2] D. Sundman, S. Chatterjee, and M. Skoglund, "A greedy pursuit algorithm for distributed compressed sensing," in *Proc. IEEE Int. Conf. on Acoust., Speech, and Sig. Proc. (ICASSP)*, 2012, pp. 2729–2732.

[3] J. Mota, J. Xavier, P. Aguiar, and M. Puschel, "Distributed basis pursuit," *IEEE Trans. Sig. Proc.*, vol. 60, pp. 1942–1956, April 2012.

[4] S. Patterson, Y. C. Eldar, and I. Keidar, "Distributed sparse signal recovery for sensor networks," in *Proc. IEEE Int. Conf. on Acoust., Speech, and Sig. Proc. (ICASSP)*, 2013, pp. 4494–4498.

[5] ——, "Distributed compressed sensing for static and time-varying networks," *IEEE Trans. Sig. Proc.*, vol. 62, no. 19, pp. 4931–4946, Oct 2014.

[6] R. Fagin, A. Lotem, and M. Naor, "Optimal aggregation algorithms for middleware," in *Symposium on Principles of Database Systems*, 2001, pp. 614–656.

[7] D. L. Donoho, A. Maleki, and A. Montanari, "Message passing algorithms for compressed sensing," in *Proc. Natl. Acad. Sci.*, vol. 106, Madrid, Spain, September 2009, pp. 18 914–18 919.

[8] A. Maleki and R. G. Baraniuk, "Least favorable compressed sensing problems for the first order methods," in *Proc. IEEE Int. Symp. Inf. Theory*, 2011, pp. 134–138.

[9] D. L. Donoho, A. Maleki, and A. Montanari, "The Noise-Sensitivity Phase Transition in Compressed Sensing," *IEEE Trans. Info. Theory*, vol. 57, pp. 6920–6941, October 2011.

[10] P. Cao and Z. Wang, "Efficient top-k query calculation in distributed networks," in *Intl. Symposium on Principles Of Distributed Computing (PODC)*, 2004, pp. 206–215.

[11] P. Han, R. Niu, and M. Ren, "Distributed approximate message passing for compressed sensing," *arXiv preprint arXiv:1404.3766*, 2014.

[12] L. Anitori, A. Maleki, M. Otten, R. G. Baraniuk, and P. Hoogeboom, "Design and Analysis of Compressed Sensing Radar Detectors," *IEEE Trans. Signal Proc.*, vol. 61, pp. 813–827, February 2013.

[13] M. Bayati and A. Montanari, "The Dynamics of Message Passing on Dense Graphs, with Applications to Compressed Sensing," *IEEE Trans. Info. Theory*, vol. 57, pp. 764–785, February 2011.