entropy

MDPI

Article

Sparse Estimation Based on a New Random Regularized Matching Pursuit Generalized Approximate Message Passing Algorithm

Yongjie Luo [1], Guan Gui [2,*], Xunchao Cong [1] and Qun Wan [1]

[1] Department of Electronic Engineering, University of Electronic Science and Technology of China, No. 2006, Xiyuan Ave., West Hi-Tech Zone, Chengdu 611731, China; 201311020326@std.uestc.edu.cn (X.C.); wanqun@uestc.edu.cn (Q.W.)

[2] College of Telecommunication and Information Engineering, Nanjing University of Posts and Telecommunications, 66 Xinmofan Road, Nanjing 210003, China; guiguan@njupt.edu.cn

* Correspondence: guiguan@njupt.edu.cn

Academic Editors: Badong Chen and Jose C. Principe
Received: 14 February 2016; Accepted: 19 May 2016; Published: 28 May 2016

Abstract: Approximate Message Passing (AMP) and Generalized AMP (GAMP) algorithms usually suffer from serious convergence issues when the elements of the sensing matrix do not exactly match the zero-mean Gaussian assumption. To stabilize AMP/GAMP in these contexts, we have proposed a new sparse reconstruction algorithm, termed the Random regularized Matching pursuit GAMP (RrMpGAMP). It utilizes a random splitting support operation and some dropout/replacement support operations to make the matching pursuit steps regularized and uses a new GAMP-like algorithm to estimate the non-zero elements in a sparse vector. Moreover, our proposed algorithm can save much memory, be equipped with a comparable computational complexity as GAMP and support parallel computing in some steps. We have analyzed the convergence of this GAMP-like algorithm by the replica method and provided the convergence conditions of it. The analysis also gives an explanation about the broader variance range of the elements of the sensing matrix for this GAMP-like algorithm. Experiments using simulation data and real-world synthetic aperture radar tomography (TomoSAR) data show that our method provides the expected performance for scenarios where AMP/GAMP diverges.

Keywords: compressed sensing; random regularization; matching pursuit; generalized approximate message passing; replica method

1. Introduction

Compressed Sensing (CS) has been a research focus in recent years. The idea of sparse estimation and compressed sensing has also been applied in adaptive filtering [1–4] and nonlinear dynamics systems [5,6]. Define the support of an unknown K-sparse ($K \ll N$) vector $\mathbf{x} \in \mathbb{C}^N$ as: $S = \{j : j \in \{1, \ldots, N\}, x_j \neq 0\}$. We consider an under-determined noisy linear system:

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \boldsymbol{\epsilon} \tag{1}$$

where $\mathbf{y} \in \mathbb{C}^M$ is the measurement vector, $\boldsymbol{\epsilon} \in \mathbb{C}^M$ is the additive Gaussian noise vector and $\mathbf{A} \in \mathbb{C}^{M \times N}$ ($M < N$) is the projection matrix, also termed the sensing matrix. The aim of CS is to reconstruct $\mathbf{x}$ from $\mathbf{y}$ and $\mathbf{A}$.

On the one hand, it is well known that the CS problem can be rephrased into a probabilistic inference problem on the bipart factor graph shown in Figure 1a. Donoho *et al.* proposed the Approximate Message Passing (AMP) algorithm [7,8] to solve it. Rangan *et al.* generalized AMP to

adapt to an arbitrary sparse prior and arbitrary measurement noise; their work is called the Generalized Approximate Message Passing (GAMP) algorithm [9]. Since the derivation of AMP/GAMP is based on a zero-mean Gaussian projection matrix hypothesis, they cannot work well if the projection matrix does not satisfy the hypothesis, as shown in Section 4. This phenomenon motivates us to improve the AMP/GAMP algorithm for a more general projection matrix. On the other hand, although ℓ_0 methods, such as Orthogonal Matching Pursuit (OMP) [10] and Compressive Sampling Matching Pursuit (CoSaMP) [11], are simple and effective, they use Least Squares (LS) to calculate the amplitude of non-zero elements. Therefore, these methods have to face the matrix inversion problem. When the scale of the projection matrix is small, this problem is not serious. However, when the scale becomes larger, the multipliers grow cubically as a function of sparsity K. This drawback stimulates us to find some approach to replace the LS step in ℓ_0 methods and to overcome the disadvantages of OMP and CoSaMP.

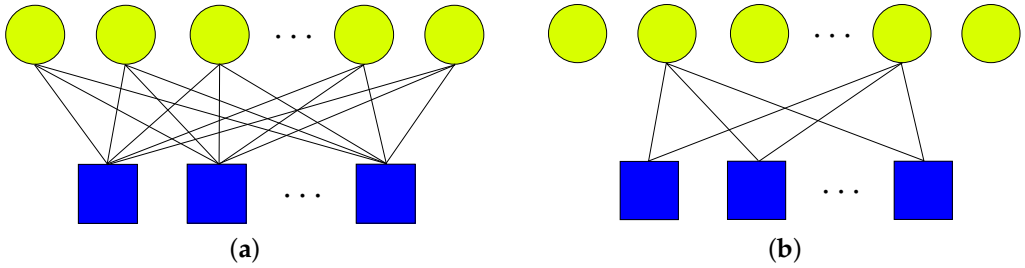

Figure 1. The factor graph description of a linear system. (**a**) Bipart factor graph; (**b**) Restricted factor graph.

In this paper, we propose a new sparse estimation algorithm, termed Random regularized Matching pursuit Generalized Approximate Message Passing (RrMpGAMP), to solve the CS problem. Our method can be regarded as a kind of ℓ_0 optimization algorithm, termed Random regularized Matching Pursuit (RrMP), but the mean and variance estimation of $\mathbf{x}$ are the output of a restricted GAMP algorithm, termed the Fixed support GAMP (FsGAMP). In this way, it allows a more general projection matrix and does not calculate the inverse matrix directly. The computational complexity of RrMpGAMP is comparable to GAMP, and the memory occupation of RrMpGAMP is much less than GAMP and other LS-based algorithms.

The rest of this paper is organized as follows. We describe the RrMpGAMP algorithm in Section 2, then analyze the convergence of FsGAMP by the replica method in Section 3 and then perform experiments in Section 4. We draw a conclusion in the last Section 5. The derivation of the FsGAMP algorithm is provided in the Appendix.

2. Random Regularized Matching Pursuit Generalized Approximate Message Passing

As mentioned in the Introduction, the RrMpGAMP algorithm includes two parts: the RrMP algorithm and the FsGAMP algorithm, which is embedded in RrMP. The former is a new ℓ_0 optimization algorithm; the latter is a tiny revised GAMP algorithm.

At the beginning, we assume the support S has been known, which means the probability of every zero element in $\mathbf{x}$ is one. Therefore, we just need to consider the probability of other non-zero elements in $\mathbf{x}$. We suppose that the Probability Distribution Function (PDF) of $\mathbf{x}$ and the likelihood function of $\mathbf{z} = \mathbf{A}\mathbf{x}$ are separable, *i.e.*,

$$p_{\boldsymbol{x}}(\mathbf{x}) = 1 \times p_{\mathbf{x}_S}(\mathbf{x}_S) = \prod_{j=1}^{K} p_{x_{S_j}}(x_{S_j}) \tag{2}$$

$$p_{\boldsymbol{y}|\boldsymbol{z}}(\mathbf{y}|\mathbf{z}) = \prod_{m=1}^{M} p_{\mathrm{y}_m|\mathrm{z}_m}(y_m|z_m) \tag{3}$$

Therefore, the probabilistic form of Equation (1) can be written as:

$$\begin{aligned} p_{\boldsymbol{x}|\boldsymbol{y}}(\mathbf{x}|\mathbf{y}) &= 1 \times p_{\boldsymbol{x}_S|\boldsymbol{y}}(\mathbf{x}_S|\mathbf{y}) \propto p_{\boldsymbol{y}|\boldsymbol{x}_S}(\mathbf{y}|\mathbf{x}_S) p_{\boldsymbol{x}_S}(\mathbf{x}_S) \\ &= \prod_{m=1}^{M} p_{\mathrm{y}_m|\mathbf{z}_m}\left(y_m | a_{mS_k} x_{S_k} + \sum_{j \neq k}^{K} a_{mS_j} x_{S_j}\right) \prod_{j=1}^{K} p_{\mathrm{x}_{S_j}}(x_{S_j}) \end{aligned} \tag{4}$$

Actually we do not know S in the first place; a straightforward idea is to find it by some matching pursuit-based method.

2.1. Random Regularized Matching Pursuit Algorithm

A traditional matching pursuit algorithm (MP or OMP) finds an index of the non-zero element of $\mathbf{x}$ at each iteration. It does not rollback even if an incorrect index has been chosen. Our proposed algorithm sequentially pursues multiple indexes at each iteration and has the ability to undo, like the well-known Regularized OMP (ROMP) algorithm and the CoSaMP algorithm.

We use the subscript l to denote *left*, r to denote *right* and $|U|$ to designate the cardinality of temporary support U. Define three functions appearing in the algorithm: $\texttt{FixSuppGAMP}(\mathbf{y}, \mathbf{A}, U)$ calculates the mean and variance estimation of $\mathbf{x}$; $R_{2s\backslash s}(\mathbf{v})$ finds the indexes of the $2s < K$ absolutely largest entries of $\mathbf{v}$, where s is the probe length, and then shuffles these indexes and bisects them; the outputs are the left and right indexes' subset and the amplitudes supported on them, respectively; support updating function $H(\mathbf{u}, \mathbf{c}, \Lambda, S)$ generates a new indexes' set by a group of rules; we will explain it soon.

The steps of RrMP are listed in Algorithm 1.

Algorithm 1 Random Regularized Matching Pursuit.

Input: $\mathbf{y}, \mathbf{A}, K, s$
Output: $\mathbf{x}^{n+1}$
1: $S^0 \leftarrow \emptyset, \mathbf{x}^0 \leftarrow \mathbf{0}$
2: $(\Lambda_l, \mathbf{c}_l, \Lambda_r, \mathbf{c}_r) \leftarrow R_{2s\backslash s}(\mathbf{A}^T(\mathbf{y} - \mathbf{A}\mathbf{x}^n))$
3: $U_l \leftarrow S^n \cup \Lambda_l, U_r \leftarrow S^n \cup \Lambda_r$
4: $\mathbf{u}_l \leftarrow \texttt{FixSuppGAMP}(\mathbf{y}, \mathbf{A}, U_l)$
5: $\mathbf{u}_r \leftarrow \texttt{FixSuppGAMP}(\mathbf{y}, \mathbf{A}, U_r)$
6: $\theta = \arg\min_{\{l,r\}} \{\|\mathbf{y} - \mathbf{A}\mathbf{u}_l\|_2, \|\mathbf{y} - \mathbf{A}\mathbf{u}_r\|_2\}$
7: $\Lambda^{n+1} \leftarrow \Lambda_\theta, \mathbf{c}^{n+1} \leftarrow \mathbf{c}_\theta, \mathbf{u}^{n+1} \leftarrow \mathbf{u}_\theta, U^{n+1} \leftarrow U_\theta$
8: $S^{n+1} \leftarrow H(\mathbf{u}^{n+1}, \mathbf{c}^{n+1}, \Lambda^{n+1}, S^n)$
9: $\mathbf{x}^{n+1} \leftarrow \texttt{FixSuppGAMP}(\mathbf{y}, \mathbf{A}, S^{n+1})$

They are separated into two stages: the first stage is Lines 2–5, and the second stage is Lines 6–9. At the first stage, Line 2 obtains two index candidates, Λ_l and Λ_r, and the correlation coefficients between the residual and columns of $\mathbf{A}$, $\mathbf{c}_l$ and $\mathbf{c}_r$. It is a key step, since the shuffle and bisecting operations are equivalent to a random search in N dimensional $\{0, 1\}$ space. Line 3 merges the previous support set S^n with two current candidates, respectively. Lines 4 and 5 roughly estimate $\mathbf{x}$ by the FsGAMP algorithm. At this stage, the candidate indexes are not replaced/dropped. Therefore, the number of possible incorrect indexes is typically greater than that at the second stage. This is the meaning of "roughly". At the second stage, Lines 6 and 7 choose the minimal residual branch as a new candidate. This prune operation can be regarded as a regularization since the algorithm chooses a small residual branch and discards a big residual branch. In another words, the algorithm discards half of the candidate indexes pursued by Line 2. Line 8 updates the support set. Line 9 estimates $\mathbf{x}$ again. Note that Lines 3–5 can be calculated by parallel computing.

We define the precision τ as the ratio of the residual norm over the measurement norm, and set the error bound to 0.01, which means 99.99% of the measurement energy has been recovered. When

$|P|$ equals sparsity K, if the precision does not reach the error bound, set $S^{n+1} = P$, keep running RrMP at most $2s$ times and then stop the algorithm.

The idea of the support updating function comes from two observations: (i) not all indexes in the candidate set are correct, so the algorithm should drop or replace a part of the indexes, as shown in Equations (7) and (8); the dropout/replacement operations can be regarded as a regularization; (ii) the candidate indexes corresponding to correlations $\mathbf{c}^{n+1}$ and to amplitudes $\mathbf{u}^{n+1}$ are not overlapping entirely; some correct indexes belong to the former, and some belong to the latter. It is necessary to merge them, as shown in Equation (9).

Now, we define the support updating function $H(\mathbf{u}^{n+1}, \mathbf{c}^{n+1}, \Lambda^{n+1}, S^n)$ as follows. Firstly, define two threshold functions corresponding to the sparse representation coefficients $\mathbf{u}$, which are evolved with iterations:

$$a \triangleq \min_{i \in S^n} \{|(\mathbf{u}^{n+1})|_i\} \tag{5}$$

$$b \triangleq \max_{j \in \Lambda^{n+1}} \{|(\mathbf{u}^{n+1})|_j\} \tag{6}$$

Secondly, define the indexes' updating procedure just for $\mathbf{u}$:

$$P \triangleq \begin{cases} S^n \cup \{j : \arg\max_{j \in \Lambda^{n+1}} \{|(\mathbf{u}^{n+1})_j|\}\}, & \frac{a}{2} > b \\ S^n \cup \{j : j \in \Lambda^{n+1}, |(\mathbf{u}^{n+1})_j| \geq \frac{a}{2}\}, & a > b \geq \frac{a}{2} \\ S^n \cup \{j : j \in \Lambda^{n+1}, |(\mathbf{u}^{n+1})_j| \geq \frac{b}{2}\}, & b > a \geq \frac{b}{2} \\ \left\{i : i \in S^n, |(\mathbf{u}^{n+1})_i| \geq \frac{b}{2}\right\} \cup \left\{j \in \Lambda^{n+1}, |(\mathbf{u}^{n+1})_j| \geq \frac{b}{2}\right\}, & \frac{b}{2} \geq a \end{cases} \tag{7}$$

Thirdly, define the the indexes' updating procedure just for the correlation coefficients $\mathbf{c}$:

$$Q \triangleq \{j : j \in \Lambda^{n+1}, |(\mathbf{c}^{n+1})_j| \geq 0.5\|\mathbf{c}^{n+1}\|_\infty\} \tag{8}$$

Lastly, we merge the two index sets to obtain the output of:

$$S^{n+1} = P \cup Q \tag{9}$$

These update rules are inspired by the regularized OMP algorithm [12,13]. It seems a bit complex, but the basic rationale is simple. We segment the absolute amplitudes on the pursued support at each iteration to ladders; the height of each ladder is dominated by the minimum of the absolute amplitudes, which is supported on the previous pursued support set S^n, and by the maximum of absolute amplitudes, which is supported on the current candidate support set Λ^{n+1}. Since the square of the amplitude is energy, the absolute amplitudes in the same ladder step can be explained to obey a similar energy level. For Equation (7),

- If $\frac{a}{2} > b$, this means that the absolute amplitudes of the current candidate indexes are much smaller than the absolute amplitudes of the previous pursued indexes, so the algorithm chooses only one current candidate index. This situation corresponds to a sharp ladder step.
- If $a > b \geq \frac{a}{2}$, this means the absolute amplitudes of the current candidate indexes are just a little smaller than the absolute amplitudes of the previous pursued indexes, so the algorithm chooses those indexes whose absolute amplitudes are greater than $a/2$. This situation corresponds to a flattened ladder step.
- If $b > a \geq \frac{b}{2}$, this means the absolute amplitudes of the current candidate indexes are greater than the absolute amplitudes of the previous pursued indexes, but the minimum absolute amplitude of the previous pursued indexes is not a very small value, so the algorithm chooses those indexes whose absolute amplitudes are greater than $b/2$. This situation also corresponds to a flattened ladder step.

- If $\frac{b}{2} > a$, that means the minimum absolute amplitude of the previous pursued indexes is very small, so the algorithm drops a part of the previous pursued indexes and adds a part of the current candidate indexes. This situation corresponds to a sharp ladder step.

However, it is not enough to choose indexes just by absolute amplitudes. Some correlations about the residual and the columns of $\mathbf{A}$ are significant. The algorithm finds these indexes by Line 2 in Algorithm 1. Maybe, the absolute amplitudes supporting them are not significant, such that they are excluded by Equation (7). We can imagine these indexes as free electrons that are escaping to other energy levels. Therefore, the algorithm should retrieve them. This is the meaning of Equation (8). The rules are heuristic, the explanations are analogies to physics.

2.2. Fixed Support GAMP Algorithm

The concept of message passing on a factor graph can be found in [14]. In contrast to AMP/GAMP, the FsGAMP algorithm does not reform Equation (1) to a bipart graph. It restricts the edges to a part of variable nodes, which correspond to the temporary support U, as shown in Figure 1b. Based on this factor graph and the posterior probability Equation (4), firstly, we define the messages from the factor node to the variable node. For the Max-Sum rule, they are:

$$\begin{aligned}\mathbf{\Delta}_{\mathsf{f}_m\to\mathsf{x}_{U_k}}(t,x_{U_k}) \triangleq \max_{\mathbf{x}_U}\Big[&\log p_{\mathsf{y}_m|\mathsf{z}_m}\big(y_m\big|a_{mU_k}x_{U_k}+\textstyle\sum_{j\neq k}^{|U|}a_{mU_j}x_{U_j}\big)\\&+\textstyle\sum_{j\neq k}^{|U|}\mathbf{\Delta}_{\mathsf{f}_m\leftarrow\mathsf{x}_{U_j}}(t,x_{U_j})\Big]+\mathsf{c}\end{aligned} \tag{10}$$

where c means constant ; for the Sum-Product rule, they are:

$$\begin{aligned}\mathbf{\Delta}_{\mathsf{f}_m\to\mathsf{x}_{U_k}}(t,x_{U_k}) \triangleq \log\Big[&\int_{\{x_{U_j}\}_{j\neq k}} p_{\mathsf{y}_m|\mathsf{z}_m}\big(y_m\big|a_{mU_k}x_{U_k}+\textstyle\sum_{j\neq k}^{|U|}a_{mU_j}x_{U_j}\big)\\&\times\textstyle\prod_{j\neq k}^{|U|}\exp\left(\mathbf{\Delta}_{\mathsf{f}_m\leftarrow\mathsf{x}_{U_j}}(t,x_{U_j})\right)\Big]+\mathsf{c}\end{aligned} \tag{11}$$

Secondly, we define the messages from the variable node to the factor node. The form of the Max-Sum rule is the same as the form of the Sum-Product rule:

$$\mathbf{\Delta}_{\mathsf{f}_m\leftarrow\mathsf{x}_{U_k}}(t+1,x_{U_k}) \triangleq \log p_{\mathsf{x}_{U_k}}(x_{U_k})+\textstyle\sum_{i\neq m}^{M}\mathbf{\Delta}_{\mathsf{f}_i\to\mathsf{x}_{U_k}}(t,x_{U_k})+\mathsf{c} \tag{12}$$

The derivation of FsGAMP is very similar with GAMP [9]. Due to space limitations, we only emphasize the difference in the main body and left the details to the Appendix. Since $|U|$ non-zero elements of $\mathbf{x}$ have been pursued, we set other elements of $\mathbf{x}$ to zero directly, such that the $\sum_{j\neq k}^{N}a_{mj}x_j$ and $\sum_{j\neq k}^{N}|a_{mj}|^2x_j$ terms in GAMP reduce to $\sum_{j\neq k}^{|U|}a_{mU_j}x_{U_j}$ and $\sum_{j\neq k}^{|U|}|a_{mU_j}|^2x_{U_j}$. Other derivation steps are the same as GAMP. It is worth noting that the derivation of GAMP is primarily based on the Central-Limit Theorem (CLT), but in FsGAMP, this condition is canceled, because we have known the index set of zero elements (though maybe incorrect), such that the mean and variance estimation of these elements are also zeros. This simplification eliminates uncertainty and brings stability to FsGAMP to adapt more types of projection matrices.

Rangan *et al.* [9,15] investigated the influence of damping in GAMP. They found that damping can induce convergence. Similar steps are used in the FsGAMP algorithm:

$$\nu^{\mathbf{p}}(t) = \delta\nu^{\mathbf{p}}(t)+(1-\delta)\nu^{\mathbf{p}}(t-1) \tag{13}$$

$$\hat{\mathbf{s}}(t) = \delta\hat{\mathbf{s}}(t)+(1-\delta)\hat{\mathbf{s}}(t-1) \tag{14}$$

$$\nu^{\mathbf{s}}(t) = \delta\nu^{\mathbf{s}}(t)+(1-\delta)\nu^{\mathbf{s}}(t-1) \tag{15}$$

$$\hat{\mathbf{x}}(t) = \delta\hat{\mathbf{x}}(t)+(1-\delta)\hat{\mathbf{x}}(t-1) \tag{16}$$

The steps of FsGAMP are listed in Algorithm 2. Notation t indicates the iteration number; $\hat{\mathbf{x}}$ and $\nu^{\mathbf{x}}$ denote the mean and variance of $\mathbf{x}$; the element-wise product and division are denoted $\odot$ and $\oslash$. Function $g_{\text{out}}(\cdot)$ and $g_{\text{in}}(\cdot)$ calculate the Bayesian estimation of $\mathbf{z}$ and $\mathbf{x}$, respectively. Please refer to [9] (Table 1) and the Appendix to find their specific forms. The prior $p_{\mathrm{x}}(x_{S_j})$ can take a Gaussian, Laplacian or spike-and-slab distribution.

Algorithm 2 Fix Support GAMP.

Input: $\mathbf{y}; \mathbf{A}; U$
Output: $\hat{\mathbf{x}}; \nu^{\mathbf{x}}$
1: $\hat{\mathbf{x}} \leftarrow \mathbf{0}^{N\times 1}; \nu^{\mathbf{x}} \leftarrow \mathbf{0}^{N\times 1}; \hat{\mathbf{s}}(t=0) \leftarrow \mathbf{0}^{M\times 1}$
2: $\hat{\mathbf{x}}_U(t=1) \leftarrow \mathbf{0}^{|U|\times 1}; \nu^{\mathbf{x}_U}(t=1) \leftarrow \mathbf{0}^{|U|\times 1}$
3: **while** $stop =$ **false do**
4: $\quad \nu^{\mathbf{p}}(t) \leftarrow |\mathbf{A}_{:,U}|^2 \nu^{\mathbf{x}_U}(t)$
5: $\quad \hat{\mathbf{z}}(t) \leftarrow \mathbf{A}_{:,U}\hat{\mathbf{x}}_U(t)$
6: $\quad \hat{\mathbf{p}}(t) \leftarrow \hat{\mathbf{z}}(t) - \hat{\mathbf{s}}(t-1) \odot \nu^{\mathbf{p}}(t)$
7: $\quad [\hat{\mathbf{z}}_0(t), \nu^{\mathbf{z}_0}(t)] \leftarrow g_{\text{out}}(\hat{\mathbf{p}}(t), \nu^{\mathbf{p}}(t))$
8: $\quad \hat{\mathbf{s}}(t) \leftarrow (1 \oslash \nu^{\mathbf{p}}(t)) \odot (\hat{\mathbf{z}}_0(t) - \hat{\mathbf{p}}(t))$
9: $\quad \nu^{\mathbf{s}}(t) \leftarrow (1 \oslash \nu^{\mathbf{p}}(t)) \odot (1 - \nu^{\mathbf{z}_0(t)} \oslash \nu^{\mathbf{p}}(t))$
10: $\quad \nu^{\mathbf{r}_U}(t) \leftarrow 1 \oslash ((|\mathbf{A}_{:,U}|^2)^T \nu^{\mathbf{s}}(t))$
11: $\quad \hat{\mathbf{r}}_U(t) \leftarrow \hat{\mathbf{x}}_U(t) + \nu^{\mathbf{r}_U}(t) \odot (\mathbf{A}_{:,U}^T \hat{\mathbf{s}}(t))$
12: $\quad [\hat{\mathbf{x}}_U(t+1), \nu^{\mathbf{x}_U}(t+1)] \leftarrow g_{\text{in}}(\hat{\mathbf{r}}_U(t), \nu^{\mathbf{r}_U}(t))$
13: $\quad$ {damping steps}
14: $\quad$ **if** $\|\hat{\mathbf{x}}_U(t+1) - \hat{\mathbf{x}}_U(t)\|_2 / \|\hat{\mathbf{x}}_U(t)\|_2 \leq \delta$ **then**
15: $\quad\quad \hat{\mathbf{x}}(U) \leftarrow \hat{\mathbf{x}}_U(t+1), \nu^{\mathbf{x}}(U) \leftarrow \nu^{\mathbf{x}_U}(t+1)$
16: $\quad\quad stop \leftarrow$ TRUE
17: $\quad$ **end if**
18: **end while**

GAMP calculates all elements of $\mathbf{x}$ in the loop, while FsGAMP estimates at most $|U|$ non-zero elements of $\mathbf{x}$ in the loop. Since $|U|$ increases gradually and $|U| \leq K$, the memory footprint rate of FsGAMP is not more than K/N. Because $K \ll N$, the memory saving is remarkable.

2.3. Computational Complexity Discussion

The computational complexity of the GAMP algorithm is dominated by four matrix-vector multipliers per iteration [9]. The multipliers of FsGAMP coincide with GAMP: Lines 4, 6, 10 and 11. At each multiplication, the FsGAMP algorithm only computes $M \times |S|$ multipliers, while the GAMP algorithm needs $M \times N$ multipliers.

RrMpGAMP has two nested loops; the outer loop is RrMP; the inner loops are three FsGAMPs. Because the replacement/dropout operations in the RrMP algorithm are random, we cannot exactly calculate how many new indexes are chosen per iteration. However, if we roughly estimate the number of winners in these candidates to $s/2$ per iteration (usually, this number is greater than $s/2$ with the step in observation), the outer loop has $J = \lceil K/(s/2) \rceil + 2s$ iterations. Furthermore, $2s$ can be removed, since it is a small quantity compared to $\lceil K/(s/2) \rceil$. Because the number of iterations of GAMP (as well as FsGAMP) is also uncertain, we simply use the maximal iteration number B as the worst setting. Firstly, taking no account of M and B, we have:

$$\begin{aligned}(2s + \frac{s}{2}) + (2(\frac{s}{2} + s) + \frac{s}{2}) + (2(\frac{2s}{2} + s) + \frac{3s}{2}) + \cdots + (2(\frac{(J-1)s}{2} + s) + \frac{Js}{2}) \\ = 2[s + \frac{s}{2} + s + \frac{2s}{2} + s + \cdots + \frac{(J-1)s}{2} + s] + (\frac{s}{2} + \frac{2s}{2} + \frac{3s}{2} + \cdots + \frac{Js}{2}) \quad (17)\end{aligned}$$

where the first square brackets correspond to Lines 4–5 in Algorithm 1, and the second square brackets correspond to Line 9 in Algorithm 1. After summation and simplification, we have:

$$Equation\ (17) = \frac{Js(3J+7)}{4} \tag{18}$$

Secondly, considering M, B, four matrix-vector multiplies per iteration, and expanding J, we have:

$$Equation\ (18) \Rightarrow O(4MBJ^2s) = O(16MKB) \tag{19}$$

For GAMP, the computational complexity can be estimated as $O(4MNB)$, If $4K < N$, the computational complexity of RrMpGAMP is less than GAMP.

Using the matching pursuit-based methods to solve a M-row n-column linear system, the computing time mainly depends on the least squares step, especially when the number of equations M is large and the number of variables n is not small. For example, the Householder-LS method needs $O(2n^2(M-n/3))$ flops ([16], Algorithm 5.3.2). Because OMP finds one column index at a time and n increases from one to K, the computational complexity of OMP using Householder-LS is:

$$\begin{aligned}\sum_{n=1}^{K} O(2n^2(M-n/3)) &= O\Big(\sum_{n=1}^{K} 2Mn^2 - \sum_{n=1}^{K}\frac{2}{3}n^3\Big)\\ &= O\Big(2M\frac{K(K+1)(2K+1)}{6} - \frac{2}{3}(\frac{K(K+1)}{2})^2\Big)\\ &= O(MK^3)\end{aligned} \tag{20}$$

since $K \ll M$. We compare two computational complexity results. For RrMpGAMP, it is $O(16MBK)$; for OMP using Householder-LS, it is $O(MK^3)$. If $16B < K^2$, the computational complexity of RrMpGAMP is less than Householder-LS OMP.

3. Convergence Discussion

We use the replica method [17,18] to asymptotically describe the evaluation of the logarithm of the partition function of the posterior distribution Equation (4). We concentrate on the Sum-Product FsGAMP algorithm in this discussion. The main advantage is that replica computations give a statistical physical meaning to the linear system. The replica method was used for compressed sensing in recent years [19–22]; we follow the research of [23].

We use the zero-mean Gaussian matrix assumption in the derivation of FsGAMP and replica analysis, because FsGAMP just works on the RrMP pursued support set, and the cardinality of it is much smaller than the cardinality of a sparse vector, *i.e.*, $|S| \ll N$. We know that the brief propagation on the factor graph tends to diverge if there are many cycles on it [14]. When the number of cycles decreases, the brief propagation tends to converge. The factor graph form of compressed sensing is a bipart graph, and there are too many cycles on it. However, if we restrict the connections to the pursued variable nodes and sequentially construct the connections by some kind of pursuit strategy, such as RrMP pursuit, the number of cycles increases from zero to a relatively small integer, compared to the huge number of bipart connection cycles. These relatively small cycles and sequential pursuit method can be seen as a regularization to the non-zero-mean reality. This is our intuition.

3.1. The Replica Method Analysis of FsGAMP

The zero-mean Gaussian measurement noise is assumed an equivalent variance Δ for every entry of $\boldsymbol{\epsilon}$, and generally, $\Delta \neq \Delta_0$, where Δ_0 is the true noise variance. This assumption means that the noise variance (inside $v^{\mathbf{z}_0}(t)$ term), which is estimated by the FsGAMP algorithm, usually does not equal

the true noise variance. The influence of their difference is shown in Equation (74). We rewrite the definition of the CS problem as follows:

$$y_\mu = \sum_{i=1}^{N} A_{\mu i} s_i + \epsilon_\mu \quad \mu = 1, \ldots, M \tag{21}$$

where $\mathbf{s}$ is the original signal, and we use the notation s to replace notation x in Equation (1) to avoid ambiguity in the following derivation and use μ to replace m in accordance with the replica computations' convention. We denote $\alpha = M/N$ with the number of measurements per variable. In the asymptotic analysis, we are interested in the case of large system limitation $N \to \infty$, while keeping signal density $p_{x^0}(x^0)$ and measurement rate α of order one, where we let $x^0 = s$. It is also necessary to assume that the components of signal $\mathbf{s}$ and measurement $\mathbf{y}$ are also order one, such that we can consider the elements of sensing matrix $\mathbf{A}$ to be zero-mean. It is worth noting that the variance of the order of $\mathbf{A}$'s elements is $O(1/K)$, not $O(1/N)$, because we have pursued the indexes of non-zero elements. This makes it not necessary for the FsGAMP algorithm to summarize N entries in one row of $\mathbf{A}$. In this way, the variance range of $\mathbf{A}$'s elements enlarges.

Before the start of the computation, we briefly review the link between a Hamiltonian and linear system estimation problems (we know the form of compressed sensing is a linear system). The posterior distribution of a linear system in the Boltzmann form is:

$$\begin{aligned} p(\mathbf{x}|\mathbf{y}, \boldsymbol{\theta}) &= \frac{1}{Z(\boldsymbol{\theta})} e^{-E(\mathbf{x}|\mathbf{y}, \boldsymbol{\theta})} \\ \Rightarrow E(\mathbf{x}|\mathbf{y}, \boldsymbol{\theta}) &= -\log(p(\mathbf{x}|\boldsymbol{\theta})) - \log(p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta})) \end{aligned} \tag{22}$$

where $p(\mathbf{x}|\boldsymbol{\theta})$ is the prior, $\boldsymbol{\theta}$ are the model parameters, such as:

$$p(\mathbf{x}|\boldsymbol{\theta}) = \prod_{i=1}^{N} \left[(1-\rho)\delta(x_i) + \rho \mathcal{N}(x_i|\tilde{\boldsymbol{\theta}}) \right] \tag{23}$$

Additionally, $p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta})$ is likelihood:

$$p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}) = \prod_{\mu=1}^{M} \mathcal{N}(y_\mu|(\mathbf{A}\mathbf{x})_\mu, \boldsymbol{\theta}) \tag{24}$$

Then, we get the Hamiltonian:

$$E(\mathbf{x}|\mathbf{y}, \boldsymbol{\theta}) = -\sum_{i=1}^{N} \log(p_i(x_i)) + \frac{1}{\Delta} \sum_{\mu=1}^{M} (y_\mu - (\mathbf{A}\mathbf{x})_\mu)^2 + \frac{M}{2} \log(2\pi\Delta) \tag{25}$$

Now, we derive the potential of Equation (21) under the pursued $|S|$ column indexes' condition, as shown in Equation (4). The meaning of "potential" in inference theory equals the meaning of "Bethe free entropy" in statistical physics. For the simplification of notation, we ignore the uppercase letter S and just use the subscript, such as i, to indicate S_i in Equation (4). The aim is to sample a vector $\mathbf{x}$ from the probability measure:

$$\begin{aligned} & p_{\boldsymbol{x}|\boldsymbol{y}}(\mathbf{x}|\mathbf{y}) \\ =& \frac{1}{Z} \prod_{i=1}^{K} p_{\mathrm{x}_i}(x_i) \prod_{\mu=1}^{M} \frac{1}{\sqrt{2\pi\Delta}} e^{-\frac{[y_\mu - \sum_{i=1}^{K} A_{\mu i} x_i]^2}{2\Delta}} \\ =& \frac{1}{Z} \prod_{i=1}^{K} p_{\mathrm{x}_i}(x_) \prod_{\mu=1}^{M} \frac{1}{\sqrt{2\pi\Delta}} e^{-\frac{[\sum_{i=1}^{K} A_{\mu i}(x_i - s_i) + \epsilon_\mu]^2}{2\Delta}} \end{aligned} \tag{26}$$

Equation (26) can be seen as the Boltzmann measure on a disordered system with Hamiltonian:

$$H(\mathbf{x}) = -\sum_{i=1}^{K} \log[p_{\mathrm{x}_i}(x_i)] + \frac{[\sum_{i=1}^{K} A_{\mu i}(x_i - s_i) + \epsilon_\mu]^2}{2\Delta} \tag{27}$$

where the partition function Z is the normalization constant of the full posterior distribution Equation (26) and $\mathbf{x}$ is the configuration of the Hamiltonian. The thermodynamic properties of the disordered system are characterized by the average free entropy $\mathbb{E}_{\mathbf{A},\mathbf{s},\boldsymbol{\epsilon}}(\log Z)$, where:

$$Z(\mathbf{A},\mathbf{s},\boldsymbol{\epsilon}) = \int \Big[\prod_{i=1}^{K} \mathrm{d}x_i \prod_{i=1}^{K} p_{\mathrm{x}_i}(x_i)\Big] \prod_{\mu=1}^{M} \frac{1}{\sqrt{2\pi\Delta}} e^{-\frac{[\sum_{i=1}^{K} A_{\mu i}(x_i - s_i) + \epsilon_\mu]^2}{2\Delta}} \tag{28}$$

Because the expectation of the logarithm function is a hard problem, the average free entropy (also called the Bethe free entropy) can be evaluated via the replica trick as:

$$\Phi \equiv \lim_{K\to\infty} \frac{1}{K} \mathbb{E}_{\mathbf{A},\mathbf{s},\boldsymbol{\epsilon}}\{\log Z\} = \lim_{K\to\infty} \frac{1}{K} \lim_{n\to 0} \frac{\mathbb{E}_{\mathbf{A},\mathbf{s},\boldsymbol{\epsilon}}\{Z^n\} - 1}{n} \tag{29}$$

where n independent replicas are introduced to reform the original disordered system to a new system (*i.e.*, the name of the method) and $\mathbb{E}_{\mathbf{A},\mathbf{s},\boldsymbol{\epsilon}}$ is the average over all of the sources of disorder in Equation (21). Each configuration of the new system is indexed by an n-tuple $\boldsymbol{i} = (i_1, \ldots, i_n)$, where each $i_a, a = 1, \ldots, n$ has a continuous non-zero state value because i_a corresponds to a non-zero element in $\mathbf{x}$, which has been found by the matching pursuit process of the RrMP algorithm.

Now, the problem of computing the free energy is converted into computing the n-th moment of the partition function Z.

$$[Z(\mathbf{A},\mathbf{s},\boldsymbol{\epsilon})]^n = \frac{1}{(2\pi\Delta)^{\frac{Mn}{2}}} \int \Big[\prod_{i,a} \mathrm{d}x_i^a \prod_{i,a} p_{\mathrm{x}_i^a}(x_i^a)\Big] \prod_{\mu} e^{-\frac{\sum_{a=1}^{n}[\sum_{i=1}^{K} A_{\mu i} s_i + \epsilon_\mu - \sum_{i=1}^{K} A_{\mu i} x_i^a]^2}{2\Delta}} \tag{30}$$

The average replicated partition function can be rearranged as:

$$\mathbb{E}_{\mathbf{A},\mathbf{s},\boldsymbol{\epsilon}}\{Z^n\} = \frac{1}{(2\pi\Delta)^{\frac{Mn}{2}}} \int \Big[\prod_{i,a} \mathrm{d}x_i^a \prod_{i,a} p_{\mathrm{x}_i^a}(x_i^a)\Big] \prod_{\mu} \mathbb{E}_{\mathbf{A},\mathbf{s},\boldsymbol{\epsilon}}\Big\{e^{-\frac{\sum_{a=1}^{n}[\sum_{i=1}^{K} A_{\mu i} s_i + \epsilon_\mu - \sum_{i=1}^{K} A_{\mu i} x_i^a]^2}{2\Delta}}\Big\} \tag{31}$$

where $a, b, \ldots, n$ denote the replica indexes. Define:

$$X_\mu = \mathbb{E}_{\mathbf{A},\boldsymbol{\epsilon}}\Big\{e^{-\frac{\sum_{a=1}^{n}[\sum_{i=1}^{K} A_{\mu i} s_i + \epsilon_\mu - \sum_{i=1}^{K} A_{\mu i} x_i^a]^2}{2\Delta}}\Big\} \tag{32}$$

Equation (31) can be rewritten to:

$$\mathbb{E}_{\mathbf{A},\mathbf{s},\boldsymbol{\epsilon}}\{Z^n\} = \frac{1}{(2\pi\Delta)^{\frac{Mn}{2}}} \mathbb{E}_{\mathbf{s}}\Big\{\int \Big[\prod_{i,a} \mathrm{d}x_i^a \prod_{i,a} p_{\mathrm{x}_i^a}(x_i^a)\Big] \prod_{\mu=1}^{M} X_\mu\Big\} \tag{33}$$

In order to compute X_μ, firstly, we should define a variable:

$$v_\mu^a = \sum_{i=1}^{K} A_{\mu i}(x_i^0 - x_i^a) + \epsilon_\mu, \quad a = 1, \ldots, n \tag{34}$$

where $x_i^0 \triangleq s_i$, to help the following derivation. By applying the central limit theorem to v_μ^a, since it is a sum of *i.i.d.* Gaussian terms of the sensing matrix $\mathbf{A}$ at a fixed signal $\mathbf{s}$ and configuration $\mathbf{x}$, we conclude that v_μ^a obeys a joint Gaussian distribution. When the matrix $\mathbf{A}$ has *i.i.d.* elements with zero-mean and

variance $1/K$, we introduce the order parameters to summarize an amount of microscopic states of the replicas to four macroscopic states as follows:

$$u = \frac{1}{K}\sum_{i=1}^{K}(s_i)^2 \tag{35}$$

$$m^a = \frac{1}{K}\sum_{i=1}^{K}x_i^a s_i, \quad a = 1,\ldots,n \tag{36}$$

$$Q^a = \frac{1}{K}\sum_{i=1}^{K}(x_i^a)^2, \quad a = 1,\ldots,n \tag{37}$$

$$q^{ab} = \frac{1}{K}\sum_{i=1}^{K}x_i^a x_i^b, \quad a < b, b = 2,\ldots,n \tag{38}$$

These parameters constitute a matrix, called the overlap matrix:

$$\mathbf{Q} = \begin{pmatrix} u & m^1 & m^2 & \cdots & m^n \\ m^1 & Q^1 & q^{12} & \cdots & q^{1n} \\ m^2 & q^{12} & Q^2 & \ddots & q^{2n} \\ \vdots & \vdots & \ddots & \ddots & q^{(n-1)n} \\ m^n & q^{1n} & \cdots & q^{(n-1)n} & Q^n \end{pmatrix} \tag{39}$$

Furthermore, according to the fact that both $\mathbf{A}$ and $\boldsymbol{\epsilon}$ are zero-mean, we conclude the first two moments of the joint Gaussian distribution:

$$\mathbb{E}_{\mathbf{A},\boldsymbol{\epsilon}}\{v_\mu^a\} = 0 \tag{40}$$

$$\begin{aligned}\mathbb{E}_{\mathbf{A},\boldsymbol{\epsilon}}\{(v_\mu^a)^2\} &= \mathbb{E}_{\mathbf{A},\boldsymbol{\epsilon}}\Big\{\sum_i A_{\mu i}^2(x_i^0 - x_i^a)^2\Big\} + \Delta_0 \\ &= \frac{1}{K}\sum_i (x_i^0 - x_i^a)^2 + \Delta_0 \\ &= Q^a - 2m^a + < s^2 > + \Delta_0 \end{aligned} \tag{41}$$

$$\begin{aligned}\mathbb{E}_{\mathbf{A},\boldsymbol{\epsilon}}\{v_\mu^a v_\mu^b\} &= \mathbb{E}_{\mathbf{A},\boldsymbol{\epsilon}}\Big\{\sum_i A_{\mu i}^2(x_i^0 - x_i^a)(x_i^0 - x_i^b)\Big\} + \Delta_0 \\ &= q^{ab} - (m^a + m^b) + < s^2 > + \Delta_0 \end{aligned} \tag{42}$$

where $< s >^2 = \int s^2 p_{\mathrm{s}}(s)\mathrm{d}s$. The dependence on the measurement index μ of v_μ and X_μ is canceled due to the averaging; therefore, we note $v_\mu = v$ and $X_\mu = X$ for simplicity. In order to further calculate the expectation in Equation (33), we apply the replica symmetric ansatz:

$$m^a = m, \forall a, Q^a = Q, \forall a, q^{ab} = q, \forall (a, b : a \neq b) \tag{43}$$

which is valid for the inference problem on a locally tree-like or highly dense factor graph under the prior matching condition. Fortunately, by the matching pursuit process in RrMP, we found the positions of non-zero elements in the signal, such that the condition has been satisfied. Based on this ansatz, Equation (39) equals:

$$\mathbf{Q} = \begin{pmatrix} u & m & m & \cdots & m \\ m & Q & q & \cdots & q \\ m & q & Q & \ddots & q \\ \vdots & \vdots & \ddots & \ddots & q \\ m & q & \cdots & q & Q \end{pmatrix} \tag{44}$$

We want to compute:

$$X = \mathbb{E}_{\mathbf{v}}\Big\{e^{-(1/2\Delta)\sum_{a=1}^{n}(v^a)^2}\Big\} \tag{45}$$

with a probability distribution:

$$p(\mathbf{v}) = \frac{1}{\sqrt{(2\pi)^n \det(\mathbf{G})}} e^{-(1/2)\sum_{a,b} v^a (\mathbf{G}^{-1})_{ab} v^b} \tag{46}$$

where the covariance matrix $\mathbf{G}$ of $\{v^a\}$ under the replica symmetric ansatz is given by:

$$G_{aa} = \mathbb{E}_{\mathbf{v}}\{v^a v^a\} = Q - 2m + < s^2 > + \Delta_0 \tag{47}$$

$$G_{ab} = \mathbb{E}_{\mathbf{v}}\{v^a v^b\} = q - 2m + < s^2 > + \Delta_0 \tag{48}$$

such that:

$$\mathbf{G} = (< s^2 > -2m + q + \Delta_0)\mathbf{1}_n + (Q - q)\mathbf{I}_n \tag{49}$$

where $\mathbf{1}_n$ is a $n \times n$ matrix with elements all equivalent to one and $\mathbf{I}_n$ is an unitary diagonal matrix.

Then, Equation (45) equals:

$$\begin{aligned} X &= \frac{1}{\sqrt{(2\pi)^n \det(\mathbf{G})}} \int (\mathrm{d}\mathbf{v}) e^{-\frac{1}{2}\mathbf{v}^T(\mathbf{G}^{-1} + \mathbf{I}_n/\Delta)\mathbf{v}} \\ &= \frac{1}{\sqrt{\det(\mathbf{I}_n + \mathbf{G}/\Delta)}} \end{aligned} \tag{50}$$

The eigenvectors of $\mathbf{G}$ can be divided into two categories: one eigenvector of the form $(1, 1, \ldots, 1)$ with associated eigenvalue $Q - q + n(q - 2m + < s^2 > + \Delta_0)$, and $n - 1$ eigenvectors of the form $(0, \ldots, 0, -1, 1, 0, \ldots, 0)$ with eigenvalues $Q - q$, where the couple $(-1, 1)$ shifts one by one. Now, we have:

$$\det\Big(\mathbf{I}_n + \frac{\mathbf{G}}{\Delta}\Big) = \frac{1 + \frac{1}{\Delta}(Q - q + n(< s^2 > -2m + q + \Delta_0))}{(1 + \frac{Q-q}{\Delta})^{1-n}} \tag{51}$$

such that:

$$\lim_{n \to 0} X = e^{-\frac{n}{2}[\frac{< s^2 > -2m + q + \Delta_0}{Q - q + \Delta} + \log(\frac{Q - q + \Delta}{\Delta})]} \tag{52}$$

Now, back to Equation (33), we need to guarantee that the order parameters m, q, Q coincide with their definition Equation (35), and this guarantee can be obtained by Dirac functions:

$$\delta\Big(\sum_{i=1}^{K} x_i^a s_i - K m^a\Big), \quad a = 1, \ldots, n \tag{53}$$

$$\delta\Big(\sum_{i=1}^{K} (x_i^a)^2 - K Q^a\Big), \quad a = 1, \ldots, n \tag{54}$$

$$\delta\Big(\sum_{i=1}^{K} x_i^a x_i^b - K q^{ab}\Big), \quad a < b, b = 2, \ldots, n \tag{55}$$

Since Dirac functions in the frequency field are the Fourier transform of const $1/2\pi$ in the time field, we have:

$$\delta(\sum_{i=1}^{K} x_i^a s_i - Km^a) = \int \mathrm{d}\tilde{m}^a \frac{1}{2\pi} e^{j\tilde{m}^a[Km^a - \sum_{i=1}^{K} x_i^a s_i]} \tag{56}$$

$$\delta(\sum_{i=1}^{K} (x_i^a)^2 - KQ^a) = \int \mathrm{d}\tilde{Q}^a \frac{1}{2\pi} e^{j\tilde{Q}^a[KQ^a - \sum_{i=1}^{K} (x_i)^2]} \tag{57}$$

$$\delta(\sum_{i=1}^{K} x_i^a x_i^b - Kq^{ab}) = \int \mathrm{d}\tilde{q}^{ab} \frac{1}{2\pi} e^{j\tilde{q}^{ab}[Kq^{ab} - \sum_{i=1}^{K} x_i^a x_i^b]} \tag{58}$$

where $j = \sqrt{-1}$. It is worth noting that $\tilde{m}^a$,$\tilde{Q}^a$ and $\tilde{q}^{ab}$ also satisfy the replica symmetric ansatz. Rewriting const one in the time field as the inverse Fourier transform of Dirac function $2\pi\delta(\cdot)$ in the frequency field and substituting the right-hand side of Equations (56)–(58), we obtain:

$$1 = \int \mathrm{d}m^a \Big[\int \mathrm{d}\tilde{m}^a e^{-\tilde{m}^a[Km^a - \sum_{i=1}^{K} x_i^a s_i]} \Big] \tag{59}$$

$$1 = \int \mathrm{d}Q^a \Big[\int \mathrm{d}\tilde{Q}^a e^{-\tilde{Q}^a[\frac{K}{2}Q^a - \frac{K}{2}\sum_{i=1}^{K} (x_i^a)^2]} \Big] \tag{60}$$

$$1 = \int \mathrm{d}q^{ab} \Big[\int \mathrm{d}\tilde{q}^{ab} e^{-\tilde{q}^a[Kq^{ab} - \sum_{i=1}^{K} x_i^a x_i^b]} \Big] \tag{61}$$

and consider n replicas:

$$\begin{aligned} 1 = & \int \Big[\prod_{a}^{n} \mathrm{d}\tilde{Q}^a \mathrm{d}Q^a \mathrm{d}\tilde{m}^a \mathrm{d}m^a \Big] \Big[\int \prod_{b,a<b}^{n,(n-1)/2} \mathrm{d}\tilde{q}^{ab} \mathrm{d}q^{ab} \Big] \\ & \times \exp\Big\{ \sum_{a}^{n} \tilde{m}^a (Km^a - \sum_{i}^{K} x_i^a x^0) + \sum_{a}^{n} \tilde{Q}^a \Big(\frac{K}{2} Q^a - \frac{1}{2} \sum_{i}^{K} (x_i^a)^2 \Big) - \sum_{b,a\neq b}^{n,(n-1)} \tilde{q}^{ab} \Big(Kq^{ab} - \sum_{i}^{K} x_i^a x_i^b \Big) \Big\} \end{aligned} \tag{62}$$

Plugging Equation (62) into Equation (33), we obtain:

$$\begin{aligned} & \mathbb{E}_{\mathbf{A},\mathbf{s},\boldsymbol{\epsilon}}\{Z^n\} \\ & = \frac{1}{(2\pi\Delta)^{\frac{Mn}{2}}} \int \Big[\prod_{a}^{n} \mathrm{d}\tilde{Q}^a \mathrm{d}Q^a \mathrm{d}\tilde{m}^a \mathrm{d}m^a \Big] \\ & \Big[\int \prod_{b,a<b}^{n,(n-1)/2} \mathrm{d}\tilde{q}^{ab} \mathrm{d}q^{ab} e^{K\left[\frac{1}{2}\sum_{a} \tilde{Q}^a Q^a - \frac{1}{2} \sum_{b,a\neq b}^{n,(n-1)} \tilde{q}^{ab} q^{ab} - \sum_{a} \tilde{m}^a m^a\right]} \prod_{\mu}^{M} X \Big] \\ & \times \Big\{ \int \mathrm{d}x^0 p_{\mathrm{x}^0}(x^0) \prod_{a}^{n} \mathrm{d}x^a p_{\mathrm{x}^a}(x^a) e^{-\frac{1}{2}\sum_{a}^{n} \tilde{Q}^a (x^a)^2 + \frac{1}{2}\sum_{b,a\neq b}^{n,(n-1)} \tilde{q}^{ab} x^a x^b + \sum_{a}^{n} \tilde{m}^a x^a x^0} \Big\}^K \end{aligned} \tag{63}$$

We denote Γ as the integration in the $\{\cdot\}^K$. The exponential part of Γ has many cross terms of n replicas, such that we should decouple them by linearizing the exponent. According to the Hubbard–Stratonovich transform:

$$e^{\frac{y}{2}} = \frac{1}{\sqrt{2\pi}} \int_{\mathbb{R}} e^{\pm z\sqrt{y} - \frac{z^2}{2}} \mathrm{d}z \triangleq \int_{\mathbb{R}} e^{\pm z\sqrt{y}} \mathcal{D}z \tag{64}$$

with the Gaussian measure:

$$\mathcal{D}z = \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}} \mathrm{d}z \tag{65}$$

and the square completion:

$$(\sum_{a=1}^{n} x^a)^2 = 2\sum_{b=a+1}^{n} x^a x^b + \sum_{a=1}^{n} (x^a)^2 \tag{66}$$

we have:

$$\begin{aligned} e^{\frac{\tilde{q}\sum_{b,a\neq b}^{n,(n-1)} x^a x^b}{2}} &= \int e^{z\sqrt{\tilde{q}}(\sum_a^n x^a) - \frac{\tilde{q}}{2}\sum_a^n (x^a)^2} \mathcal{D}z \\ &= \int e^{z\sqrt{\tilde{q}}(\sum_a^n x^a)} e^{-\frac{\tilde{q}}{2}\sum_a^n (x^a)^2} \mathcal{D}z \end{aligned} \tag{67}$$

Using the replica symmetric ansatz again, we obtain:

$$\Gamma = \int \mathrm{d}x^0 p_{x^0}(x^0) \int \mathcal{D}z \left(\int \mathrm{d}x p_x(x) e^{-\frac{1}{2}(\tilde{Q}+\tilde{q})x^2 + \tilde{m}x(x^0) + z\sqrt{\tilde{q}}x} \right)^n \tag{68}$$

Define $f(z, x^0) \triangleq \int \mathrm{d}x p_x(x) e^{-\frac{1}{2}(\tilde{Q}+\tilde{q})x^2 + \tilde{m}x(x^0) + z\sqrt{\tilde{q}}x}$; since $\lim_{n\to 0}(f(z,x^0))^n = 1 + n\log f(z,x^0)$, we have:

$$\begin{aligned} \lim_{n\to 0} \int (f(z,x^0))^n \mathcal{D}z &= \int \lim_{n\to 0} (f(z,x^0))^n \mathcal{D}z \\ &= 1 + n\int \log f(z,x^0) \mathcal{D}z \\ &\approx e^{n\int \log f(z,x^0)\mathcal{D}z} \end{aligned} \tag{69}$$

such that:

$$\begin{aligned} \lim_{n\to 0} \Gamma &\approx \int \mathrm{d}x^0 p_{x^0}(x^0) \Big[1 + n \int \log f(z,x^0) \mathcal{D}z \Big] \\ &= \int p_{x^0}(x^0) \mathrm{d}x^0 + \int \mathrm{d}x^0 p_{x^0}(x^0) \Big[n \int \log f(z,x^0) \mathcal{D}z \Big] \\ &= 1 + n \int \big[\mathrm{d}x^0 \mathcal{D}z \big] p_{x^0}(x^0) \log f(z,x^0) \\ &\approx e^{n\int [\mathrm{d}x^0 \mathcal{D}z] p_{x^0}(x^0) \log f(z,x^0)} \end{aligned} \tag{70}$$

Now, we combine Equations (52) and (70) with Equation (63) and use the replica symmetric ansatz again:

$$\lim_{n\to 0} \mathbb{E}_{\mathbf{A},\mathbf{s},\boldsymbol{\epsilon}} \{Z^n\} \approx \int \mathrm{d}\tilde{Q}\mathrm{d}Q\mathrm{d}\tilde{m}\mathrm{d}m\mathrm{d}\tilde{q}\mathrm{d}q e^{nK\Phi(\tilde{Q},Q,\tilde{m},m,\tilde{q},q)} \tag{71}$$

where:

$$\begin{aligned} &\Phi(\tilde{Q}, Q, \tilde{m}, m, \tilde{q}, q) \\ &= \frac{1}{2}(\tilde{Q}Q - 2\tilde{m}m + \tilde{q}q) \\ &- \frac{1}{2}\frac{M}{K}\Big(\frac{< s^2 > -2m + q + \Delta_0}{Q - q + \Delta} + \log(Q - q + \Delta) - \log\Delta \Big) \\ &+ \int [\mathrm{d}x^0 \mathcal{D}z] p_{x^0}(x^0) \log \int \mathrm{d}x p_x(x) e^{-\frac{1}{2}(\tilde{Q}+\tilde{q})x^2 + \tilde{m}x(x^0) + z\sqrt{\tilde{q}}x} \end{aligned} \tag{72}$$

The integration Equation (71) is intractable, otherwise. We use the saddle point method to estimate this integration by taking the optimum of $\Phi(\tilde{Q}, Q, \tilde{m}, m, \tilde{q}, q)$. It is worth noting that the replica trick needs $\lim_{n\to 0}\lim_{K\to\infty}(\cdot)$, but the saddle point estimation needs $\lim_{K\to\infty}\lim_{n\to 0}(\cdot)$; furthermore, Equation (71) is obtained under the condition $\lim_{n\to 0}(\cdot)$. Therefore, we assume that the limits can be exchanged, and the saddle point estimation of the integration Equation (71) should be done before we

really let $\lim_{n\to 0}(\cdot)$. These assumptions are not rigorous, but verified in the inference problem. Using the saddle point method and making derivatives for m, Q, q, respectively, we get:

$$\frac{\partial \Phi}{\partial m} = 0 \Rightarrow \tilde{m} = \frac{M}{K}\frac{1}{Q-q+\Delta} \tag{73}$$

$$\frac{\partial \Phi}{\partial Q} = 0 \Rightarrow \tilde{Q} = \frac{M}{K}\frac{-<s^2>+2m-2q+Q+\Delta-\Delta_0}{(Q-q+\Delta)^2} \tag{74}$$

$$\frac{\partial \Phi}{\partial q} = 0 \Rightarrow \tilde{q} = \frac{M}{K}\frac{<s^2>-2m+q+\Delta_0}{(Q-q+\Delta)^2} \tag{75}$$

and:

$$\tilde{m} = \tilde{Q} + \tilde{q} \tag{76}$$

Using the saddle point method and making derivatives for $\tilde{m}$,$\tilde{Q}$, $\tilde{q}$, respectively, and substituting Equation (76) for the derivatives, we get:

$$\frac{\partial \Phi}{\partial \tilde{m}} = 0 \Rightarrow m = \int \mathrm{d}x^0 \Big[x^0 p_{\mathrm{x}^0}(x^0) \int g_{\mathrm{mean}}(x^0 + z\frac{\sqrt{\tilde{q}}}{\tilde{m}}, \frac{1}{\tilde{m}})\mathcal{D}z \Big] \tag{77}$$

$$\frac{\partial \Phi}{\partial \tilde{q}} = 0 \Rightarrow q = \int \mathrm{d}x^0 \Big[p_{\mathrm{x}^0}(x^0) \int \Big(g_{\mathrm{mean}}(x^0 + z\frac{\sqrt{\tilde{q}}}{\tilde{m}}, \frac{1}{\tilde{m}}) \Big)^2 \mathcal{D}z \Big] \tag{78}$$

$$\frac{\partial \Phi}{\partial \tilde{Q}} = 0 \Rightarrow Q = \Big\{ \int \mathrm{d}x^0 \Big[p_{\mathrm{x}^0}(x^0) \int g_{\mathrm{var}}(x^0 + z\frac{\sqrt{\tilde{q}}}{\tilde{m}}, \frac{1}{\tilde{m}})\mathcal{D}z \Big] \Big\} + q \tag{79}$$

where the input end probability distribution is defined as:

$$p_{\mathrm{x}|\mathrm{r}}(x|\hat{r}, v^r) \triangleq \frac{p_{\mathrm{x}}(x)\mathcal{N}(x;\hat{r}, v^r)}{\int p_{\mathrm{x}}(x)\mathcal{N}(x;\hat{r}, v^r)\mathrm{d}x} \tag{80}$$

Then, the mean function is given by:

$$g_{\mathrm{mean}}(\hat{r}, v^r) = \int x p_{\mathrm{x}|\mathrm{r}}(x|\hat{r}, v^r)\mathrm{d}x \tag{81}$$

and the variance function is given by:

$$g_{\mathrm{var}}(\hat{r}, v^r) = \Big[\int x^2 p_{\mathrm{x}|\mathrm{r}}(x|\hat{r}, v^r)\mathrm{d}x \Big] - \big(g_{\mathrm{mean}}(\hat{r}, v^r) \big)^2 \tag{82}$$

In fact, Equations (81) and (82) correspond to the results of the $g_{\mathrm{in}}(\cdot)$ function in Algorithm 2.

3.2. The Prior Matching Conditions and Nishimori Conditions

The replica symmetric ansatz means that all of the replicas belong to the same state configuration, such that every replica has the same statistical properties. This state configuration is given by the macroscopic order parameters m, Q and q. Based on the replica symmetric ansatz, m is a correlation between the original sparse signal $\mathbf{s}$ and its estimation $\mathbf{x}$ about all of the replicas; then, we have:

$$m = \mathbb{E}_{\mathbf{y}}\{ s_i \mathbb{E}_{\mathrm{x}|\mathbf{y}}\{x_i\} \} \tag{83}$$

Similarly, Q is a self-correlation of these replicas; then, we have:

$$Q = \mathbb{E}_{\mathbf{y}}\{ \mathbb{E}_{\mathrm{x}|\mathbf{y}}\{x_i^2\} \} = \mathbb{E}_{\mathbf{s}}\{s_i^2\} \tag{84}$$

Another parameter q is the correlation between these replicas. As soon as the measurement and the original signal satisfy the correct reconstruction condition, such as the Restricted Isometry Property (RIP), the differences between these replicas are reflected in the measurement noise. Therefore,

averaging the correlations with an infinitely large number of the replicas will eliminate fluctuations caused by the noise and retain the energy of the signals; then, we have:

$$q = \mathbb{E}_{\mathbf{y}}\{\mathbb{E}_{\mathbf{x}|\mathbf{y}}\{x_i\}\mathbb{E}_{\mathbf{x}|\mathbf{y}}\{x_i\}\} = \mathbb{E}_{\mathbf{y}}\{s_i\mathbb{E}_{\mathbf{x}|\mathbf{y}}\{x_i\}\} = m \tag{85}$$

Now, we consider the average variance $V(t)$ and the mean-squared error $E(t)$:

$$V(t) \triangleq \frac{1}{K}\sum_{i}^{K} g_{\text{var}}(\hat{r}(t), \nu^r(t)) \tag{86}$$

$$E(t) \triangleq \frac{1}{K}\sum_{i}^{K} \left(g_{\text{mean}}(\hat{r}(t), \nu^r(t)) - s_i\right)^2 \tag{87}$$

According to the derivation of belief propagation and Equation (86), the probability distribution of x_i at the $(t+1)$-th iteration is:

$$p(x_i, t+1) \approx \frac{1}{Z} p_{\mathbf{x}_i}(x_i) e^{-\frac{M}{K}\frac{x_i - s_i - z\sqrt{E(t)+\frac{M}{K}\Delta_0}}{2(V(t)+\Delta)}} \tag{88}$$

where Z is a normalization const. Then, the average variance $V(t+1)$ and the mean-squared error $E(t+1)$ are:

$$V(t+1) = \int \mathrm{d}s\mathcal{D}z\; p_{\mathbf{s}}(s) g_{\text{var}}\left(s + z\sqrt{\frac{E(t)+\Delta_0}{M/K}}, \frac{V(t)+\Delta}{M/K}\right) \tag{89}$$

$$E(t+1) = \int \mathrm{d}s\mathcal{D}z\; p_{\mathbf{s}}(s)\left[g_{\text{mean}}\left(s + z\sqrt{\frac{E(t)+\Delta_0}{M/K}}, \frac{V(t)+\Delta}{M/K}\right) - s\right]^2 \tag{90}$$

Combining Equations (73), (75), (79), (89) and (90), and noticing that $s = x^0$, we get:

$$V(t+1) = Q - q \tag{91}$$

$$E(t+1) = < s^2 > -2m + q \tag{92}$$

Furthermore, as soon as we assume:

$$Q = < s^2 >, \tag{93}$$

and remember:

$$q = m, \tag{94}$$

and if the two conditions are satisfied at the fix point, then we have:

$$V(t+1) = E(t+1), \tag{95}$$

which means the average variance and the mean-squared error are equivalent. We call Equations (94) and (93) the prior matching conditions and Equations (95), (83), (84) and (85) the Nishimori conditions. When these conditions are satisfied, the convergence of the FsGAMP algorithm can be guaranteed.

4. Experiments

We compare multiple probe length RrMpGAMP, e.g., RrMpGAMP-L4, with five well-known algorithms: OMP, CoSaMP, basis pursuit (BP) with the interior point method to solve linear programming, AMP and GAMP, where the notation s in Section 2 is replaced with L to avoid confusion. We use MATLAB (Version R2014b)'s *pinv* function to implement the LS calculation in OMP and CoSaMP and modify CoSaMP by adding a new LS step before the prune step to improve the estimation accuracy.

Notice that *pinv* is the matrix Moore–Penrose pseudoinverse based on an SVD decomposition. It is suitable for the case that the matrix has more rows than columns and is not of full rank; then, the overdetermined least squares problem does not have a unique solution. The implementation of OMP and CoSaMP comes from Needell's work [24]; BP comes from the software package ℓ_1-MAGIC [25]; AMP comes from Kamilov's work [26]; and GAMP comes from *gamplab* [27].

We do $Q = 60$ trials in each experiment and use the relative mean square error (RMSE): $\xi = 1/Q \times \sum_{l=1}^{Q} \|\hat{\mathbf{x}}^l - \mathbf{x}^l\|_2 / \|\mathbf{x}^l\|_2$ as the performance metric. Set $M = 128, N = 256$ in these experiments, except the last experiment. Define the sparsity-measurement ratio $\rho \triangleq K/M$ and choose K locations at random as the support of $\mathbf{x}$. The amplitude of non-zero entry x_i is independently drawn from $N(x; 0, 1)$. The signal-to-noise ratios (SNR) are calculated by $SNR = 10\log_{10}(\|\mathbf{Ax}\|^2/(M\nu))$.

4.1. Zero-Mean Gaussian Projection Matrix Cases

In the first two experiments, the elements of $\mathbf{A}$ are independently drawn from $N(a; 0, 1/N)$, and the columns are normalized. The first experiment, termed the ν test, investigates the reconstruction performance *versus* noise power ν. Fixing $K = 20$, the range of ν is $[10^{-6}, 10^{-5}, 10^{-4}, 5\times10^{-4}, 10^{-3}, 5\times10^{-3}, 10^{-2}]$. Figure 2a shows that four types $(L = 2/4/6/8)$ of RrMpGAMP perform the same as GAMP, better than OMP and CoSaMP and much better than BP and AMP.

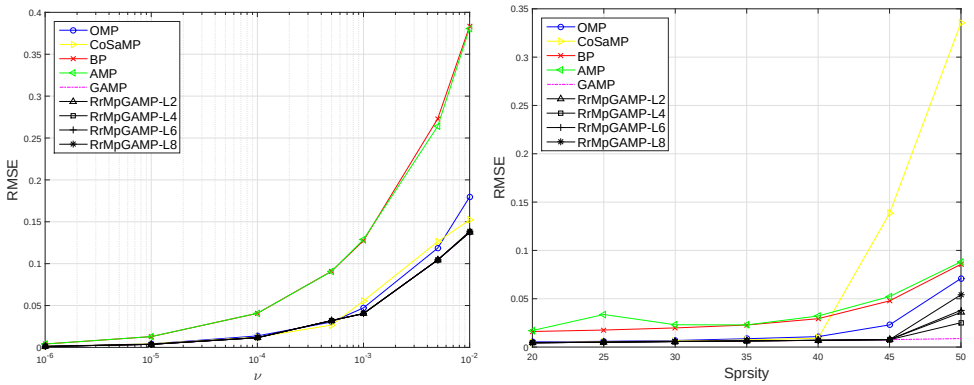

Figure 2. Reconstruction performance comparison using the zero-mean Gaussian projection matrix. (**a**) Noise power test; (**b**) Sparsity range test.

The second experiment, termed the sparsity test, investigates the reconstruction performance *versus* sparsity K. Fixing $SNR = 40$ dB, the range of K is $[20, 25, 30, 35, 40, 45, 50]$. Figure 2b shows that four types $(L = 2/4/6/8)$ of RrMpGAMP all perform the same as GAMP when $K \leq 45$ or equally $\rho \leq 0.35$. Notice that CoSaMP degrades obviously when $K \geq 40$, and RrMpGAMP-L4 is the best one in the $(M = 128, N = 256)$ setting, so we choose RrMpGAMP-L4 for the next three experiments.

4.2. More General Projection Matrix Cases

The third experiment investigates the reconstruction performance of various sparse projection matrices. Define the sparsity ratio of matrix $\mathbf{A}$ as η; set the range of η to $[0.1, 0.2, \ldots, 0.9]$; and fix $K = 20$, $SNR = 20$ dB. Figure 3a shows that RrMpGAMP-L4 performs the same as GAMP when $\eta > 0.2$ and evidently better than the other four competitors.

For AMP and GAMP, although good performance can be achieved by zero-mean *i.i.d.* matrix $\mathbf{A}$, it tends to drastically decline even for a small positive bias. The fourth experiment, termed the γ test, shows this phenomenon. The elements of $\mathbf{A}$ are independently drawn from a Gaussian distribution $N(a; \gamma/N, 1/N)$, where the mean is controlled by a positive parameter γ. Set the range of γ to $[1, 1.6, 1.8, 2, 2.2, 2.4, 3.4, 3.6, 3.8, 4, 5, 10, 20, 40, 60, 80, 100]$, and fix $K = 20, SNR = 20$ dB,

Figure 3b shows that GAMP violently diverges at $\gamma = 2$; AMP diverges at $\gamma = 4$; but RrMpGAMP-L4 maintains good performance until $\gamma = 40$. Although OMP, CoSaMP and BP also work well, RrMpGAMP-L4 performs a little better than OMP and CoSaMP when $\gamma < 40$ and obviously better than BP.

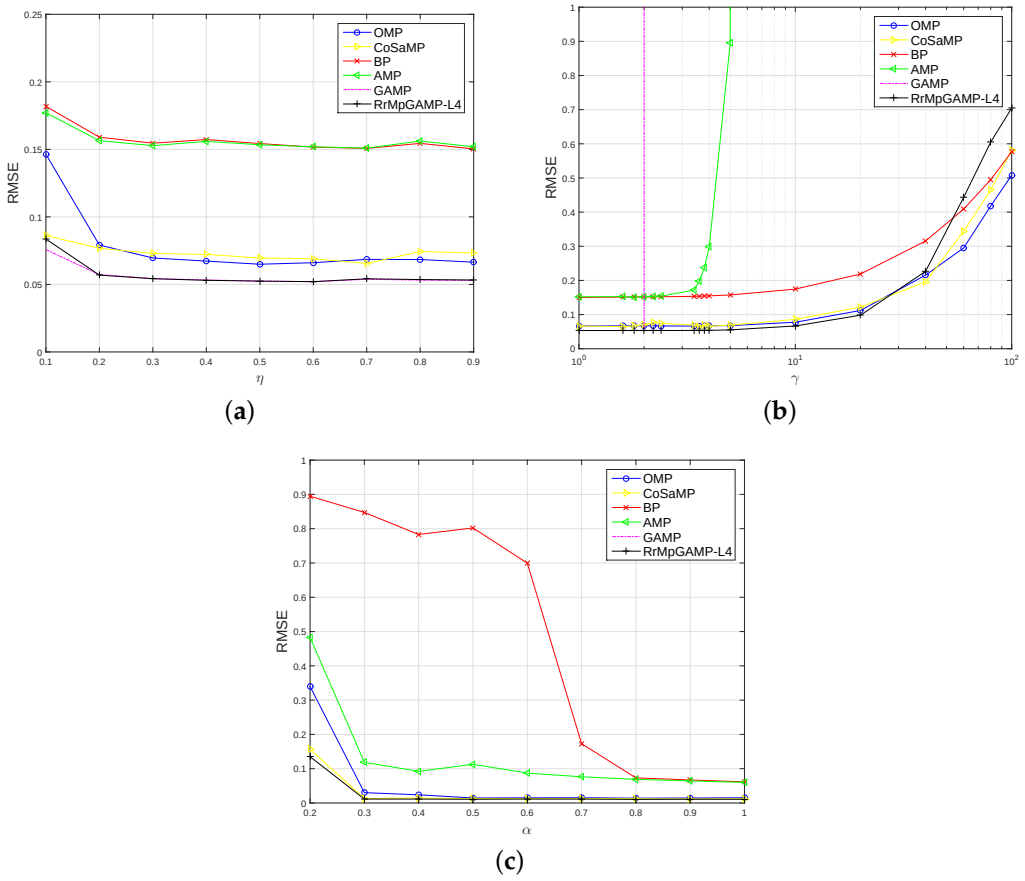

Figure 3. Reconstruction performance comparison using three general form projection matrices. (**a**) Sparse matrix test; (**b**) Non-zero-mean matrix test; (**c**) Strong correlated matrix test.

The fifth experiment, termed the α test, considers an even more troublesome setup for strong correlated $\mathbf{A}$, and the elements of $\mathbf{A}$ are neither normal nor *i.i.d.* distributed. Fixing $K = 12$, $SNR = 30$ dB, the projection matrix is constructed by $\mathbf{A} = (1/N)\mathbf{PQ}$. The elements of $\mathbf{P}$ and $\mathbf{Q}$ *i.i.d.* obey Gaussian distributions $p_{mr} \sim N(p; 0, 1)$ and $q_{rn} \sim N(q; 0, 1)$, where $\mathbf{P} \in \mathbb{R}^{M \times R}$, $\mathbf{Q} \in \mathbb{R}^{R \times N}$ and $R \triangleq \lfloor \alpha N \rfloor$. The variation of α changes the size of $\mathbf{P}$ and $\mathbf{Q}$ and changes the rank of matrix $\mathbf{A}$. This means $\mathbf{A}$ is *low rank* for $\alpha < M/N$. Set the range of α to $[0.2,\ 0.3, \ldots, 1.0]$. Since GAMP totally diverges ($\xi \approx 10^6$) at every α, it is not visible in Figure 3c. This figure shows that RrMpGAMP-L4 keeps stable even in low rank scenario $\alpha \approx 0.3$, obviously better than AMP.

4.3. TomoSAR Imaging Application

The last experiment shows four TomoSAR imaging results. TomoSAR imaging is a spatial scatterer distribution reconstruction problem [28]. SAR imaging algorithms can be categorized into two classes: finding a dense solution and finding a sparse solution. The former is based on Nyquist sampling theory and discrete Fourier transform, e.g., the Polar Format Algorithm (PFA) [29,30]; the latter is based on compressed sampling theory and a sparse-induced algorithm, such as OMP. Tomography in the SAR imaging is especially used in inferring forest structure from several acquisitions taken at different view angles. The images are not sparse in that case. However, there are some reasons to

motivate a compressed sensing TomoSAR radar, e.g., a target in a wild dry lake bed or the scatters of a target are very sparse. We simply describe the *TomoSAR imaging* model. The two-dimensional spatial spectrum of the TomoSAR echo is given by:

$$Y(f,\theta) = \int_x \int_y g(x,y) \exp\{-2jf(x\cos\theta + y\sin\theta)\} \tag{96}$$

where $f = 2\pi/\lambda$ is the spatial frequency, (x,y) is the position of a scatterer in the target coordinate, θ is the rotation angle between the radar coordinate and the target coordinate and $g(x,y)$ is the scattering coefficient. Using summation to replace integration, the discrete variables are: spatial frequency $\mathbf{f} \in \mathbb{C}^P$, where P is frequency sampling number; rotation angle $\boldsymbol{\theta} \in \mathbb{C}^Q$, where Q is the angle sampling number; scattering coefficient vector $\mathbf{x} \in \mathbb{C}^N$, which is composed of $g(x,y)$; where $N \geq P \times Q$ is the pixel number of a TomoSAR image; received echo signal $\mathbf{y} \in \mathbb{C}^M$; and complex Gaussian noise $\boldsymbol{\epsilon} \in \mathbb{C}^M$, where $M = P \times Q$. The linear system form of Equation (96) is the same as Equation (1). Define $F(p,q,n) \triangleq \exp\{-2jf_p(x_n\cos\theta_q + y_n\sin\theta_q)\}$, and then:

$$\mathbf{A} = \begin{pmatrix} F(1,1,1) & F(1,1,2) & \cdots & F(1,1,N) \\ \cdots & \cdots & \cdots & \cdots \\ F(P,1,1) & F(P,1,2) & \cdots & F(P,1,N) \\ \cdots & \cdots & \cdots & \cdots \\ F(1,Q,1) & F(1,Q,2) & \cdots & F(1,Q,N) \\ \cdots & \cdots & \cdots & \cdots \\ F(P,Q,1) & F(P,Q,2) & \cdots & F(P,Q,N) \end{pmatrix}$$

Because the scatterers are very sparse, it is feasible to randomly draw some rows from $\mathbf{A}$ to decrease P and Q, and to transform the TomoSAR imaging into a CS problem.

In this experiment, the TomoSAR echo data comes from a real-world crawler crane. $P = 101$ frequency samples are spaced drawn from the $1(\text{GHz})$ bandwidth centered at carrier frequency $f_c = 9(\text{GHz})$; $Q = 101$ angle samples are spaced drawn from $87.5^\circ \sim 92.5^\circ$ centered at 90° azimuth. The total pixels number is $N = P \times Q$. We draw $M = \lfloor 0.5N \rfloor$ rows from $\mathbf{A}$ randomly. Since the measurements number M is usually chosen to be $O(K\log N)$, K can be estimated roughly as $\tau M/\log N$; we set $K = 50$. Note that the elements of $\mathbf{A}$ and $\mathbf{y}$ are complex, so the prior of non-zero elements in $\mathbf{x}$ should be complex, as well.

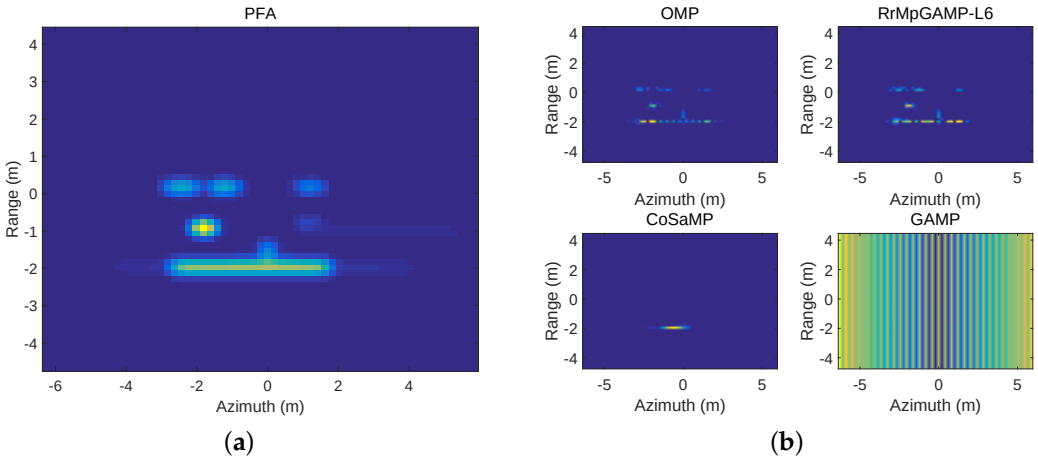

Figure 4. TomoSAR imaging results comparison using four Compressed Sensing (CS) algorithms and Polar Format Algorithm (PFA) algorithm as a reference. (**a**) PFA reconstructed image; (**b**) CS reconstructed images.

Use the PFA imaging result in Figure 4a as the reference, and compare OMP, CoSaMP and GAMP to RrMpGAMP-L6. Figure 4b shows that CoSaMP and GAMP cannot recover the TomoSAR image;

OMP and RrMpGAMP-L6 work well, but RrMpGAMP-L6 performs a little better than OMP; see the reconstructed bottom margin. RrMpGAMP-L6 recovers a more continuous segment than OMP.

5. Conclusions

While the AMP/GAMP algorithm has been shown to be a very good approach for sparse signal recovery, it is also sensitive to problems that deviate from its assumption. In this paper, we propose the stable RrMpGAMP algorithm, which matches GAMP's accuracy, performs better than AMP and GAMP, and remains robust to various projection matrices, with a small memory footprint. We use the replica method to analyze the FsGAMP algorithm, which is embedded in the RrMpGAMP algorithm, and find that enlarging the variance range of the elements of the sensing matrix does not break the convergence property of FsGAMP. We also obtain the convergence conditions of FsGAMP. Experiments confirm that our proposed algorithm performs very well in simulation and practical problems for which AMP and GAMP cannot be applied. In all cases, RrMpGAMP provides better performance as compared to BP, OMP and CoSaMP. Exact analysis of the RrMP algorithm remains an open problem for future work.

Acknowledgments: This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grants 61401501 and 61401069.

Author Contributions: Qun Wan conceived of the TomoSAR imaging experiment and provided the original data. Xunchao Cong and Yongjie Luo performed this experiment. Guan Gui designed the first two experiments and analyzed the data with Yongjie Luo. Qun Wan suggested the idea of combine RrMP and GAMP algorithms. Yongjie Luo raised the idea of RrMP. Yongjie Luo and Guan Gui raised the idea of FsGAMP, and analyzed the computational complexity; Yongjie Luo completed the replica computation, and designed the three non-zero mean sensing matrix experiments. Yongjie Luo and Guan Gui wrote the paper, Xunchao Cong and Qun Wan checked the manuscript and contributed to the rearrangement of the materials. All of the authors read and approved the final manuscript.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix

We derive Sum-Product FsGAMP briefly. The message from the factor node to the variable node has been defined as Equation (11), and the message from the variable node to the factor node has been defined as Equation (12). For the simplicity of description, we rewrite the symbols as follows:

$$\mathbf{\Delta}_{m\to k}(t,x_k) \triangleq \log\Big[\int_{\{x_j\}_{j\neq k}} p_{\mathsf{y}_m|\mathsf{z}_m}\big(y_m\big|a_{mk}x_k+\textstyle\sum_{j\neq k}^{|U|}a_{mj}x_j\big)\times\prod_{j\neq k}^{|U|}\exp\big(\mathbf{\Delta}_{m\leftarrow j}(t,x_j)\big)\Big]+\mathsf{c} \tag{A1}$$

$$\mathbf{\Delta}_{m\leftarrow k}(t+1,x_k) \triangleq \log p_{\mathsf{x}_k}(x_k)+\textstyle\sum_{i\neq m}^{M}\mathbf{\Delta}_{i\to k}(t,x_k)+\mathsf{c} \tag{A2}$$

where $m := \mathsf{f}_m$, $k := \mathsf{x}_{U_k}$, $x_k := x_{U_k}$, $x_j := x_{U_j}$, $a_{mk} := a_{mU_k}$, $a_{mj} := a_{mU_j}$, $j := \mathsf{x}_{U_j}$, $p_{\mathsf{x}_k} := p_{\mathsf{x}_{U_k}}$, $i := \mathsf{f}_i$, $T := |U|$. Furthermore, we omit the superscripts $|U|$ and M in the follow up of the derivation.

Appendix A1. Message from the Factor Node to the Variable Node

Define:

$$z_m \triangleq a_{mk}x_k+\textstyle\sum_{j\neq k}a_{mj}x_j \tag{A3}$$

For the GAMP algorithm, the Central-Limit Theorem (CLT) makes z_m conditionally Gaussian for large N, *i.e.*,

$$z_m|x_k \sim \mathcal{N}(z_m;a_{mk}x_k+\hat{p}_{mk}(t),\nu^p_{mk}(t)) \tag{A4}$$

where:

$$\hat{p}_{mk}(t) \triangleq \sum_{j \neq k} a_{mj} \hat{x}_{mj}(t) \tag{A5}$$

$$v^p_{mk}(t) \triangleq \sum_{j \neq k} |a_{mj}|^2 v^x_{mj}(t) \tag{A6}$$

In contrast to GAMP, FsGAMP not only works for a large N, but also works for a small N, because we embed FsGAMP into the RrMP algorithm, and the RrMP's pursuit is sequential. Although CLT is not be satisfied, the RrMP's pursuit means that all of the amplitudes that are supported on the pursued indexes are non-zero. As long as we assume that these amplitudes obey a Gaussian distribution independently, the linear weighted sum of these amplitudes also obeys a Gaussian distribution. Therefore, Equations (A4)–(A6) are still available. Now, we have:

$$\mathbf{\Delta}_{m \to k}(t, x_k) \approx \log \left[\int_{z_m} p_{\mathsf{y}_m|\mathsf{z}_m}(y_m|z_m) \mathcal{N}\left(z_m; a_{mk} x_k + \hat{p}_{mk}(t), v^p_{mk}(t)\right) \right] + \mathsf{c} \tag{A7}$$

and we define:

$$H(a_{mk} x_k + \hat{p}_{mk}(t), y_m, v^p_{mk}(t)) \triangleq \log \left[\int_{z_m} p_{\mathsf{y}_m|\mathsf{z}_m}(y_m|z_m) \mathcal{N}\left(z_m; a_{mk} x_k + \hat{p}_{mk}(t), v^p_{mk}(t)\right) \right] \tag{A8}$$

In order to eliminate the dependence on the subscript m of the $H(\cdot, \cdot, \cdot)$ function, define:

$$\hat{p}_m(t) \triangleq \sum_{j} a_{mj} \hat{x}_{mj}(t) \tag{A9}$$

$$v^p_m(t) \triangleq \sum_{j} |a_{mj}|^2 v^x_{mj}(t) \tag{A10}$$

and then plug them into Equation (A7):

$$\mathbf{\Delta}_{m \to k}(t, x_k) \approx H(a_{mk}(x_k - \hat{x}_{mk}(t)) + \hat{p}_m(t), y_m, v^p_{mk}(t)) + \mathsf{c} \tag{A11}$$

$$= H(a_{mk}(x_k - \hat{x}_k(t)) + \hat{p}_m(t) + O(1/T), y_m, v^p_m(t) + O(1/T)) + \mathsf{c} \tag{A12}$$

where we assume x_k is $O(1)$, a_{mk} is $O(1/\sqrt{T})$ and *i.i.d.* drawn from a zero-mean distribution, such that z_m is $O(1)$. Based on these assumptions, other scales are defined for the sequential derivation. We list all the scales in Table A1.

Table A1. FsGAMP variable scales.

Variable	Order	Variable	Order	Variable	Order
$\hat{x}_{mk}(t)$	$O(1)$	$v^x_{mk}(t)$	$O(1)$	$\hat{x}_{mk}(t) - \hat{x}_k(t)$	$O(1/\sqrt{T})$
$\hat{x}_k(t)$	$O(1)$	$v^x_k(t)$	$O(1)$	$v^r_{mk}(t) - v^r_k(t)$	$O(1/T)$
$\hat{r}_{mk}(t)$	$O(1)$	$v^r_{mk}(t)$	$O(1)$	$\hat{p}_{mk}(t) - \hat{p}_m(t)$	$O(1/\sqrt{T})$
$\hat{r}_k(t)$	$O(1)$	$v^r_k(t)$	$O(1)$	$v^p_{mk}(t) - v^p_m(t)$	$O(1/T)$
$\hat{z}_m(t)$	$O(1)$	$v^z_m(t)$	$O(1)$	$a_{mk}(\hat{x}_k(t) - \hat{x}_{mk}(t))$	$O(1/T)$
$\hat{p}_m(t)$	$O(1)$	$v^p_m(t)$	$O(1)$		
$\hat{s}_m(t)$	$O(1)$	$v^s_m(t)$	$O(1)$		

It is worth noting that the assumption of x_k and a_{mk} can be adjusted to other orders, leading to other derivation forms.

Applying the second order Taylor-series expansion to Equation (A12), dropping higher order terms and dropping $O(1/T)$ perturbations of the $H'(\cdot,\cdot,\cdot)$ and $H''(\cdot,\cdot,\cdot)$, since $\hat{p}_m(t)$ and $\nu_m^p(t)$ are $O(1)$, we get:

$$\begin{aligned}\boldsymbol{\Delta}_{m\to k}(t,x_k) \approx\ & H(\hat{p}_m, y_m, \nu_m^p(t)) \\ & + a_{mk}(x_k - \hat{x}_k(t))H'(\hat{p}_m(t), y_m, \nu_m^p(t)) \\ & + \frac{1}{2}|a_{mk}|^2(x_k - \hat{x}_k(t))^2 H''(\hat{p}_m(t), y_m, \nu_m^p(t)) + \mathsf{c}\end{aligned} \tag{A13}$$

where $H'(\cdot,\cdot,\cdot)$ and $H''(\cdot,\cdot,\cdot)$ are the first derivative and second derivative of function $H(\cdot,\cdot,\cdot)$ with respect to the first argument.

Define:

$$\hat{s}_m(t) \triangleq H'(\hat{p}_m(t), y_m, \nu_m^p(t)) \tag{A14}$$

$$\nu_m^s(t) \triangleq H''(\hat{p}_m(t), y_m, \nu_m^p(t)) \tag{A15}$$

and notice that $H(\hat{p}_m(t), y_m, \nu_m^p(t))$ is a constant w.r.t. variable x_k, such that it can be dropped; then, Equation (A13) can be rewritten to:

$$\boldsymbol{\Delta}_{m\to k}(t,x_k) \approx \left[\hat{s}_m(t)a_{mk} + \nu_m^s(t)|a_{mk}|^2\hat{x}_k(t)\right]x_k - \frac{1}{2}\nu_m^s(t)|a_{mk}|^2 x_k^2 + \mathsf{c} \tag{A16}$$

It can be seen that Equation (A16) is a quadratic form, which means the PDF function $\frac{1}{Z}\exp(\boldsymbol{\Delta}_{m\to k}(t,x_k))$ can be approximated as a Gaussian distribution.

Now, simplify Equations (A14) and (A15) further. Denote:

$$H(\hat{p}, y, \nu^p) \triangleq \log\Big[\int_z p_{\mathsf{y}|\mathsf{z}}(y|z)\mathcal{N}(z;\hat{p},\nu^p)\Big] \tag{A17}$$

and then:

$$\begin{aligned}H'(\hat{p}, y, \nu^p) &= \frac{\partial}{\partial\hat{p}}\log\int_z p_{\mathsf{y}|\mathsf{z}}(y|z)\frac{1}{\sqrt{2\pi\nu^p}}e^{-\frac{(z-\hat{p})^2}{2\nu^p}} \\ &= \frac{\partial}{\partial\hat{p}}\Big\{-\frac{\hat{p}^2}{2\nu^p} + \log\int_z e^{(\log p_{\mathsf{y}|\mathsf{z}}(y|z)) - \frac{z^2}{2\nu^p} + \frac{\hat{p}z}{\nu^p}}\Big\} \\ &= -\frac{\hat{p}}{\nu^p} + \int_z \frac{z}{\nu^p}\frac{e^{(\log p_{\mathsf{y}|\mathsf{z}}(y|z)) - \frac{z^2}{2\nu^p} + \frac{\hat{p}z}{\nu^p}}}{Z(\hat{p})}\frac{1}{\nu^p} \\ &= -\frac{\hat{p}}{\nu^p} + \frac{1}{\nu^p}\int_z z\frac{p_{\mathsf{y}|\mathsf{z}}(y|z)\mathcal{N}(z;\hat{p},\nu^p)}{\int_u p_{\mathsf{y}|\mathsf{z}}(y|u)\mathcal{N}(u;\hat{p},\nu^p)} \qquad \text{(A18)} \\ &= \frac{1}{\nu^p}\left(\mathbb{E}\{z|y,\hat{p},\nu^p\} - \hat{p}\right) \qquad \text{(A19)}\end{aligned}$$

where the $\frac{p_{\mathsf{y}|\mathsf{z}}(y|z)\mathcal{N}(z;\hat{p},\nu^p)}{\int_u p_{\mathsf{y}|\mathsf{z}}(y|u)\mathcal{N}(u;\hat{p},\nu^p)}$ term in Equation (A18) is a probability density function.

In the similar way, and using the property of variance $\mathrm{var}\{\mathbf{u}\} = \mathbb{E}\{\mathbf{u}^2\} - (\mathbb{E}\{\mathbf{u}\})^2$, we obtain:

$$H''(\hat{p}, y, \nu^p) = -\frac{1}{\nu^p}\Big(1 - \frac{\mathrm{var}\{z|y,\hat{p},\nu^p\}}{\nu^p}\Big) \tag{A20}$$

Recover the time index (t) and subscript m; we denote the first two moments of intermediate random variable z_m as:

$$\hat{z}_m(t) \triangleq \mathbb{E}\{z_m | y_m, \hat{p}_m(t), v_m^p(t)\} \tag{A21}$$

$$v_m^z(t) \triangleq \mathrm{var}\{z_m | y_m, \hat{p}_m(t), v_m^p(t)\} \tag{A22}$$

They are the output function $g_{\mathrm{out}}(\cdot,\cdot)$ in Line 7 of Algorithm 2.

Appendix A2. Message from the Variable Node to the Factor Node

Plug Equation (A16) into Equation (A2); we have:

$$\begin{aligned}\boldsymbol{\Delta}_{m\leftarrow k}(t+1,x_k) &= \log p_{\mathsf{x}_k}(x_k) + \sum_{i\neq m} \boldsymbol{\Delta}_{i\to k}(t,x_k) + \mathsf{c} \\ &\approx \log p_{\mathsf{x}_k}(x_k) + \sum_{i\neq m} \left[\hat{s}_i(t)a_{ik} + v_i^s(t)|a_{ik}|^2\hat{x}_k(t)\right]x_k - \frac{1}{2}v_i^s(t)|a_{ik}|^2x_k^2 + \mathsf{c} \\ &= \log p_{\mathsf{x}_k}(x_k) - \frac{1}{2v_{mk}^r(t)}(x_k - \hat{r}_{mk}(t))^2 + \mathsf{c}\end{aligned} \tag{A23}$$

where:

$$v_{mk}^r(t) \triangleq \frac{1}{\sum_{i\neq m}|a_{ik}|^2 v_i^s(t)} \tag{A24}$$

$$\hat{r}_{mk}(t) \triangleq \hat{x}_k(t) + v_{mk}^r(t)\sum_{i\neq m} a_{ik}\hat{s}_i(t) \tag{A25}$$

they are $O(1)$ quantities. Observe the form of Equation (A23); it implies a PDF function $\frac{p_{\mathsf{x}}(x)\mathcal{N}(x;\hat{r}_{mk}(t),v_{mk}^r(t))}{\int_u p_{\mathsf{x}}(u)\mathcal{N}(u;\hat{r}_{mk}(t),v_{mk}^r(t))}$. We can write the first moment of this PDF function:

$$\hat{x}_{mk}(t+1) \triangleq \mathbb{E}\{x_k|\hat{r}_{mk}(t),v_{mk}^r(t)\} \approx \int_x x\frac{p_{\mathsf{x}}(x)\mathcal{N}(x;\hat{r}_{mk}(t),v_{mk}^r(t))}{\int_u p_{\mathsf{x}}(u)\mathcal{N}(u;\hat{r}_{mk}(t),v_{mk}^r(t))} \tag{A26}$$

This form is very similar to the integration in Equation (A18). Therefore, if we define another function:

$$G(\hat{r},v^r) \triangleq \log\int_x p_{\mathsf{x}}(x)\mathcal{N}(x;\hat{r},v^r) \tag{A27}$$

it is very similar to Equation (A8), and we can obtain:

$$G'(\hat{r},v^r) = \frac{1}{v^r}\left(\mathbb{E}\{x|\hat{r},v^r\} - \hat{r}\right) \tag{A28}$$

$$G''(\hat{r},v^r) = \frac{1}{v^r}\left(\frac{\mathrm{var}\{x|\hat{r},v^r\}}{v^r} - 1\right) \tag{A29}$$

from the same derivation as Appendix A1.

Denote the form of Equation (A26) as:

$$g_{\mathrm{in}}(\hat{r},v^r) \triangleq \int_x x\frac{p_{\mathsf{x}}(x)\mathcal{N}(x;\hat{r},v^r)}{\int_u p_{\mathsf{x}}(u)\mathcal{N}(u;\hat{r},v^r)} \tag{A30}$$

then, from Equation (A28), we have:

$$g_{\mathrm{in}}(\hat{r},v^r) = \hat{r} + v^r G'(\hat{r},v^r) \tag{A31}$$

For Equation (A31), differentiate w.r.t. $\hat{r}$ and plug in Equation (A29); we have:

$$g'_{\text{in}}(\hat{r}, \nu^r) = 1 + \nu^r G''(\hat{r}, \nu^r) = \frac{1}{\nu^r} \text{var}\{x|\hat{r}, \nu^r\} \tag{A32}$$

Now, we get:

$$\text{var}\{x|\hat{r}, \nu^r\} = \nu^r g'_{\text{in}}(\hat{r}, \nu^r) \tag{A33}$$

Recover the time index (t) and subscript mk, and rewrite Equation (A26) and (A33) as:

$$\mathbb{E}\{x_k|\hat{r}_{mk}(t), \nu^r_{mk}(t)\} = \hat{x}_{mk}(t+1) \approx g_{\text{in}}(\hat{r}_{mk}(t), \nu^r_{mk}(t)) \tag{A34}$$

$$\text{var}\{x_k|\hat{r}_{mk}(t), \nu^r_{mk}(t)\} = \nu^x_{mk}(t+1) \approx \nu^r_{mk}(t) g'_{\text{in}}(\hat{r}_{mk}(t), \nu^r_{mk}(t)) \tag{A35}$$

In order to eliminate the dependence on the subscript m of Equation (A34), we define:

$$\nu^r_k(t) \triangleq \frac{1}{\sum_i |a_{ik}|^2 \nu^s_i(t)} \tag{A36}$$

$$\hat{r}_k(t) \triangleq \hat{x}_k(t) + \nu^r_k(t) \sum_i a_{ik} \hat{s}_i(t) \tag{A37}$$

and assume $\nu^r_{mk}(t) \approx \nu^r_k(t)$; using the first order Taylor-series expansion, we have:

$$\begin{aligned} \hat{x}_{mk}(t+1) &\approx g_{\text{in}}(\hat{r}_{mk}(t), \nu^r_{mk}(t)) \\ &\approx g_{\text{in}}(\hat{r}_k(t) - a_{mk}\hat{s}_m(t)\nu^r_k(t), \nu^r_k(t)) \\ &\approx g_{\text{in}}(\hat{r}_k(t), \nu^r_k(t)) - a_{mk}\hat{s}_m(t)\nu^r_k(t) g'_{\text{in}}(\hat{r}_k(t), \nu^r_k(t)) \end{aligned} \tag{A38}$$

For simplicity, define:

$$\hat{x}_k(t+1) \triangleq g_{\text{in}}(\hat{r}_k(t), \nu^r_k(t)) \tag{A39}$$

$$\nu^x_k(t+1) \triangleq \nu^r_k(t) g'_{\text{in}}(\hat{r}_k(t), \nu^r_k(t)) \tag{A40}$$

then, Equation (A26) is $\hat{x}_{mk}(t+1) \approx \hat{x}_k(t+1) - a_{mk}\hat{s}_m(t)\nu^x_k(t+1)$.

Finally, $\hat{p}_m(t)$ in Equation (A9) and $\nu^p_m(t)$ in Equation (A10) can be rewritten as:

$$\hat{p}_m(t+1) = \sum_j a_{mj}\hat{x}_{mj}(t+1) \approx \sum_j a_{mj}\hat{x}_j(t+1) - \hat{s}_m(t) \sum_j |a_{mj}|^2 \nu^x_j(t+1) \tag{A41}$$

and define $\nu^p_j(t+1) \cong \sum_j |a_{mj}|^2 \nu^x_j(t+1)$. We close the loop.

References

1. Wu, Z.; Peng, S. Proportionate Minimum Error Entropy Algorithm for Sparse System Identification. *Entropy* **2015**, *17*, 5995–6006.
2. Ma, W.; Qu, H. Maximum correntropy criterion based sparse adaptive filtering algorithms for robust channel estimation under non-Gaussian environments. *J. Frankl. Inst.* **2015**, *352*, 2708–2727.
3. Ma, W.; Chen, B. Sparse least logarithmic absolute difference algorithm with correntropy induced metric penality. *Circuit Syst. Signal Process.* **2015**, *35*, 1077–1089.
4. Chen, B.; Zhao, S. Online efficient learning with quantized KLMS and L1 regularization. In Proceedings of the International Joint Conference on Neural Networks, (IJCNN 2012), Brisbane, Australia, 10–15 June 2012; pp. 1–6.
5. Wang, W.X.; Yang, R.; Lai, Y.C.; Kovanis, V.; Grebogi, C. Predicting Catastrophes in Nonlinear Dynamical Systems by Compressive Sensing. *Phys. Rev. Lett.* **2011**, *106*, doi:10.1103/PhysRevLett.106.154101.

6. Brunton, S.L.; Tu, J.H.; Bright, I.; Kutz, J.N. Compressive Sensing and Low-Rank Libraries for Classification of Bifurcation Regimes in Nonlinear Dynamical Systems. *SIAM J. Appl. Dyn. Syst.* **2014**, *13*, 1716–1732.
7. Donoho, D.; Maleki, A.; Motanari, A. Message Passing algorithms for compressed sensing. *Proc. Natl. Acad. Sci. USA* **2009**, *106*, 18914–18919.
8. Donoho, D.; Maleki, A.; Montanari, A. Message passing algorithms for compressed sensing: I. motivation and construction. In Proceedings of the 2010 IEEE Information Theory Workshop (ITW), Cairo, Egypt, 6–8 January 2010; pp. 1–5.
9. Rangan, S. Generalized approximate message passing for estimation with random linear mixing. In Proceedings of the 2011 IEEE International Symposium on Information Theory Proceedings (ISIT), St. Petersburg, Russia, 31 July–5 August 2011; pp. 2168–2172.
10. Pati, Y.; Rezaiifar, R.; Krishnaprasad, P. Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition. In Proceedings of the 1993 Conference Record of the Twenty-Seventh Asilomar Conference on Signals, Systems and Computers, Pacific Grove, CA, USA, 1–3 November 1993; Volume 1, pp. 40–44.
11. Needell, D.; Tropp, J.A. CoSaMP: Iterative signal recovery from incomplete and inaccurate samples. *Appl. Comput. Harmonic Anal.* **2009**, *26*, 301–321.
12. Needell, D.; Vershynin, R. Uniform uncertainty principle and signal recovery via regularized orthogonal matching pursuit. *Found. Comput. Math.* **2009**, *9*, 317–334.
13. Needell, D.; Vershynin, R. Signal recovery from incomplete and inaccurate measurements via regularized orthogonal matching pursuit. *IEEE J. Sel. Top. Signal Process.* **2010**, *4*, 310–316.
14. Wainwright, M.J.; Jordan, M.I. Graphical models, exponential families, and variational inference. *Found. Trends Mach. Learn.* **2008**, *1*, 1–305.
15. Rangan, S.; Schniter, P.; Fletcher, A. On the convergence of approximate message passing with arbitrary matrices. In Proceedings of the 2014 IEEE International Symposium on Information Theory (ISIT), Honolulu, HI, USA, 29 June–4 July 2014; pp. 236–240.
16. Golub, G.H.; van Loan, C.F. *Matrix Computations*, 3rd ed.; The Johns Hopkins University Press: Baltimore, MD, USA, 2012.
17. Ḿezard, M.; Parisi, G.; Virasoro, M.A. Spin-Glass Theory and Beyond. In *World Scientific Lecture Notes in Physics*; World Scientific: Singapore, Singapore, 1987; Volume 9.
18. Ḿezard, M.; Montanari, A. *Information, Physics, and Computation*; Oxford University Press: Oxford, UK, 2009.
19. Rangan, S.; Goyal, V.; Fletcher, A.K. Asymptotic analysis of map estimation via the replica method and compressed sensing. *IEEE Trans. Inform. Theory* **2012**, *58*, 1902–1923.
20. Kabashima, Y.; Wadayama, T.; Tanaka, T. Statistical mechanical analysis of a typical reconstruction limit of compressed sensing. In Proceedings of the 2010 IEEE International Symposium on Information Theory (ISIT), Austin, TX, USA, 13–18 June 2010; pp. 1533–1537.
21. Ganguli, S.; Sompolinsky, H. Statistical mechanics of compressed sensing. *Phys. Rev. Lett.* **2010**, *104*, doi:10.1103/PhysRevLett.104.188701.
22. Solomon, A.T.; Bruhtesfa, E.G. Compressed Sensing Performance Analysis via Replica Method Using Bayesian framework. In Proceedings of the 17th UKSIM-AMSS International Conference on Modelling and Simulationm, Cambridge, UK, 25–27 March 2015; pp. 281–289.
23. Krzakala, F.; Mézard, M.; Sausset, F.; Sun, Y.; Zdeborová, L. Probabilistic reconstruction in compressed sensing: Algorithms, phase diagrams, and threshold achieving matrices. *J. Stat. Mech. Theory Exp.* **2012**, *2012*, doi:10.1088/1742-5468/2012/08/P08009.
24. Needell, D. Available online: http://www.cmc.edu/pages/faculty/DNeedell (accessed on 23 May 2016).
25. ℓ_1-MAGIC. Available online: http://users.ece.gatech.edu/ justin/l1magic (accessed on 23 May 2016).
26. Kamilov, U.S. Available online: http://www.ukamilov.com (accessed on 23 May 2016).
27. GAMP. Available online: http://gampmatlab.wikia.com/wiki/Generalized_Approximate_Message_Passing (accessed on 23 May 2016).
28. Zhu, X.X.; Bamler, R. Superresolving SAR Tomography for Multidimensional Imaging of Urban Areas: Compressive Sensing-Based TomoSAR inversion. *IEEE Signal Process. Mag.* **2014**, *31*, 51–58.

29. Carrara, W.G.; Goodman, R.S.; Majewski, R.M. *Spotlight Synthetic Aperture Radar: Signal Processing Algorithms*; Artech House: Norwood, MA, USA, 1995.
30. Cong, X.; Liu, J.; Long, K.; Liu, Y.; Zhu, R.; Wan, Q. Millimeter-wave spotlight circular synthetic aperture radar (SCSAR) imaging for Foreign Object Debris on airport runway. In Proceedings of the 12th International Conference on Signal Processing (ICSP), HangZhou, China, 26–30 October 2014; pp. 1968–1972.