SPARSE SIGNAL RECOVERY USING GENERALIZED APPROXIMATE MESSAGE PASSING WITH BUILT-IN PARAMETER ESTIMATION

Shuai Huang and Trac D. Tran

Department of Electrical and Computer Engineering
Johns Hopkins University, Baltimore, MD, USA
{shuaihuang, trac}@jhu.edu

ABSTRACT

The generalized approximate message passing (GAMP) algorithm under the Bayesian setting shows significant advantages in recovering under-sampled sparse signals from corrupted observations. Compared to conventional convex optimization methods, it has a much lower complexity and is computationally tractable. Under the GAMP framework, the sparse signal and the observation are viewed to be generated according to some pre-specified probability distributions in the input and output channels. However, the parameters of the distributions are usually unknown in practice and need to be decided. In this paper, we propose an extended GAMP algorithm with built-in parameter estimation (PE-GAMP). Specifically, PE-GAMP treats the parameters as unknown random variables with simple priors and jointly estimates them with the sparse signals along the recovery process. Sparse signal recovery experiments confirm PE-GAMP's convergence behavior and show that its performance matches the oracle GAMP algorithm that has the knowledge of the true parameter values.

Index Terms— Sparse signal recovery, approximate message passing, parameter estimation, belief propagation, compressive sensing.

1. INTRODUCTION

Sparse signal recovery (SSR) plays the critical role in the Compressive Sensing (CS) framework [1–4]. Besides signal recovery, it also lays the foundation for applications such as dictionary learning [5], sparse representation-based classification [6], etc. Specifically, SSR tries to recover the sparse signal $\boldsymbol{x} \in \mathbb{R}^N$ given a $M \times N$ sensing matrix $\boldsymbol{A}$ and a measurement vector $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{w} \in \mathbb{R}^M$, where $M < N$ and $\boldsymbol{w} \in \mathbb{R}^M$ is the unknown noise introduced in this process. Although the problem itself is ill-posed, perfect recovery is still possible provided that $\boldsymbol{x}$ is sufficiently sparse and $\boldsymbol{A}$ is incoherent enough [1]. Lasso [7], a.k.a l_1-minimization, is one of most popular approaches proposed to solve this problem:

$$\arg\min_{\boldsymbol{x}} \quad \|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2 + \gamma\|\boldsymbol{x}\|_1 \,, \tag{1}$$

where $\|\boldsymbol{y} - \boldsymbol{A}\boldsymbol{x}\|_2^2$ is the data-fidelity term, $\|\boldsymbol{x}\|_1$ is the sparsity-promoting term, and γ balances the trade-off between them.

From a probabilistic view, Lasso is equivalent to a maximum likelihood (ML) estimation of the signal $\boldsymbol{x}$ under the assumption that the entries of $\boldsymbol{x}$ are i.i.d. distributed following the Laplace distribution $p(x_j) \propto \exp(-\lambda|x_j|)$, and those of $\boldsymbol{w}$ are i.i.d. distributed following the Gaussian distribution $p(w_i) \propto \exp\left(-w_i^2/2\theta\right)$. Let $\boldsymbol{z} = \boldsymbol{A}\boldsymbol{x}$, we have $p(y_i|\boldsymbol{x}) \propto \exp\left(-(y_i - z_i)^2/2\theta\right)$. The ML estimation is then $\arg\max_{\boldsymbol{x}} p(\boldsymbol{x}, \boldsymbol{y})$, which is the same as (1).

Under the Bayesian setting it is possible to design efficient iterative algorithms to compute either the maximum a posterior (MAP) or minimum mean square error (MMSE) estimate of the signal $\boldsymbol{x}$. Most notable among them are the "message-passing" based algorithms [8–11]. They perform probabilistic inferences on the corresponding factor graph using Gaussian and/or quadratic approximations of loopy belief propagation (loopy BP), hence the name message passing [12]. Loopy BP has two variants: *sum-product* message passing for the MMSE estimate of $\boldsymbol{x}$ and *max-sum* message passing for the MAP estimate of $\boldsymbol{x}$. *Approximate* message passing (AMP) is proposed based on a quadratic approximation of *max-sum* message passing [8–10]. It has low complexity and can be used to find solutions of Lasso accurately. In fact, AMP is able to match the performance of theoretical Lasso in noiseless signal recovery experiments [8]. Its empirical convergence is guaranteed in the large system limit for $\boldsymbol{A}$ with i.i.d Gaussian entries [10].

Various methods based on the above AMP framework has been proposed to perform sparse signal recovery [11, 13, 14]. [13, 14] treat each AMP iteration as a signal denoising process and introduces the denoiser into the AMP algorithm. In [11], a generalized version of the AMP algorithm (GAMP) is proposed to work with essentially arbitrary input and output channel distributions. It can approximate both the *sum-product* and *max-sum* message passings using only scalar estimations and linear transforms. The parameters $\{\boldsymbol{\lambda}, \boldsymbol{\theta}\}$ in the input and output channels are usually unknown, and need to be decided for the AMP/GAMP algorithm. Various methods have been proposed to estimate the parameters for the GAMP algorithm. For example, the Expectation-Maximization (EM) algorithm [15] can be used to perform parameter estimation [16–19].

1.1. Main Contributions

In this paper, we propose an extension to the GAMP framework by treating the parameters $\boldsymbol{\lambda}, \boldsymbol{\theta}$ as unknown random variables with simple prior distributions and estimating them jointly with the signal $\boldsymbol{x}$ under the same framework: PE-GAMP. This enables us to *compute the following posterior distributions of the parameters directly from loopy belief propagation*.

- ***Sum-product*** **message passing:** The "marginal" posterior distributions $\{p(\boldsymbol{\lambda}|\boldsymbol{y}), p(\boldsymbol{\theta}|\boldsymbol{y})\}$ can be obtained.
- ***Max-sum*** **message passing:** The "joint" posterior distribution $\{p(\boldsymbol{\lambda}, \tilde{\boldsymbol{x}}|\boldsymbol{y}), p(\boldsymbol{\theta}, \tilde{\boldsymbol{x}}|\boldsymbol{y})\}$ can be obtained, where $\tilde{\boldsymbol{x}}$ are the values that maximizes the joint posterior distribution.

This work is partially supported by National Science Foundation under Grants ECCS-1443936 and CCF-1422995.

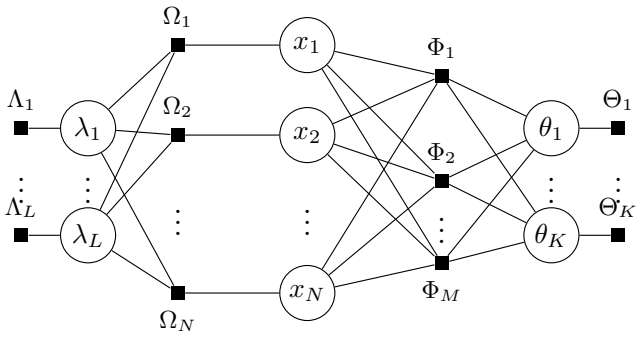


Fig. 1: The factor graph for the proposed PE-GAMP. "■" represents the factor node, and "◯" represents the variable node. $\boldsymbol{\lambda} = \{\lambda_1, \cdots, \lambda_L\}$ and $\boldsymbol{\theta} = \{\theta_1, \cdots, \theta_K\}$ are the parameters. $\boldsymbol{x} = [x_1, \cdots, x_N]^{\mathrm{T}}$ is the sparse signal.

For the *sum-product* message passing, if the input and output channel distributions $p(\boldsymbol{x}|\boldsymbol{\lambda}), p(\boldsymbol{y}|\boldsymbol{z}, \boldsymbol{\theta})$ are simple enough so that the integration involved in the message passing process can be computed, the parameter estimation will be automatically taken care of and no special treatments are needed. However, in practice the channel distributions are usually complicated, and the integration usually doesn't have closed-form solutions. In this case, we can compute the MMSE or MAP estimates of the parameters $\boldsymbol{\lambda}, \boldsymbol{\theta}$ using Dirac delta approximations of the posterior distributions and use them to simplify the message passing process. For the *max-sum* message passing, the maximization problem involving multiple variables can be efficiently solved by using the approximate maximizing parameters.

Previous EM based parameter estimation can only be used with *sum-product* message passing, since it relies on the marginal distribution $p(\boldsymbol{x}|\boldsymbol{y}, \hat{\boldsymbol{\lambda}}^{(t)}, \hat{\boldsymbol{\theta}}^{(t)})$ to compute the expectation of the log likelihood $\log p(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\lambda}, \boldsymbol{\theta})$. While our proposed PE-GAMP could be applied to both *sum-product* and *max-sum* message passings, which give MMSE and MAP estimations of the signal respectively. Using the popular input and output channel distributions adopted by the sparse signal recovery model, we can also show that the recovery performance of the proposed PE-GAMP is able to match the oracle-GAMP that knows the true parameter values. A longer version of this paper with more details and experiments is given in [20].

2. GAMP WITH BUILT-IN PARAMETER ESTIMATION

The generalized factor graph for the proposed PE-GAMP framework that treats the parameters as random variables is shown in Fig. 1. Inference tasks performed on the factor graph rely on the "messages" passed among connected nodes of the graph. Here we adopt the same notations used by [11]. Take the messages being passed between the factor node Φ_m and the variable node x_n for example, $\Delta_{\Phi_m \to x_n}$ is the message from Φ_m to x_n, and $\Delta_{\Phi_m \leftarrow x_n}$ is the message from x_n to Φ_m. Both $\Delta_{\Phi_m \to x_n}$ and $\Delta_{\Phi_m \leftarrow x_n}$ can be viewed as functions of x_n. In the following section 2.1 and 2.2, we give the messages being passed on the generalized factor graph in "log" domain for the *sum-product* message passing algorithm and the *max-sum* message passing algorithm respectively.

2.1. Sum-product Message Passing

Sum-product message passing is used to compute the marginal distributions of the random variables in the graph: $p(\boldsymbol{x}|\boldsymbol{y}), p(\boldsymbol{\lambda}|\boldsymbol{y}), p(\boldsymbol{\theta}|\boldsymbol{y})$. In the following, we first present the *sum-product* message updates equations in the $(t+1)$-th iteration.

$$\Delta^{(t+1)}_{\Phi_m \to x_n} = \text{const} + \log \int_{\boldsymbol{x} \backslash x_n, \boldsymbol{\theta}} \Big[\Phi_m\left(y_m, \boldsymbol{x}, \boldsymbol{\theta}\right) \times \exp\Big(\textstyle\sum_{j \neq n} \Delta^{(t)}_{\Phi_m \leftarrow x_j} + \sum_v \Delta^{(t)}_{\Phi_m \leftarrow \theta_v} \Big) \Big] \tag{2a}$$

$$\Delta^{(t+1)}_{\Phi_m \leftarrow x_n} = \text{const} + \Delta^{(t+1)}_{\Omega_n \to x_n} + \textstyle\sum_{i \neq m} \Delta^{(t+1)}_{\Phi_i \to x_n} \tag{2b}$$

$$\Delta^{(t+1)}_{\Omega_n \to x_n} = \text{const} + \log \int_{\boldsymbol{\lambda}} \Omega_n(x_n, \boldsymbol{\lambda}) \cdot \exp\Big(\textstyle\sum_u \Delta^{(t)}_{\Omega_n \leftarrow \lambda_u} \Big) \tag{2c}$$

$$\Delta^{(t+1)}_{\Omega_n \leftarrow x_n} = \text{const} + \textstyle\sum_i \Delta^{(t+1)}_{\Phi_i \to x_n}\,, \tag{2d}$$

where $\boldsymbol{x} \backslash x_n$ denotes the sequence obtained by removing x_n from $\boldsymbol{x}$, $\Phi_m(y_m, \boldsymbol{x}, \boldsymbol{\theta}) = p(y_m|\boldsymbol{x}, \boldsymbol{\theta})$ and $\Omega_n(x_n, \boldsymbol{\lambda}) = p(x_n|\boldsymbol{\lambda})$. Similarly, we can write the message updates involving the variable nodes λ_l, θ_k as follows:

$$\Delta^{(t+1)}_{\Omega_n \to \lambda_l} = \text{const} + \log \int_{x_n, \boldsymbol{\lambda} \backslash \lambda_l} \Big[\Omega_n(x_n, \boldsymbol{\lambda}) \times \exp\Big(\Delta^{(t+1)}_{\Omega_n \leftarrow x_n} + \textstyle\sum_{u \neq l} \Delta^{(t)}_{\Omega_n \leftarrow \lambda_u} \Big) \Big] \tag{3a}$$

$$\Delta^{(t+1)}_{\Omega_n \leftarrow \lambda_l} = \text{const} + \textstyle\sum_{j \neq n} \Delta^{(t+1)}_{\Omega_j \to \lambda_l} + \log p(\lambda_l) \tag{3b}$$

$$\Delta^{(t+1)}_{\Phi_m \to \theta_k} = \text{const} + \log \int_{\boldsymbol{\theta} \backslash \theta_k, \boldsymbol{x}} \Big[\Phi_m\left(y_m, \boldsymbol{x}, \boldsymbol{\theta}\right) \times \exp\Big(\textstyle\sum_j \Delta^{(t)}_{\Phi_m \leftarrow x_j} + \sum_{v \neq k} \Delta^{(t)}_{\Phi_m \leftarrow \theta_v} \Big) \Big] \tag{3c}$$

$$\Delta^{(t+1)}_{\Phi_m \leftarrow \theta_k} = \text{const} + \textstyle\sum_{i \neq m} \Delta^{(t+1)}_{\Phi_i \to \theta_k} + \log p(\theta_k)\,, \tag{3d}$$

where $p(\lambda_l), p(\theta_k)$ are the pre-specified priors of the parameters. Let $\Gamma(x_n), \Gamma(\lambda_l), \Gamma(\theta_k)$ denote the factor nodes in the neighborhood of the variable nodes x_n, λ_l, θ_k respectively, we can sum up all the messages passed on to the variable nodes:

$$\Delta^{(t+1)}_{x_n} = \Delta^{(t+1)}_{\Omega_n \to x_n} + \textstyle\sum_{\Phi_m \in \Gamma(x_n)} \Delta^{(t+1)}_{\Phi_m \to x_n} \tag{4a}$$

$$\Delta^{(t+1)}_{\lambda_l} = \log p(\lambda_l) + \textstyle\sum_{\Omega_n \in \Gamma(\lambda_l)} \Delta^{(t+1)}_{\Omega_n \to \lambda_l} \tag{4b}$$

$$\Delta^{(t+1)}_{\theta_k} = \log p(\theta_k) + \textstyle\sum_{\Phi_m \in \Gamma(\theta_k)} \Delta^{(t+1)}_{\Phi_m \to \theta_k} \tag{4c}$$

We then have the marginal distributions: $p(x_n|\boldsymbol{y}) \propto \exp(\Delta^{(t+1)}_{x_n})$, $p(\lambda_l|\boldsymbol{y}) \propto \exp(\Delta^{(t+1)}_{\lambda_l})$, $p(\theta_k|\boldsymbol{y}) \propto \exp(\Delta^{(t+1)}_{\theta_k})$. The MMSE estimate of $\boldsymbol{x}$ is as follows:

$$\hat{x}_n = \mathbb{E}\left[x_n|\boldsymbol{y}\right] = \textstyle\int_{x_n} x_n p(x_n|\boldsymbol{y})\,. \tag{5}$$

2.2. Max-sum Message Passing

Max-sum message passing is used to compute the "joint" MAP estimates of the random variables in the graph:

$$(\hat{\boldsymbol{x}}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\theta}}) = \arg\max_{\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\theta}} p(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\theta}|\boldsymbol{y})\,. \tag{6}$$

For the *max-sum* message passing, the message updates from the variable nodes to the factor nodes are the same as the aforementioned *sum-product* message updates, i.e. (2b,2d,3b,3d). We only need to change the message updates from the factor nodes to the variable nodes by replacing "$\int$" in (2a,2c,3a,3c) with "max". Take $\Delta^{(t+1)}_{\Omega_n \to \lambda_l}$ for example, we will have

$$\Delta^{(t+1)}_{\Omega_n \to \lambda_l} = \text{const} + \max_{x_n, \boldsymbol{\lambda} \backslash \lambda_l} \Big[\log \Omega_n(x_n, \boldsymbol{\lambda}) + \Delta^{(t+1)}_{\Omega_n \leftarrow x_n} + \textstyle\sum_{u \neq l} \Delta^{(t)}_{\Omega_n \leftarrow \lambda_u} \Big] \tag{7}$$

Note: For the *sum-product* message passing, PE-GAMP naturally produces MMSE estimation of $\boldsymbol{x}$. Based on $p(x_n|\boldsymbol{y})$, we can also compute the MAP estimation of $\boldsymbol{x}$: $\hat{x}_n = \arg\max_{x_n} p(x_n|\boldsymbol{y})$. For the *max-sum* message passing, PE-GAMP naturally produces the "joint" MAP estimation of $\boldsymbol{x}$. Unfortunately, it is not possible to obtain any meaningful MMSE estimation of $\boldsymbol{x}$ in this case.

3. PARAMETER ESTIMATION

The priors $p(\lambda_l), p(\theta_k)$ on the parameters are usually chosen to be some simple distributions. If we do not have any knowledge on how $\boldsymbol{\lambda}, \boldsymbol{\theta}$ are distributed, we can fairly assume a uniform prior and treat $p(\lambda_l), p(\theta_k)$ as constants. Since λ_l, θ_k are treated as random variables in the PE-GAMP framework, they will be jointly estimated along with the signal $\boldsymbol{x}$ in the message-updating process.

3.1. Sum-product Message Passing

Take λ_l for example, in the PE-GAMP, we propose to approximate the underlying distribution $p^{(t+1)}_{\Omega_n \leftarrow \lambda_l}(\lambda_l|\boldsymbol{y}) \propto \exp(\Delta^{(t+1)}_{\Omega_n \leftarrow \lambda_l})$ using Dirac delta function:

$$p^{(t+1)}_{\Omega_n \leftarrow \lambda_l}(\lambda_l|\boldsymbol{y}) \approx \delta\left(\lambda_l - \hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l}\right), \tag{8}$$

where $\delta(\cdot)$ is the Dirac delta function, $\hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l}$ can be computed using either the MAP or MMSE estimation:

1. MAP estimation of λ_l: $\hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l} \coloneqq \arg\max_{\lambda_l} \Delta^{(t+1)}_{\Omega_n \leftarrow \lambda_l}$.
2. MMSE estimation of λ_l: $\hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l} \coloneqq \mathbb{E}[\lambda_l|\Delta^{(t+1)}_{\Omega_n \leftarrow \lambda_l}]$. It is the mean of the distribution $\frac{1}{\mathcal{C}}\exp(\Delta^{(t+1)}_{\Omega_n \leftarrow \lambda_l})$, where $\mathcal{C}$ is a normalizing constant.

The formulations for the rest parameters can be derived similarly. The reason behind the choice of Dirac delta approximation of $p^{(t+1)}_{\Omega_n \leftarrow \lambda_l}(\lambda_l|\boldsymbol{y})$ is its simplicity, it amounts to the scalar MAP or MMSE estimation of λ_l from the posterior distribution $p^{(t+1)}_{\Omega_n \leftarrow \lambda_l}(\lambda_l|\boldsymbol{y})$. Other approximations often make it quite difficult to compute the message $\Delta^{(t+1)}_{\Omega_n \to \lambda_l}$ in (3a) due to the lack of closed-form solutions.

3.1.1. Updated Messages under Dirac Delta Approximation

The updated messages from the factor nodes to the variable nodes are then:

$$\begin{aligned}\Delta^{(t+1)}_{\Phi_m \to x_n} &= \text{const} + \log \int_{\boldsymbol{x}\setminus x_n} \left[\Phi_m\left(y_m, \boldsymbol{x}, \hat{\boldsymbol{\theta}}^{(t)}_{\Phi_m}\right) \right.\\ &\qquad \left. \times \exp\left(\textstyle\sum_{j\neq n} \Delta^{(t)}_{\Phi_m \leftarrow x_j}\right)\right] \end{aligned} \tag{9a}$$

$$\Delta^{(t+1)}_{\Omega_n \to x_n} = \text{const} + \log \Omega_n(x_n, \hat{\boldsymbol{\lambda}}^{(t)}_{\Omega_n}) \tag{9b}$$

$$\begin{aligned}\Delta^{(t+1)}_{\Omega_n \to \lambda_l} &= \text{const} + \log \int_{x_n} \left[\Omega_n\left(x_n, \lambda_l, \hat{\boldsymbol{\lambda}}^{(t)}_{\Omega_n} \setminus \hat{\lambda}^{(t)}_{\Omega_n \leftarrow \lambda_l}\right) \right.\\ &\qquad \left. \times \exp\left(\Delta^{(t+1)}_{\Omega_n \leftarrow x_n}\right)\right] \end{aligned} \tag{9c}$$

$$\begin{aligned}\Delta^{(t+1)}_{\Phi_m \to \theta_k} &= \text{const} + \log \int_{\boldsymbol{x}} \left[\Phi_m\left(y_m, \boldsymbol{x}, \theta_k, \hat{\boldsymbol{\theta}}^{(t)}_{\Phi_m} \setminus \hat{\theta}^{(t)}_{\Phi_m \leftarrow \theta_k}\right) \right.\\ &\qquad \left. \times \exp\left(\textstyle\sum_j \Delta^{(t)}_{\Phi_m \leftarrow x_j}\right)\right], \end{aligned} \tag{9d}$$

where $\hat{\boldsymbol{\lambda}}^{(t)}_{\Omega_n}$, $\hat{\boldsymbol{\theta}}^{(t)}_{\Phi_m}$ are scalar estimates from the previous t-th iteration.

3.2. Max-sum Message Passing

Take λ_l for example, a straightforward way to solve the problems in (7) is to iteratively maximize each varaible in $\{x_n, \boldsymbol{\lambda}\setminus\lambda_l\}$ while keeping the rest fixed until convergence. However, it is inefficient and quite unnecessary. In practice, one iteration would suffice. Hence we propose to use the following solutions as the approximate maximizing parameters:

$$\hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l} \coloneqq \arg\max_{\lambda_l} \log \Omega_n\left(\hat{x}^{(t)}_n, \lambda_l, \hat{\boldsymbol{\lambda}}^{(t)}_{\Omega_n} \setminus \hat{\lambda}^{(t)}_{\Omega_n \leftarrow \lambda_l}\right) + \Delta^{(t)}_{\Omega_n \leftarrow \lambda_l}. \tag{10}$$

The updated messages from the factor nodes to the variable nodes can be obtained by replacing "$\int$" in (9) with "$\max$" like before.

4. NUMERICAL RESULTS

For the sparse signal recovery task, we usually assume the sparse signal $\boldsymbol{x}$ and the noise $\boldsymbol{w}$ are generated from the following popular input and output channels:

- Bernoulli-Gaussian (BG) Input Channel: $p(x_j|\boldsymbol{\lambda}) = (1-\lambda_1)\delta(x_j) + \lambda_1\mathcal{N}(x_j; \lambda_2, \lambda_3)$.
- Laplace Input Channel: $p(x_j|\boldsymbol{\lambda}) = \frac{\lambda_1}{2}\exp(-\lambda_1|x_j|)$.
- Additive White Gaussian Noise (AWGN) Output Channel: $p(w_i|\boldsymbol{\theta}) = \mathcal{N}(w_i; 0, \theta_1)$.

Using the above channels we can create two sparse signal recovery models: 1) BG+AWGN; 2) Laplace+AWGN.

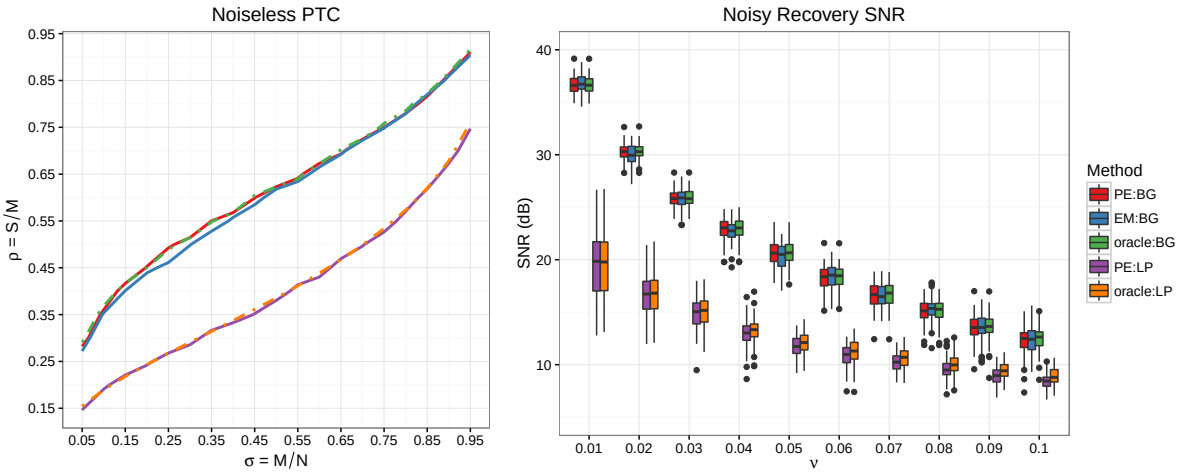
Discussion: For the model with BG input channel, *max-sum* message passing cannot be used to perform the inference task on the sparse signal since the $\tilde{\boldsymbol{x}}$ that maximizes the messages $\Delta^{(t+1)}_{\Omega_n \to \lambda_l}$ would be $\mathbf{0}$. $p(\boldsymbol{x}, \hat{\boldsymbol{\lambda}}, \hat{\boldsymbol{\theta}}|\boldsymbol{y})$ from *sum-product* message passing cannot produce any useful MAP estimation of $\boldsymbol{x}$ for the same reason. In this case, we can only use *sum-product* message passing to perform MMSE estimation of $\boldsymbol{x}$.

For the model with Laplace input channel, although *max-sum* message passing can be used to obtain the MAP estimation of $\boldsymbol{x}$, it cannot be used to compute the MAP estimation of λ_1, since the $\hat{\lambda}_1$ that maximizes (10) is always ∞ and the maximizing $\hat{\theta}_1$ is always 0. On the other hand, *sum-product* message passing can be used to compute the MMSE estimation and MAP estimation of x_n based on $p(x_n|\boldsymbol{y})$, however they doesn't have the best recovery performance. Here we propose to employ *sum-product* message passing to compute the "marginal" MAP estimates $\{\hat{\lambda}_1, \hat{\theta}_1\}$ using the marginal posterior distributions $p(\lambda_1|\boldsymbol{y}), p(\theta_1|\boldsymbol{y})$, as opposed to the MAP estimates in (11). $\{\hat{\lambda}_1, \hat{\theta}_1\}$ can then be used as the inputs to *max-sum* message passing to obtain the MAP estimate of $\boldsymbol{x}$. This essentially is the Lasso mentioned at the beginning of this paper, except now that we have provided a way to automatically estimate the parameters.

In this case, the two recovery models mentioned earlier both rely on *sum-product* message passing to perform parameter estimation. "MMSE parameter estimation" is often quite difficult to compute. In this paper we will focus on using the "MAP parameter estimation" approach to estimate the parameters. Using Dirac delta approximation we can compute MAP estimations of the parameters as follows:

$$\hat{\lambda}^{(t+1)}_{\Omega_n \leftarrow \lambda_l} = \arg\max_{\lambda_l} \Delta^{(t+1)}_{\Omega_n \leftarrow \lambda_l} \tag{11a}$$

$$\hat{\theta}^{(t+1)}_{\Phi_m \leftarrow \theta_k} = \arg\max_{\theta_k} \Delta^{(t+1)}_{\Phi_m \leftarrow \theta_k}. \tag{11b}$$



(a) (b)

Fig. 2: "PE" denotes the proposed PE-GAMP, "EM" denotes the EM-GAMP, "oracle" denotes the oracle-GAMP; "BG" denotes the recovery model with Bernoulli-Gaussian input channel, "LP" denotes the recovery model with Laplace input channel (i.e. Lasso). (a) The phase transition curves (PTC) in the noiseless case; (b) The boxplots showing the SNR (dB) of the recovered sparse signal $\hat{\boldsymbol{x}}$ under noisy conditions.

Specifically, we will use the line search method to find the maximizing $\hat{\lambda}_l^{(t+1)}, \hat{\theta}_k^{(t+1)}$, which requires the computation of the derivatives of $\Delta_{\Omega_n \leftarrow \lambda_l}^{(t+1)}, \Delta_{\Phi_m \leftarrow \theta_k}^{(t+1)}$ with respect to λ_l, θ_k. A detailed formulation of the MAP parameter estimation can be found in [20].

Since we don't have any knowledge about the priors of $\boldsymbol{\lambda}, \boldsymbol{\theta}$, we fairly choose the "uniform" prior for each parameter.The recovery performance of the proposed PE-GAMP will be compared with EM-GAMP [16] and the oracle GAMP algorithms that already know the true parameters.

4.1. Noiseless Sparse Signal Recovery

We first perform noiseless sparse signal recovery experiments and draw the empirical phase transition curves (PTC) of PE-GAMP, EM-GAMP [16] and oracle-GAMP. We fix $N = 1000$ and vary the oversampling ratio $\sigma = \frac{M}{N} \in [0.05, 0.1, 0.15, \cdots, 0.95]$ and the undersampling ratio $\rho = \frac{S}{M} \in [0.05, 0.1, 0.15, \cdots, 0.95]$, where S is the sparsity of the signal, i.e. the number of nonzero coefficients. For each combination of σ and ρ, we randomly generate 100 pairs of $\{\boldsymbol{x}, \boldsymbol{A}\}$. The nonzero entries of the sparse signal $\boldsymbol{x} \in \mathbb{R}^N$ are i.i.d. Gaussian $\mathcal{N}(0, 1)$. $\boldsymbol{A}$ is a $M \times N$ random Gaussian matrix with normalized and centralized rows. Given the measurement vector $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}$ and the sensing matrix $\boldsymbol{A}$, we try to recover the signal $\boldsymbol{x}$. If $\epsilon = \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2 / \|\boldsymbol{x}\|_2 < 10^{-3}$, the recovery is considered to be a success. Based on the 100 trials, we compute the success rate of the recovery for each combination of σ and ρ.

For the recovery model with BG input channel, the true parameters $\boldsymbol{\lambda}, \boldsymbol{\theta}$ are known and can be directly used by the oracle-GAMP. However, for the recovery model with Laplace input channel, the parameter λ_1 are unknown since the entries of $\boldsymbol{x}$ are not actually generated according to a Laplace distribution. Here we will perform maximum likelihood estimation of λ_1 based on the true signal $\boldsymbol{x}$ and use it for the oracle-GAMP.

The results of the two recovery models are shown in Fig. 2(a). The PTC curve is the contour that correspond to the 0.5 success rate in the domain $(\sigma, \rho) \in (0, 1)^2$, it divides the domain into a "success" phase (lower-right) and a "failure" phase (upper-left). Specifically, EM-GAMP by [16] doesn't have an implementation for the recovery model with Laplace input channel, we can only show the results for the model with BG input channel. We can see that the performance of PE-GAMP matches that of the oracle-GAMP in both models. PE-GAMP does the job fairly well in estimating the parameters $\boldsymbol{\lambda}, \boldsymbol{\theta}$ and recovering the sparse signals $\boldsymbol{x}$.

4.2. Noisy Sparse Signal Recovery

We next try to recover the sparse signal $\boldsymbol{x}$ from a noisy measurement vector $\boldsymbol{y}$. We would like to see how the proposed PE-GAMP behaves when an increasing amount of noise is added to the measurement. Specifically, $S = 200, M = 500, N = 1000$ are fixed, and $\boldsymbol{y}$ is generated: $\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x} + \nu\boldsymbol{w}$. $\nu > 0$ controls the amount of noise added to $\boldsymbol{y}$, the entries of $\boldsymbol{w}$ are i.i.d Gaussian $\mathcal{N}(0, 1)$. For each ν, we randomly generate 100 triples of $\{\boldsymbol{x}, \boldsymbol{A}, \boldsymbol{w}\}$. The signal to noise ratio (SNR) is used to evaluate the performances and the box plots of the results are shown in Fig. 2(b). For the recovery model with BG input channel, we can see that the proposed PE-GAMP is able to perform as well as the oracle-GAMP in recovering noisy sparse signals. For the recovery model with Laplace input channel, the proposed PE-GAMP performs slightly worse than the oracle-GAMP, this is probably due to the mismatch between the assumed distribution and the actual distribution of $\boldsymbol{x}$.

5. CONCLUSION AND FUTURE WORK

We proposed a message passing algorithm with built-in parameter estimation to recover under-sampled sparse signals. The parameters are treated as random variables with pre-specified priors, their posterior distributions can then be directly approximated by loopy belief propagation. This allows us to perform MAP and MMSE estimations of the parameters and to update them during the message passing process to recover sparse signals. In this paper we mainly focus on MAP parameter estimation and perform numerical experiments on a series of noiseless and noisy sparse signal recovery experiments. The experiments show that the proposed PE-GAMP is able to match the performance of the oracle GAMP that knows the true parameters. In the future we would like to explore the MMSE estimation of the parameters and apply the proposed PE-GAMP on real datasets.

6. REFERENCES

[1] E. J. Candès and T. Tao, "Decoding by linear programming," *IEEE Transactions on Information Theory*, vol. 51(12), pp. 4203–4215, 2005.

[2] E. J. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Transactions on Information Theory*, vol. 52(2), pp. 489–509, 2006.

[3] D. L. Donoho, "Compressed sensing," *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1289–1306, 2006.

[4] E. J. Candès and T. Tao, "Near-optimal signal recovery from random projections: Universal encoding strategies?," *IEEE Transactions on Information Theory*, vol. 52, no. 12, pp. 5406–5425, 2006.

[5] M. Aharon, M. Elad, and A. Bruckstein, "K-svd: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, 2006.

[6] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 210–227, 2009.

[7] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society*, vol. 58, pp. 267–288, 1994.

[8] D. L. Donoho, A. Maleki, and A. Montanari, "Message-passing algorithms for compressed sensing," *Proceedings of the National Academy of Sciences*, vol. 106, no. 45, pp. 18914–18919, 2009.

[9] D. L. Donoho, A. Maleki, and A. Montanari, "Message-passing algorithms for compressed sensing: I. motivation and construction," *IEEE Information Theory Workshop on Information Theory*, pp. 1–5, Jan. 2010.

[10] M. Bayati and A. Montanari, "The dynamics of message passing on dense graphs, with applications to compressed sensing," *IEEE Transactions on Information Theory*, vol. 57, no. 2, pp. 764–785, Feb 2011.

[11] S. Rangan, "Generalized approximate message passing for estimation with random linear mixing," in *Proceedings of 2011 IEEE International Symposium on Information Theory (ISIT)*, July 2011, pp. 2168–2172.

[12] M. J. Wainwright and M. I. Jordan, "Graphical models, exponential families, and variational inference," *Foundations and Trends in Machine Learning*, vol. 1, no. 1-2, pp. 1–305, 2008.

[13] C. Guo and M. E. Davies, "Near optimal compressed sensing without priors: Parametric sure approximate message passing," *IEEE Transactions on Signal Processing*, vol. 63, no. 8, pp. 2130–2141, April 2015.

[14] C. A. Metzler, A. Maleki, and R. G. Baraniuk, "From denoising to compressed sensing," *IEEE Transactions on Information Theory*, vol. 62, no. 9, pp. 5117–5144, Sep. 2016.

[15] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society*, vol. 39, no. 1, pp. 1–38, 1977.

[16] J. Vila and P. Schniter, "Expectation-maximization bernoulli-gaussian approximate message passing," in *Conference Record of the Forty Fifth Asilomar Conference on Signals, Systems and Computers (ASILOMAR)*, Nov. 2011, pp. 799–803.

[17] J. P. Vila and P. Schniter, "Expectation-maximization gaussian-mixture approximate message passing," *IEEE Transactions on Signal Processing*, vol. 61, no. 19, pp. 4658–4672, Oct 2013.

[18] U. S. Kamilov, S. Rangan, A. K. Fletcher, and M. Unser, "Approximate message passing with consistent parameter estimation and applications to sparse learning," *IEEE Transactions on Information Theory*, vol. 60, no. 5, pp. 2969–2985, May 2014.

[19] F. Krzakala, Marc Mézard, Francois Sausset, Yifan Sun, and Lenka Zdeborov, "Probabilistic reconstruction in compressed sensing: algorithms, phase diagrams, and threshold achieving matrices," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2012, no. 08, pp. P08009, 2012.

[20] S. Huang and T. D. Tran, "Sparse signal recovery using generalized approximate message passing with built-in parameter estimation," *CoRR*, vol. abs/1606.00901, 2016, `http://arxiv.org/abs/1606.00901/`.