Hybrid Approximate Message Passing

Sundeep Rangan, *Fellow, IEEE*, Alyson K. Fletcher, *Member, IEEE*, Vivek K. Goyal, *Fellow, IEEE*, Evan Byrne, *Student Member, IEEE*, and Philip Schniter, *Fellow, IEEE*

***Abstract*—Gaussian and quadratic approximations of message passing algorithms on graphs have attracted considerable recent attention due to their computational simplicity, analytic tractability, and wide applicability in optimization and statistical inference problems. This paper presents a systematic framework for incorporating such approximate message passing (AMP) methods in general graphical models. The key concept is a partition of dependencies of a general graphical model into strong and weak edges, with the weak edges representing small, linearizable couplings of variables. AMP approximations based on the central limit theorem can be readily applied to aggregates of many weak edges and integrated with standard message passing updates on the strong edges. The resulting algorithm, which we call hybrid generalized approximate message passing (HyGAMP), can yield significantly simpler implementations of sum-product and max-sum loopy belief propagation. By varying the partition of strong and weak edges, a performance-complexity tradeoff can be achieved. Group sparsity and multinomial logistic regression problems are studied as examples of the proposed methodology.**

***Index Terms*—Approximate message passing, belief propagation, sum-product algorithm, max-sum algorithm, group sparsity, multinomial logistic regression.**

I. Introduction

FOR high-dimensional optimization and inference problems, message-passing algorithms constructed from graphical models have become widely-used in many fields [2]–[4]. The fundamental principle of graphical models is to decompose high-dimensional problems into sets of smaller low-dimensional problems. The decomposition is represented using a bipartite graph, where the problem variables and factors are represented by the graph vertices and the dependencies between them represented by edges. Message passing methods such as loopy belief propagation (BP) use this graphical structure to perform optimization or approximate inference in an iterative manner. In each iteration, optimization or inference is performed "locally" on the sub-problems associated with each factor, and "messages" are passed between the variables and factors to account for the coupling between these sub-problems.

Recently, so-called "approximate message passing" (AMP) [5]–[7] and generalized AMP (GAMP) [8] methods have been developed for the case where the measurement factors depend weakly on a large number of random variables. By linearizing these weak dependencies, one can simplify standard loopy-BP algorithms and rigorously analyze their behavior in the high-dimensional limit [7]. AMP algorithms of this form have been proposed for maximum a posteriori (MAP) and minimum mean-squared error (MMSE) inference in linear models [5], [6], generalized linear models [8], and generalized bilinear models [9]–[11]. These AMP algorithms, however, assume that the underlying random variables are independent. Similarly, they assume that measurements are conditionally independent given these random variables. Thus, one may wonder how to extend these AMP methods to prior (and/or likelihood) models that include dependencies among variables (and/or measurements). By exploiting such dependencies, one can greatly improve the performance of optimization or inference. (We will show an example of this phenomenon in Section VI.)

As one solution, we present *Hybrid GAMP* (HyGAMP) algorithms for what we call *graphical model problems with linear mixing*. The basic idea is to partition the edges of the graphical model into *weak* and *strong* subsets and represent the dependencies among the weak edges using a linear transform. Assuming that the individual components of this linear transform are individually weak, the messages propagating on the weak edges can be simplified using AMP-style approximations and combined with standard loopy-BP messages on the strong edges. The proposed approach is thus a hybrid of AMP and standard loopy-BP techniques.

We detail the HyGAMP methodology using two common variants of loopy BP: the *sum-product* algorithm for inference (i.e., computation of the posterior mean) and the *max-sum* algorithm for optimization (i.e., computation of the posterior mode). For the sum-product loopy BP algorithm, we argue that the weak-edge messages can be approximated by Gaussian densities whose mean and variance computations are simplified by the Central Limit Theorem (CLT). For max-sum loopy BP, we argue that the weak-edge messages can use quadratic approximations whose parameters are easily computed using least-squares techniques.

Manuscript received October 10, 2016; revised March 23, 2017 and May 2, 2017; accepted May 25, 2017. Date of publication June 8, 2017; date of current version June 28, 2017. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Gustau Camps-Valls. The work of S. Rangan was supported in part by the National Science Foundation under Grants 1116589, 1302336, and 1547332, and in part by the industrial affiliates of NYU WIRELESS. The work of A. K. Fletcher was supported in part by the National Science Foundation under Grants 1254204 and 1738286 and in part by the Office of Naval Research under Grant N00014-15-1-2677. The work of V. K. Goyal was supported in part by the National Science Foundation under Grant 1422034. The work of E. Byrne and P. Schniter was supported in part by the National Science Foundation under Grant CCF-1527162. This paper was presented in part at the 2012 IEEE International Symposium on Information Theory, Cambridge, MA, USA, July 1–6, 2012. *(Corresponding author: Philip Schniter.)*

S. Rangan is with the Department of Electrical and Computer Engineering, New York University, Brooklyn, NY 11201 USA (e-mail: srangan@nyu.edu).

A. K. Fletcher is with the Departments of Statistics, Mathematics, Computer Science, and Electrical Engineering, University of California, Los Angeles, CA 90095 USA (e-mail: akfletcher@ucla.edu).

V. K. Goyal is with the Department of Electrical and Computer Engineering, Boston University, Boston, MA 02215 USA (e-mail: v.goyal@ieee.org).

E. Byrne and P. Schniter are with the Department of Electrical and Computer Engineering, The Ohio State University, Columbus, OH 43210 USA (e-mail: byrne.133@osu.edu; schniter@ece.osu.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSP.2017.2713759

The proposed approach can be considered as a generalization of the *turbo AMP* method proposed in [12] for clustered-sparse signal recovery. The idea behind turbo AMP is to i) partition the overall factor graph into sub-graphs with weak edges and sub-graphs with strong edges, ii) perform AMP-style message passing within the weak sub-graphs and standard sum-product BP within the strong sub-graphs, and iii) periodically interchange messages between neighboring sub-graphs. Although the turbo-AMP idea has been applied to channel estimation and equalization, wavelet image denoising, video compressive sensing, hyperspectral unmixing, and other problems in, e.g., [13]–[19], a concrete turbo-AMP algorithm that applies to generic factor graphs has never been stated. HyGAMP fills this gap. Furthermore, turbo AMP methods have been proposed exclusively with sum-product message passing. HyGAMP extends the turbo-AMP idea to max-sum message passing. Going further still, the proposed HyGAMP method generalizes turbo-AMP by allowing factor graphs with vector-valued variable nodes (in the strong and/or weak sub-graphs). As such, HyGAMP facilitates the application of AMP techniques to problems such as *group-sparse estimation* and *multinomial logistic regression*, which are outside the reach of AMP and turbo AMP.

The use of AMP-style approximations on portions of a factor graph has also been applied with joint parameter estimation and decoding for CDMA multiuser detection in [20]; in a wireless interference coordination problem in [21], and in the context of compressed sensing [22, Section 7]. The HyGAMP framework presented here unifies and extends all of these examples and thus provides a systematic procedure for incorporating Gaussian approximations of message passing in a modular manner in general graphical models.

A shorter version of this paper was published in [1]. This longer version includes derivations of the proposed algorithms, additional experiments, and many additional explanations, clarifications, and examples throughout. Note that, since the publication of [1], the HyGAMP methodology has been used to solve a variety of problems, including multiuser detection in massive MIMO [23], [24], inference for neuronal connectivity [25], fitting neural mass spatio-temporal models [26], user activity detection in cloud-radio random access [27], and decoding from pooled data [28].

II. Graphical Model Problems With Linear Mixing

Let $\mathbf{x}$ and $\mathbf{z}$ be real-valued block column vectors

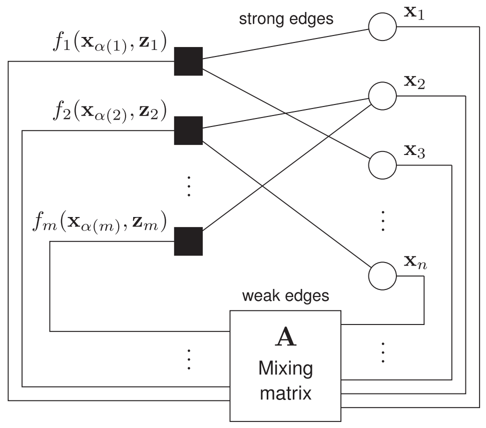
$$\mathbf{x} = [\mathbf{x}_1^\mathsf{T}, \ldots, \mathbf{x}_n^\mathsf{T}]^\mathsf{T}, \qquad \mathbf{z} = [\mathbf{z}_1^\mathsf{T}, \ldots, \mathbf{z}_m^\mathsf{T}]^\mathsf{T}, \tag{1}$$

where $^\mathsf{T}$ denotes transposition, and consider a function of these vectors of the form

$$F(\mathbf{x}, \mathbf{z}) := \sum_{i=1}^{m} f_i(\mathbf{x}_{\alpha(i)}, \mathbf{z}_i), \tag{2}$$

where, for each i, $f_i(\cdot)$ is a real-valued function; $\alpha(i)$ is a subset of the indices $\{1, \ldots, n\}$; and $\mathbf{x}_{\alpha(i)}$ is the concatenation of the vectors $\{\mathbf{x}_j,\ j \in \alpha(i)\}$. We will be interested in computations on this function subject to linear constraints of the form

$$\mathbf{z}_i = \sum_{j=1}^{n} \mathbf{A}_{ij}\mathbf{x}_j = \mathbf{A}_i\mathbf{x}, \tag{3}$$



strong edges
$\mathbf{x}_1$
$f_1(\mathbf{x}_{\alpha(1)}, \mathbf{z}_1)$
$\mathbf{x}_2$
$f_2(\mathbf{x}_{\alpha(2)}, \mathbf{z}_2)$
$\mathbf{x}_3$
$f_m(\mathbf{x}_{\alpha(m)}, \mathbf{z}_m)$
$\mathbf{x}_n$
weak edges
$\mathbf{A}$
Mixing matrix

Fig. 1. Factor graph representation of the linear mixing estimation and optimization problems. The variable nodes (circles) are connected to the factor nodes (squares) either directly (strong edges) or via the output of the linear mixing matrix $\mathbf{A}$ (weak edges). The basic GAMP algorithm [8] handles the special case where there exists no strong edges and where the variables $\mathbf{x}_j$ are scalar valued.

where each $\mathbf{A}_{ij}$ is a real-valued matrix and $\mathbf{A}_i$ is the matrix with block columns $\{\mathbf{A}_{ij}\}_{j=1}^n$. We will also let $\mathbf{A}$ be the matrix with block rows $\{\mathbf{A}_i\}_{i=1}^m$, so that we can write the linear constraints simply as $\mathbf{z} = \mathbf{A}\mathbf{x}$.

The function $F(\mathbf{x}, \mathbf{z})$ is naturally described via a graphical model as shown in Fig. 1. Specifically, we associate with $F(\mathbf{x}, \mathbf{z})$ a bipartite *factor graph* $G = (V, E)$ whose vertices V consist of n *variable nodes* corresponding to the (vector-valued) variables $\mathbf{x}_j$ and m *factor nodes* corresponding to the factors $f_i(\cdot)$ in (2). There is an edge $(i, j) \in E$ in the graph if and only if the variable $\mathbf{x}_j$ has some influence on the factor $f_i(\mathbf{x}_{\alpha(i)}, \mathbf{z}_i)$. This influence can occur in one of two mutually exclusive ways:

- The index j is in $\alpha(i)$, so that the variable $\mathbf{x}_j$ directly appears in the sub-vector $\mathbf{x}_{\alpha(i)}$ in the factor $f_i(\mathbf{x}_{\alpha(i)}, \mathbf{z}_i)$. In this case, (i, j) will be called a *strong edge*, since $\mathbf{x}_j$ can have an arbitrary and potentially-large influence on the factor.
- The matrix $\mathbf{A}_{ij}$ is nonzero, so that $\mathbf{x}_j$ affects $f_i(\mathbf{x}_{\alpha(i)}, \mathbf{z}_i)$ through its linear influence on $\mathbf{z}_i$ in (3). In this case, (i, j) will be called a *weak edge*, since the approximations we will make in the algorithms below assume that $\mathbf{A}_{ij}$ are "small." The set of weak edges into the factor node i will be denoted $\beta(i)$.

When we say that $\mathbf{A}_{ij}$ are "small," we do not mean small in an absolute sense, but rather that $\mathbf{A}_{ij}$ are such that no individual $\mathbf{x}_j$ can have a significant effect on the sum $\sum_{j=1}^n \mathbf{A}_{ij}\mathbf{x}_j$, and likewise that no individual $\mathbf{z}_i$ can have a significant effect on the sum $\sum_{i=1}^m \mathbf{z}_i^\mathsf{T}\mathbf{A}_{ij}$. One example is when $\boldsymbol{A}$ is drawn with i.i.d. sub-Gaussian entries for sufficiently large m and n. Matrices of this type are assumed in derivation and analysis of the AMP methods [5]–[8].

Together, $\alpha(i)$ and $\beta(i)$ comprise the set of all indices j for which a variable node $\mathbf{x}_j$ is connected to the factor node $f_i(\cdot)$ in the graph G. The union $\partial(i) = \alpha(i) \cup \beta(i)$ is thus the neighbor set of $f_i(\cdot)$. Similarly, for any variable node $\mathbf{x}_j$, we let (with some abuse of notation) $\alpha(j)$ be the set of all indices i for which a factor node $f_i(\cdot)$ is connected to $\mathbf{x}_j$ via a strong edge, and let $\beta(j)$ be the set of all indices i for which there exists a weak edge. The union $\partial(j) = \alpha(j) \cup \beta(j)$ is thus the neighbor set of $\mathbf{x}_j$.

Given these definitions, we are interested in two problems:

- Optimization problem P-OPT: Given a function $F(\mathbf{x},\mathbf{z})$ of the form (2) and a matrix $\mathbf{A}$, compute the maximum:
$$\widehat{\mathbf{x}} = \arg\max_{\mathbf{x}\,:\,\mathbf{z}=\mathbf{A}\mathbf{x}} F(\mathbf{x},\mathbf{z}), \qquad \widehat{\mathbf{z}} = \mathbf{A}\widehat{\mathbf{x}}. \tag{4}$$
Also, for each j, compute the *marginal value* function
$$\Delta_j(\mathbf{x}_j) := \max_{\mathbf{x}_{\setminus j}\,:\,\mathbf{z}=\mathbf{A}\mathbf{x}} F(\mathbf{x},\mathbf{z}), \tag{5}$$
where $\mathbf{x}_{\setminus j}$ is composed of $\{\mathbf{x}_r\}_{r\neq j}$.
- Expectation problem P-EXP: Given a function $F(\mathbf{x},\mathbf{z})$ of the form (2), a matrix $\mathbf{A}$, and *scale factor* $u > 0$, define the joint density
$$p(\mathbf{x}) := Z^{-1}(u)\exp\left[uF(\mathbf{x},\mathbf{z})\right], \qquad \mathbf{z} = \mathbf{A}\mathbf{x} \tag{6}$$
where $Z(u)$ is a normalization constant called the partition function (which is a function of u). Then, for this density, compute the expectations
$$\widehat{\mathbf{x}} = \mathbb{E}[\mathbf{x}], \qquad \widehat{\mathbf{z}} = \mathbb{E}[\mathbf{z}]. \tag{7}$$
Also, for each j, compute the log marginal
$$\Delta_j(\mathbf{x}_j) := \frac{1}{u}\log\int \exp\left[uF(\mathbf{x},\mathbf{z})\right]\, \mathrm{d}\mathbf{x}_{\setminus j}. \tag{8}$$
We include the scale factor u so that the definition of $F(\mathbf{x},\mathbf{z})$ allows an arbitrary scaling, as in (4).

We now show that P-OPT and P-EXP commonly arise in statistical inference. Suppose that we are given a probability density $p(\mathbf{x})$ of the form (6) for some function $F(\mathbf{x},\mathbf{z})$. The function $F(\mathbf{x},\mathbf{z})$ may depend implicitly on some observed vector $\mathbf{y}$, so that $p(\mathbf{x})$ represents the posterior density of $\mathbf{x}$ given $\mathbf{y}$. In this context, the solution $(\widehat{\mathbf{x}},\widehat{\mathbf{z}})$ to the problem P-OPT is precisely the *maximum a posteriori* (MAP) estimate of $\mathbf{x}$ and $\mathbf{z}$ given the observations $\mathbf{y}$. Similarly, the solution $(\widehat{\mathbf{x}},\widehat{\mathbf{z}})$ to the problem P-EXP is precisely the *minimum mean squared error* (MMSE) estimate. For P-EXP, the function $\Delta_j(\mathbf{x}_j)$ is the log marginal density of $\mathbf{x}_j$.

The two problems are related: A standard large deviations argument [29] shows that, under suitable conditions, as $u\to\infty$ the density $p(\mathbf{x})$ in (6) concentrates around the maxima $(\widehat{\mathbf{x}},\widehat{\mathbf{z}})$ in the solution to the problem P-OPT. As a result, the solution $(\widehat{\mathbf{x}},\widehat{\mathbf{z}})$ to P-EXP converges to the solution to P-OPT.

A. Further Assumptions and Notation

In the analysis below, we will assume that, for each factor node $f_i(\cdot)$, we have that
$$\alpha(i)\cap\beta(i) = \emptyset, \tag{9}$$
i.e., the strong and weak neighbor sets are disjoint. This assumption introduces no loss of generality: If an edge (i,j) is both weak and strong, we can modify the function $f_i(\mathbf{x}_{\alpha(i)},\mathbf{z}_i)$ to "move" the influence of $\mathbf{x}_j$ from the term $\mathbf{z}_i$ into the direct term $\mathbf{x}_{\alpha(i)}$. For example, suppose that, for some i,
$$\mathbf{z}_i = \mathbf{A}_{i1}\mathbf{x}_1 + \mathbf{A}_{i3}\mathbf{x}_3 + \mathbf{A}_{i4}\mathbf{x}_4 \ \text{ and } \ \alpha(i) = \{1,2\}.$$
In this case, the edge $(i,1)$ is both strong and weak. That is, the function $f_i(\mathbf{x}_{\alpha(i)},\mathbf{z}_i)$ depends on $\mathbf{x}_1$ through both $\mathbf{x}_{\alpha(i)}$ and through $\mathbf{z}_i$. To satisfy assumption (9), we define
$$\mathbf{z}_i^{\text{new}} = \mathbf{A}_{i3}\mathbf{x}_3 + \mathbf{A}_{i4}\mathbf{x}_4$$
$$f_i^{\text{new}}(\mathbf{x}_{\alpha(i)},\mathbf{z}_i^{\text{new}}) = f_i((\mathbf{x}_1,\mathbf{x}_2),\mathbf{A}_{i1}\mathbf{x}_1 + \mathbf{z}_i^{\text{new}}),$$
under which $f_i(\mathbf{x}_{\alpha(i)},\mathbf{z}_i) = f_i^{\text{new}}(\mathbf{x}_{\alpha(i)},\mathbf{z}_i^{\text{new}})$. Thus we can replace $f_i(\cdot)$ and $\mathbf{z}_i$ with $f_i^{\text{new}}(\cdot)$ and $\mathbf{z}_i^{\text{new}}$ that obey (9).

Even when the dependence of a factor $f_i(\mathbf{x}_{\alpha(i)},\mathbf{z}_i)$ on a variable $\mathbf{x}_j$ is only through the linear term $\mathbf{z}_i$, we may still wish to "move" the dependence to a strong edge. The reason is that the HyGAMP algorithm is designed around the assumption that the linear dependence is weak, i.e., that the elements in $\mathbf{A}_{ij}$ are small. If these elements are not small, then modeling the dependence with a strong edge improves the accuracy of HyGAMP at the expense of greater computation.

One final notation: since $\mathbf{A}_{ij}\neq\mathbf{0}$ only when $j\in\beta(i)$, we may sometimes write the summation (3) as
$$\mathbf{z}_i = \sum_{j\,\in\,\beta(i)} \mathbf{A}_{ij}\mathbf{x}_j = \mathbf{A}_{i,\beta(i)}\mathbf{x}_{\beta(i)}, \tag{10}$$
where $\mathbf{x}_{\beta(i)}$ is the sub-vector of $\mathbf{x}$ with components $j\in\beta(i)$ and $\mathbf{A}_{i,\beta(i)}$ is the corresponding sub-matrix of $\mathbf{A}_i$.

III. Motivating Examples

We begin with a basic development to show that problems with a fully separable prior and likelihood fit within our model. Then we show an extension to more complicated problems. More detailed examples are deferred to Sections VI and VII.

Linear Mixing and General Output Channel—Independent Sub-Vectors: As a simple example of a graphical model with linear mixing, consider the following estimation problem: An unknown vector $\mathbf{x}$ has independent sub-vectors $\mathbf{x}_j$, each with a joint probability density $p(\mathbf{x}_j)$. The vector $\mathbf{x}$ is passed through a linear transform to yield an output $\mathbf{z} = \mathbf{A}\mathbf{x}$. Each sub-vector $\mathbf{z}_i$ then randomly generates an output $\mathbf{y}_i$ with conditional density $p(\mathbf{y}_i|\mathbf{z}_i)$. The goal is to estimate $\mathbf{x}$ given $\mathbf{A}$, the observations $\mathbf{y}$, and knowledge of the densities.

Common applications of this formulation include the following. In *compressive sensing* [22], $\mathbf{x}$ is a sparse vector and $\mathbf{A}$ is a sensing matrix. The measurements $\mathbf{y}$ are usually modeled as $\mathbf{z}$ plus Gaussian noise, in which case $p(\mathbf{y}_i|\mathbf{z}_i)$ is Gaussian. In *binary linear classification* [30], the rows of $\mathbf{A}$ are training feature vectors, the elements of $\mathbf{y}$ are binary training labels, and $\mathbf{x}$ is a weight vector learned to predict a label from its feature vector. Here, $p(\mathbf{y}_i|\mathbf{z}_i)$ is an "activation function" that accounts for error in the linear-prediction model, often based on the logistic sigmoid. When $n > m$, a sparse weight vector $\mathbf{x}$ is sought to avoid over-fitting [31]. In *digital communications* settings, $\mathbf{x}$ might be a vector of finite-alphabet symbols and $\mathbf{A}$ a matrix representing the cumulative effect of the modulation, propagation channel, and demodulation [20]. Alternatively, $\mathbf{x}$ might represent the channel impulse response, in which case $\mathbf{A}$ is constructed from a training symbol sequence [13]. In either case, $p(\mathbf{y}_i|\mathbf{z}_i)$ is usually chosen as Gaussian, although a heavy-tailed distribution can be chosen to model impulsive noise [17].

Under the assumption that the components $\mathbf{x}_j$ are independent and the components $\mathbf{y}_i$ are conditionally independent given $\mathbf{z}$, the posterior density on $\mathbf{x}$ factors as
$$p(\mathbf{x}|\mathbf{y}) = \frac{1}{Z(\mathbf{y})}\prod_{i=1}^{m} p(\mathbf{y}_i|\mathbf{z}_i)\prod_{j=1}^{n} p(\mathbf{x}_j), \qquad \mathbf{z} = \mathbf{A}\mathbf{x},$$

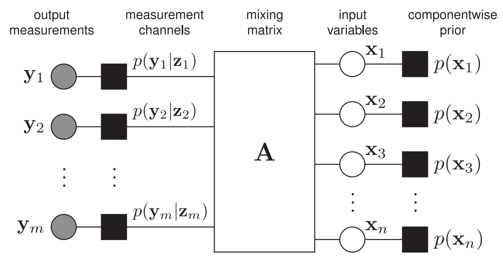


Fig. 2. An example of a simple graphical model for an estimation problem where $\mathbf{x}$ has independent components with priors $p(\mathbf{x}_j)$, $\mathbf{z} = \mathbf{A}\mathbf{x}$, and the observation vector $\mathbf{y}$ is the output of a componentwise measurement channel with transition function $p(\mathbf{y}_i|\mathbf{z}_i)$.

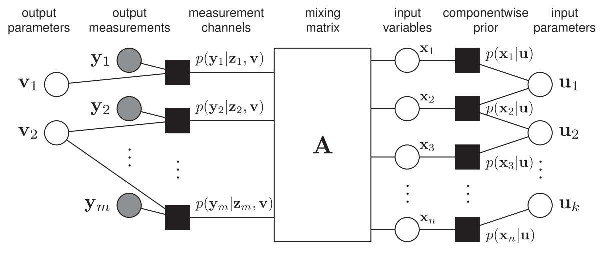


Fig. 3. A generalization of the model in Fig. 2, where the input variables $\mathbf{x}$ are themselves generated by a graphical model with latent variables $\mathbf{u}$. Similarly, the dependence of the observation vector $\mathbf{y}$ on the linear mixing output $\mathbf{z}$ is through a second graphical model.

where $Z(\mathbf{y})$ is a normalization constant. For a fixed observation $\mathbf{y}$, we can write this posterior as

$$p(\mathbf{x}|\mathbf{y}) \propto \exp\left[F(\mathbf{x},\mathbf{z})\right], \qquad \mathbf{z} = \mathbf{A}\mathbf{x},$$

where $F(\mathbf{x},\mathbf{z})$ is the log posterior, i.e.,

$$F(\mathbf{x},\mathbf{z}) = \sum_{i=1}^{m} \log p(\mathbf{y}_i|\mathbf{z}_i) + \sum_{j=1}^{n} \log p(\mathbf{x}_j),$$

and the dependence on $\mathbf{y}$ is implicit. The log posterior is therefore in the form of (2) with scale factor $u = 1$ and $m + n$ factors $\{f_i(\cdot)\}_{i=1}^{m+n}$. The first m factors can be assigned as

$$f_i(\mathbf{z}_i) = \log p(\mathbf{y}_i|\mathbf{z}_i), \qquad i = 1,\dots,m, \tag{11}$$

which do not directly depend on the terms $\mathbf{x}_j$. Thus $\alpha(i) = \emptyset$ for each $i = 1,\dots,m$. The remaining n factors are then

$$f_{m+j}(\mathbf{x}_j) = \log p(\mathbf{x}_j), \qquad j = 1,\dots,n. \tag{12}$$

For these factors, the strong edge set is the singleton $\alpha(m + j) = \{j\}$ for $j = 1,\dots,n$, and there is no linear term; we can think of $\{\mathbf{z}_{m+j}\}_{j=1}^{n}$ as zero-dimensional. The corresponding factor graph with the $m + n$ factors is shown in Fig. 2.

In the case when all $\mathbf{x}_j$ and $\mathbf{z}_i$ are scalars, the estimation problem is precisely the one targeted by GAMP [8], as mentioned in the introduction. The special subcase of measurements in additive white Gaussian noise (AWGN), i.e.,

$$y_i = z_i + w_i, \qquad w_i \sim \mathcal{N}(0,\sigma_w^2), \tag{13}$$

is the one targeted by AMP [5]–[7].

Linear Mixing and General Output Channel—Dependent Sub-Vectors: We now consider the significantly more general graphical model framework shown in Fig. 3. In this case, the input sub-vectors $\mathbf{x}_j$ may be statistically dependent on one another, with dependences described by a graphical model. Some additional latent variables, in a vector $\mathbf{u}$, may also be involved. For example, [12] used a discrete Markov chain to model clustered sparsity, [16] used discrete-Markov and Gauss-Markov chains to model slow changes in support and amplitude across multiple measurement vectors, and [14] used a discrete Markov tree to model persistence across scale in the wavelet coefficients of an image. In Section VI, we will detail the application of HyGAMP to group sparsity.

Similarly, the likelihood need not be separable in $\{\mathbf{y}_i\}$. For example, the observations $\mathbf{y}_i$ can depend on the outputs $\mathbf{z}_i$ through a second graphical model that may include additional latent variables $\mathbf{v}_i$. For example, the distribution of $\mathbf{y}_1$ given $\mathbf{z}_1$ may depend on unknown parameters $\mathbf{v}$ that also affect the distribution of $\mathbf{y}_2$ given $\mathbf{z}_2$. This technique was used in [13] to incorporate constraints on LDPC coded bits when performing turbo sparse-channel estimation, equalization, and decoding using GAMP. In Section VII, we will detail the application of HyGAMP to multinomial logistic regression.

The preceding examples offer guidance on how weak versus strong edges are typically assigned in practice. When a factor node f_i depends on a variable $\mathbf{x}_j$ as one component of a large sum $\mathbf{z}_i = \sum_{j=1}^{n} \mathbf{A}_{ij}\mathbf{x}_j$, with $\{\mathbf{A}_{ij}\}_{j=1}^{n}$ of roughly similar norm, the edge between i and j can be safely treated as *weak*. Otherwise, the edge between f_i and $\mathbf{x}_j$ is typically treated as *strong*. For example, in Fig. 3, the edge between each $\mathbf{v}_q$ and $p(\mathbf{y}_i|\mathbf{z}_i,\mathbf{v})$ is treated as strong, and the edge between each $\mathbf{u}_q$ and $p(\mathbf{x}_i|\mathbf{u})$ is treated as strong.

IV. Review of Loopy Belief Propagation

Finding exact solutions to high-dimensional P-OPT and P-EXP problems is generally intractable because they require optimization or expectation over n variables $\mathbf{x}_j$. A widely-used approximation method is loopy BP [3], [32], which reduces the high-dimensional problem to a sequence of low-dimensional problems associated with each factor $f_i(\mathbf{x}_{\alpha(i)},\mathbf{z}_i)$. We consider two common variants of loopy BP: the max-sum algorithm (MSA) for the problem P-OPT and the sum-product algorithm (SPA) for the problem P-EXP. This section will briefly review these methods, as they will be the basis of the HyGAMP algorithms described in Section V.

The MSA iteratively passes estimates of the marginal utilities $\Delta_j(\mathbf{x}_j)$ in (5) along the graph edges. Similarly, the SPA passes estimates of the log marginals $\Delta_j(\mathbf{x}_j)$ in (8). For either algorithm, we index the iterations by $t = 0,1,2,\dots$ and denote the "message" from the factor node f_i to the variable node $\mathbf{x}_j$ in the tth iteration by $\Delta_{i\to j}(t,\mathbf{x}_j)$ and the reverse message by $\Delta_{i\leftarrow j}(t,\mathbf{x}_j)$.

To describe the message updates, we introduce some additional notation. First, we note that SPA and MSA messages are equivalent up to a constant offset. That is, adding any constant (w.r.t. $\mathbf{x}_j$) to either $\Delta_{i\to j}(t,\mathbf{x}_j)$ or $\Delta_{i\leftarrow j}(t,\mathbf{x}_j)$ has no effect on the algorithm. Thus, we will use "$\equiv$" for equality up to a constant offset, i.e.,

$$\Delta(\mathbf{x}) \equiv g(\mathbf{x}) \;\Leftrightarrow\; \Delta(\mathbf{x}) = g(\mathbf{x}) + C,$$

for some constant C that does not depend on $\mathbf{x}$. Similarly, we write $p(\mathbf{x}) \propto q(\mathbf{x})$ when $p(\mathbf{x}) = Cq(\mathbf{x})$ for some constant C. Finally, for the SPA, we will fix the scale factor $u > 0$ in the problem P-EXP, and, for any function $\Delta(\cdot)$, we will write $\mathbb{E}[g(\mathbf{x});\Delta(\cdot)]$ to denote the expectation of $g(\mathbf{x})$ with respect to the density $p(\mathbf{x})$ associated with $\Delta(\cdot)$:

$$\mathbb{E}[g(\mathbf{x});\Delta(\cdot)] = \int g(\mathbf{x})p(\mathbf{x})\ \mathrm{d}\mathbf{x} \tag{14}$$

$$p(\mathbf{x}) \propto \exp\left[u\Delta(\mathbf{x})\right] \tag{15}$$

Given these definitions, the updates for the MSA and SPA variants of loopy BP are as follows:

Algorithm 1: **Loopy BP:** Consider the problems P-OPT or P-EXP above for some function $F(\mathbf{x},\mathbf{z})$ of the form (2) and matrix $\mathbf{A}$. For the problem P-EXP, fix the scale factor $u > 0$. The **MSA** for P-OPT and the **SPA** for P-EXP iterate the following steps:

0) *Initialization:* Set $t=0$ and, for each $(i,j)\in E$, set $\Delta_{i\leftarrow j}(t,\mathbf{x}_j)=0$.
1) *Factor node update:* For each edge $(i,j)\in E$, compute the function

$$H_{i\to j}(t,\mathbf{x}_{\partial(i)},\mathbf{z}_i) := f_i(\mathbf{x}_{\alpha(i)},\mathbf{z}_i) + \sum_{r\,\in\,\{\partial(i)\backslash j\}} \Delta_{i\leftarrow r}(t,\mathbf{x}_r). \tag{16}$$

For the **MSA**, compute:

$$\Delta_{i\to j}(t,\mathbf{x}_j) \equiv \max_{\substack{\mathbf{x}_{\partial(i)\backslash j}\\ \mathbf{z}_i=\mathbf{A}_i\mathbf{x}}} H_{i\to j}(t,\mathbf{x}_{\partial(i)},\mathbf{z}_i), \tag{17}$$

where the maximization is over all variables $\mathbf{x}_r$ with $r\in\partial(i)\setminus j$ and subject to the constraint $\mathbf{z}_i=\mathbf{A}_i\mathbf{x}$.

For the **SPA**, compute:

$$\Delta_{i\to j}(t,\mathbf{x}_j) \equiv \frac{1}{u}\log\int p_{i\to j}(t,\mathbf{x}_{\partial(i)})\ \mathrm{d}\mathbf{x}_{\partial(i)\backslash j}, \tag{18}$$

where the integration is over all variables $\mathbf{x}_r$ with $r\in\partial(i)\setminus j$, and $p_{i\to j}(t,\mathbf{x}_{\partial(i)})$ is the probability density

$$p_{i\to j}(t,\mathbf{x}_{\partial(i)}) \propto \exp\left[uH_{i\to j}(t,\mathbf{x}_{\partial(i)},\mathbf{z}_i=\mathbf{A}_i\mathbf{x})\right]. \tag{19}$$

2) *Variable node update:* For each $(i,j)\in E$:

$$\Delta_{i\leftarrow j}(t{+}1,\mathbf{x}_j) \equiv \sum_{\ell\,\in\,\{\partial(j)\backslash i\}} \Delta_{\ell\to j}(t,\mathbf{x}_j). \tag{20}$$

Also, let

$$\Delta_j(t{+}1,\mathbf{x}_j) \equiv \sum_{i\,\in\,\partial(j)} \Delta_{i\to j}(t,\mathbf{x}_j). \tag{21}$$

For the **MSA**, compute:

$$\widehat{\mathbf{x}}_j(t{+}1) := \arg\max_{\mathbf{x}_j} \Delta_j(t{+}1,\mathbf{x}_j). \tag{22}$$

For the **SPA**, compute:

$$\widehat{\mathbf{x}}_j(t{+}1) := \mathbb{E}\left[\mathbf{x}_j;\Delta_j(t{+}1,\cdot)\right]. \tag{23}$$

3) Increment t and return to Step 1 unless a maximum number of iterations is exceeded.

When the graph G is acyclic, it can be shown that the MSA and SPA algorithms above converge to the exact solutions to the P-OPT and P-EXP problems, respectively. When the graph G has cycles, however, the above algorithms are—in general—only approximate, but often quite accurate. The previous two statements assume that the loopy-BP messages are computed exactly, which is feasible when all variables are either Gaussian or discrete, but otherwise difficult—incurring a complexity that is exponential in general. For more details on loopy BP, see [3], [32], [33].

V. Hybrid GAMP

The HyGAMP algorithm modifies loopy BP by replacing the weak edges with approximations of their cumulative effects. By treating a subset of d dependencies as weak (as in AMP) rather than strong (as in loopy BP), the complexity of handling those dependencies shrinks from exponential in d to linear in d. In particular, HyGAMP assumes the elements of $\mathbf{A}_{ij}$ are small along any weak edge (i,j). Under this assumption, MSA-HyGAMP uses a quadratic approximation of the messages along the weak edges, reducing the factor-node update to a standard least-squares problem. Similarly, SPA-HyGAMP uses a Gaussian approximation of the weak-edge messages and applies the CLT at the factor nodes.

A derivation of the HyGAMP algorithm is given in Appendix A for the SPA and Appendix B for the MSA. We note that these derivations are "heuristic" in the sense that we do not claim any formal matching between loopy BP and the HyGAMP approximation.

To state the HyGAMP algorithm, we need additional notation. At iteration t, the HyGAMP algorithm produces estimates $\widehat{\mathbf{x}}_j(t)$ and $\widehat{\mathbf{z}}_i(t)$ of the vectors $\mathbf{x}_j$ and $\mathbf{z}_i$. Several other intermediate vectors, $\widehat{\mathbf{p}}_i(t)$, $\widehat{\mathbf{s}}_i(t)$ and $\widehat{\mathbf{r}}_j(t)$, are also produced. Associated with each of these vectors are matrices like $\mathbf{Q}^x_j(t)$ and $\mathbf{Q}^z_i(t)$ that represent Hessians for the MSA and covariances for the SPA. When referring to the inverses of these matrices, we use the notation $\mathbf{Q}^{-x}_j(t)$ to mean $(\mathbf{Q}^x_j(t))^{-1}$. Finally, for any positive definite matrix $\mathbf{Q}$ and vector $\mathbf{a}$, we define $\|\mathbf{a}\|^2_{\mathbf{Q}} := \mathbf{a}^{\mathsf{T}}\mathbf{Q}^{-1}\mathbf{a}$, which is a weighted two norm.

Algorithm 2: **HyGAMP:** Consider the problem P-OPT or P-EXP for some function $F(\mathbf{x},\mathbf{z})$ of the form (2) and matrix $\mathbf{A}$. For the problem P-EXP, fix the scale factor $u > 0$. The **MS-HyGAMP** algorithm for P-OPT and the **SP-HyGAMP** algorithm for P-EXP iterate the following steps:

0) *Initialization:* Set $t=0$ and $\widehat{\mathbf{s}}_i(-1)=\mathbf{0}\ \forall i$, and select some initial values $\Delta_{i\to j}(-1,\mathbf{x}_j)$ for each strong edge (i,j), and $\widehat{\mathbf{r}}_j(-1)$ and $\mathbf{Q}^r_j(-1)$ for each index j.
1) *Variable node update, strong edges:* For each strong edge (i,j), compute

$$\Delta_{i\leftarrow j}(t,\mathbf{x}_j) \equiv \sum_{\ell\,\in\,\{\alpha(j)\backslash i\}} \Delta_{\ell\to j}(t{-}1,\mathbf{x}_j) - \frac{1}{2}\|\widehat{\mathbf{r}}_j(t{-}1)-\mathbf{x}_j\|^2_{\mathbf{Q}^r_j(t-1)}. \tag{24}$$

2) *Variable node update, weak edges:* For each variable node j, compute

$$\Delta_j(t,\mathbf{x}_j) \equiv H^x_j(t,\mathbf{x}_j,\widehat{\mathbf{r}}_j(t{-}1),\mathbf{Q}^r_j(t{-}1)) \tag{25}$$

and

$$H^x_j(t,\mathbf{x}_j,\widehat{\mathbf{r}}_j,\mathbf{Q}^r_j) = \sum_{i\,\in\,\alpha(j)} \Delta_{i\to j}(t{-}1,\mathbf{x}_j) - \frac{1}{2}\|\widehat{\mathbf{r}}_j-\mathbf{x}_j\|^2_{\mathbf{Q}^r_j}. \tag{26}$$

For **MS-HyGAMP**,

$$\widehat{\mathbf{x}}_j(t) = \arg\max_{\mathbf{x}_j} \Delta_j(t, \mathbf{x}_j) \tag{27a}$$

$$\mathbf{Q}_j^{-x}(t) = -\frac{\partial^2}{\partial \mathbf{x}^2}\Delta_j(t, \mathbf{x}_j). \tag{27b}$$

For **SP-HyGAMP**,

$$\widehat{\mathbf{x}}_j(t) = \mathbb{E}\left(\mathbf{x}_j; \Delta_j(t, \cdot)\right) \tag{28b}$$

$$\mathbf{Q}_j^x(t) = u\,\mathrm{Cov}\left(\mathbf{x}_j; \Delta_j(t, \cdot)\right). \tag{28c}$$

3) *Factor node update, linear step:* For each factor node i, compute

$$\widehat{\mathbf{z}}_i(t) = \sum_{j\,\in\,\beta(i)} \mathbf{A}_{ij}\widehat{\mathbf{x}}_j(t) \tag{29a}$$

$$\widehat{\mathbf{p}}_i(t) = \widehat{\mathbf{z}}_i(t) - \mathbf{Q}_i^p(t)\widehat{\mathbf{s}}_i(t-1) \tag{29b}$$

$$\mathbf{Q}_i^p(t) = \sum_{j\,\in\,\beta(i)} \mathbf{A}_{ij}\mathbf{Q}_j^x(t)\mathbf{A}_{ij}^{\mathsf{T}}. \tag{29c}$$

4) *Factor node update, strong edges:* For each strong edge (i, j), compute:

$$H_{i\to j}^z(t, \mathbf{x}_{\alpha(i)}, \mathbf{z}_i, \widehat{\mathbf{p}}_i, \mathbf{Q}_i^p) := f_i(\mathbf{x}_{\alpha(i)}, \mathbf{z}_i) + \sum_{r\,\in\,\{\alpha(i)\backslash j\}} \Delta_{i\leftarrow r}(t, \mathbf{x}_r) - \frac{1}{2}\|\mathbf{z}_i - \widehat{\mathbf{p}}_i\|^2_{\mathbf{Q}_i^p}. \tag{30}$$

Then, for **MS-HyGAMP**, compute:

$$\Delta_{i\to j}(t, \mathbf{x}_j) = \max_{\mathbf{x}_{\alpha(i)\backslash j}, \mathbf{z}_i} H_{i\to j}^z(t, \mathbf{x}_{\alpha(i)}, \mathbf{z}_i, \widehat{\mathbf{p}}_i(t), \mathbf{Q}_i^p(t)), \tag{31}$$

where the maximization is jointly over $\mathbf{z}_i$ and all components $\mathbf{x}_r$ with $r \in \{\alpha(i) \setminus j\}$.
For **SP-HyGAMP**, compute:

$$\Delta_{i\to j}(t, \mathbf{x}_j) \equiv \frac{1}{u}\log\int p_{i\to j}(t, \mathbf{x}_{\alpha(i)}, \mathbf{z}_i)\, \mathrm{d}\mathbf{x}_{\alpha(i)\backslash j}\, \mathrm{d}\mathbf{z}_i, \tag{32}$$

where the integral is over $\mathbf{z}_i$ and all components $\mathbf{x}_r$ with $r \in \{\alpha(i) \setminus j\}$, and $p_{i\to j}(t, \mathbf{x}_j)$ is the probability density

$$p_{i\to j}(t, \mathbf{x}_{\alpha(i)}, \mathbf{z}_i) \propto \exp\left(uH_{i\to j}^z(t, \mathbf{x}_{\alpha(i)}.\mathbf{z}_i, \widehat{\mathbf{p}}_i(t), \mathbf{Q}_i^p(t))\right). \tag{33}$$

5) *Factor node update, weak edges:* For each factor node i, compute

$$H_i^z(t, \mathbf{x}_{\alpha(i)}, \mathbf{z}_i, \widehat{\mathbf{p}}_i, \mathbf{Q}_i^p) := f_i(\mathbf{x}_{\alpha(i)}, \mathbf{z}_i) + \sum_{r\,\in\,\alpha(i)} \Delta_{i\leftarrow r}(t, \mathbf{x}_r) - \frac{1}{2}\|\mathbf{z}_i - \widehat{\mathbf{p}}_i\|^2_{\mathbf{Q}_i^p}. \tag{34a}$$

Then, for **MS-HyGAMP**, compute:

$$(\widehat{\mathbf{x}}^0_{\alpha(i)}(t), \widehat{\mathbf{z}}^0_i(t)) := \arg\max_{\mathbf{x}, \mathbf{z}_i} H_i^z(t, \mathbf{x}_{\alpha(i)}, \mathbf{z}_i, \widehat{\mathbf{p}}_i(t), \mathbf{Q}_i^p(t)), \tag{34b}$$

$$\mathbf{D}_i^z(t) := -\frac{\partial^2}{\partial \mathbf{z}_i^2} H_i^z(t, \widehat{\mathbf{x}}^0_{\alpha(i)}, \widehat{\mathbf{z}}_i^0, \widehat{\mathbf{p}}_i(t), \mathbf{Q}_i^p(t)), \tag{34c}$$

where the maximization in (34b) is over the sub-vector $\mathbf{x}_{\alpha(i)}$ and output vector $\mathbf{z}_i$.
For **SP-HyGAMP**, let

$$\widehat{\mathbf{z}}_i^0(t) = \mathbb{E}(\mathbf{z}_i), \qquad \mathbf{D}_i^{-z}(t) = u\,\mathrm{Cov}(\mathbf{z}_i), \tag{35}$$

where $\mathbf{z}_i$ is the component of the pair $(\mathbf{x}_{\alpha(i)}, \mathbf{z}_i)$ with the joint density

$$p_i(t, \mathbf{x}_{\alpha(i)}, \mathbf{z}_i) \propto \exp\left(uH_i^z(t, \mathbf{x}_{\alpha(i)}, \mathbf{z}_i, \widehat{\mathbf{p}}_i(t), \mathbf{Q}_i^p(t))\right). \tag{36}$$

Then, for either MS-HyGAMP or SP-HyGAMP compute

$$\widehat{\mathbf{s}}_i(t) = \mathbf{Q}_i^{-p}(t)\left[\widehat{\mathbf{z}}_i^0(t) - \widehat{\mathbf{p}}_i(t)\right], \tag{37a}$$

$$\mathbf{Q}_i^s(t) = \mathbf{Q}_i^{-p}(t) - \mathbf{Q}_i^{-p}(t)\mathbf{D}_i^{-z}(t)\mathbf{Q}_i^{-p}(t). \tag{37b}$$

6) *Variable node update, linear step:* For each variable node j compute

$$\mathbf{Q}_j^{-r}(t) = \sum_{i\,\in\,\beta(j)} \mathbf{A}_{ij}^{\mathsf{T}}\mathbf{Q}_i^s(t)\mathbf{A}_{ij}, \tag{38a}$$

$$\widehat{\mathbf{r}}_j(t) = \widehat{\mathbf{x}}(t) + \mathbf{Q}_j^r(t)\sum_{i\,\in\,\beta(j)} \mathbf{A}_{ij}^{\mathsf{T}}\widehat{\mathbf{s}}_i(t). \tag{38b}$$

Increment t and return to Step 1 unless either a maximum number of iterations is exceeded or $\|\widehat{\mathbf{x}}_j(t) - \widehat{\mathbf{x}}_j(t-1)\|$ is sufficiently small.

Although the HyGAMP algorithm above appears much more complicated than standard loopy BP (Algorithm 1), HyGAMP can require dramatically less computation. Recall that the main computational difficulty of loopy BP is Step 1, the factor update. The updates (17) and (18) involve an optimization or expectation over $|\partial(i)|$ sub-vectors, where $\partial(i)$ is the set of all sub-vectors connected to the factor node i. In the HyGAMP algorithm, these computations are replaced by (31) and (32), where the optimization and expectation need only be computed over the strong edge sub-vectors $\alpha(i)$. If the number of weak edges is large, the computational savings can be dramatic. The other steps of the HyGAMP algorithms are all linear, simple least-square operations, or componentwise nonlinear functions on the individual sub-vectors.

For ease of illustration, we have only presented one form of the HyGAMP procedure. Several variants are possible:

- *Discrete distributions:* The above description assumed continuous-valued random variables x_j. The procedures can be easily modified for discrete-valued variables by appropriately replacing integrals with summations.
- *Message scheduling:* The above description also only considered a completely parallel implementation where each iteration performs exactly one update on all edges. Other so-called message schedules are also possible and may offer more efficient implementations or better convergence depending on the application (e.g., [34]–[36]).

VI. Application to Group-Sparse Signal Recovery

To illustrate the HyGAMP method, we first consider the *group-sparse estimation* problem [37], [38]. Although this problem does not utilize the full generality of HyGAMP, it provides a simple example of the HyGAMP method and has a number of existing algorithms that can be compared against.

A. HyGAMP Algorithm

A general version of the group-sparsity problem that falls within the HyGAMP framework can be described as follows. Let $\mathbf{x}$ be an n-dimensional vector with scalar components $\{x_j\}_{j=1}^n$. Vector-valued components could also be considered, but we restrict our attention to scalar components for simplicity. The component indices j of the vector $\mathbf{x}$ are divided into K (possibly overlapping) groups, $G_1, \ldots, G_K \subseteq \{1, \ldots, n\}$. We let $\gamma(j)$ be the set of group indices k such that $j \in G_k$. That is, $\gamma(j)$ is the set of groups to which the component x_j belongs.

Suppose that each group G_k can be "active" or "inactive", and each component x_j can be non-zero only when at least one group G_k is active for some $k \in \gamma(j)$. Qualitatively, a vector $\mathbf{x}$ is sparse with respect to this group structure if it is consistent with only a small number of groups being active. That is, most of the components of $\mathbf{x}$ are zero with the non-zero components having support contained in a union of a small number of groups. The group-sparse estimation problem is to estimate the vector $\mathbf{x}$ from some measurements $\mathbf{y}$. The traditional (non-group) sparse estimation problem corresponds to the special case when there are n groups of singletons, $G_j = \{j\}$.

There are many ways to model the group-sparse structure in a Bayesian manner, particularly with overlapping groups. For sake of illustration, we consider the following simple model. For each group G_k, let $\xi_k \in \{0, 1\}$ be a Boolean variable with $\xi_k = 1$ when the group G_k is active and $\xi_k = 0$ when it is inactive. We call ξ_k the "activity indicators" and model them as i.i.d. with

$$P(\xi_k = 1) = 1 - P(\xi_k = 0) = \rho \tag{39}$$

for some sparsity rate $\rho \in (0, 1)$. We assume that, given the vector $\boldsymbol{\xi}$, the components of $\mathbf{x}$ are independent with the conditional densities

$$x_j|\boldsymbol{\xi} \sim \begin{cases} 0 & \text{if } \xi_k = 0 \text{ for all } k \in \gamma(j) \\ V & \text{otherwise,} \end{cases} \tag{40}$$

where V is a random variable having the distribution of the component x_j in the event that it belongs to an active group. Finally, suppose that measurement vector $\mathbf{y}$ is generated by first passing $\mathbf{x}$ through a linear transform $\mathbf{z} = \mathbf{A}\mathbf{x}$ and then a separable componentwise measurement channel with likelihoods $p(y_i|z_i)$. Many other dependencies on the activities of $\mathbf{x}$ and measurement models $\mathbf{y}$ are possible – we use this simple model for illustration.

Under this model, the prior $\mathbf{x}$ and the measurements $\mathbf{y}$ are naturally described by a graphical model with linear mixing. Due to the independence assumptions, the posterior density of $\mathbf{x}$ given $\mathbf{y}$ factors as

$$p(\mathbf{x}|\mathbf{y}) = \frac{1}{Z(\mathbf{y})} \prod_{i=1}^{m} p(y_i|z_i) \prod_{j=1}^{n} P(x_j|\boldsymbol{\xi}_{\gamma(j)}) \prod_{k=1}^{K} P(\xi_k), \tag{41}$$

where $P(x_j|\boldsymbol{\xi}_{\gamma(j)})$ is the conditional density for the random variable in (40). The factor graph corresponding to this distribution is shown in Fig. 4.

Under this graphical model, [39, App. C] shows that SP-HyGAMP from Algorithm 2 reduces to the simple procedure outlined in Algorithm 3. A similar MS-HyGAMP variant could also be derived. In lines 8 and 9 of Algorithm 3, we used $\mathbb{E}(X|R; Q^r, \widehat{\rho})$ and $\text{var}(X|R; Q^r, \widehat{\rho})$ to denote the expectation and variance, respectively, of the scalar random variable X with

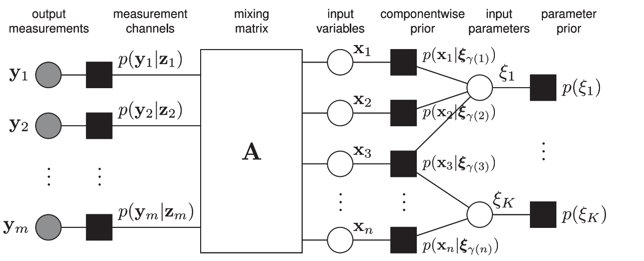

output measurements
measurement channels
mixing matrix
input variables
componentwise prior
input parameters
parameter prior
$\mathbf{y}_1$
$\mathbf{y}_2$
$\mathbf{y}_m$
$p(\mathbf{y}_1|\mathbf{z}_1)$
$p(\mathbf{y}_2|\mathbf{z}_2)$
$p(\mathbf{y}_m|\mathbf{z}_m)$
$\mathbf{A}$
$\mathbf{x}_1$
$\mathbf{x}_2$
$\mathbf{x}_3$
$\mathbf{x}_n$
$p(\mathbf{x}_1|\boldsymbol{\xi}_{\gamma(1)})$
$p(\mathbf{x}_2|\boldsymbol{\xi}_{\gamma(2)})$
$p(\mathbf{x}_3|\boldsymbol{\xi}_{\gamma(3)})$
$p(\mathbf{x}_n|\boldsymbol{\xi}_{\gamma(n)})$
ξ_1
ξ_K
$p(\xi_1)$
$p(\xi_K)$

Fig. 4. Graphical model for the group sparsity problem with overlapping groups. The group dependencies between components of the vector $\mathbf{x}$ are modeled via a set of binary latent variables $\boldsymbol{\xi}$.

Algorithm 3: SP-HyGAMP for Group Sparsity.

```
1: {Initialization}
2: $t \leftarrow 0$
3: $Q_j^r(t-1) \leftarrow \infty$
4: $\mathsf{LLR}_{j\leftarrow k}(t-1) \leftarrow \log(\rho/(1-\rho))$
5: $\widehat{\rho}_j(t) \leftarrow 1 - \prod_{k\,\in\,\gamma(j)} 1/(1 + \exp \mathsf{LLR}_{j\leftarrow k}(t-1))$
6: repeat
7:     {Basic GAMP update}
8:     $\widehat{x}_j(t) \leftarrow \mathbb{E}(X|R = \widehat{r}_j(t-1); Q_j^r(t-1), \widehat{\rho}_j(t))$
9:     $Q_j^x(t) \leftarrow \text{var}(X|R = \widehat{r}_j(t-1); Q_j^r(t-1), \widehat{\rho}_j(t))$
10:    $\widehat{z}_i(t) \leftarrow \sum_j A_{ij}\widehat{x}_j(t)$
11:    $Q_i^p(t) \leftarrow \sum_j |A_{ij}|^2 Q_j^x(t)$
12:    $\widehat{p}_i(t) \leftarrow \widehat{z}_i(t) - Q_i^p(t)\widehat{s}_i(t-1)$
13:    $\widehat{z}_i^0(t) \leftarrow \mathbb{E}(z_i|\widehat{p}_i(t), Q_i^p(t))$
14:    $Q_i^z(t) \leftarrow \text{var}(z_i|\widehat{p}_i(t), Q_i^p(t))$
15:    $\widehat{s}_i(t) \leftarrow (\widehat{z}_i^0 - \widehat{p}_i(t))/Q_i^p(t)$
16:    $Q_i^s(t) \leftarrow Q_i^{-p}(t)(1 - Q_i^z(t)/Q_i^p(t))$
17:    $Q_j^{-r}(t) \leftarrow \sum_i |A_{ij}|^2 Q_i^s(t)$
18:    $\widehat{r}_j(t) \leftarrow \widehat{x}_j(t) + Q_j^r(t)\sum_i A_{ij}\widehat{s}_i(t)$
19:    {Sparsity-rate update}
20:    $\widehat{\rho}_{j\rightarrow k}(t) \leftarrow 1 - \prod_{i\,\in\,\{\gamma(j)\backslash k\}} 1/(1 + \exp$
       $\mathsf{LLR}_{i\leftarrow k}(t-1))$
21:    Compute $\mathsf{LLR}_{j\rightarrow k}(t)$ from (44)
22:    $\mathsf{LLR}_{j\leftarrow k}(t) \leftarrow \log(\rho/(1-\rho)) + \sum_{i\,\in\,\{G_k\backslash j\}}$
       $\mathsf{LLR}_{i\rightarrow k}(t)$
23:    $\widehat{\rho}_j(t+1) \leftarrow 1 - \prod_{k\,\in\,\gamma(j)} 1/(1 + \exp \mathsf{LLR}_{j\leftarrow k}(t))$
24:    $t \leftarrow t+1$
25: until Terminate
```

density

$$X \sim \begin{cases} 0 & \text{with probability } 1 - \widehat{\rho} \\ V & \text{with probability } \widehat{\rho}; \end{cases} \tag{42}$$

and R is an AWGN corrupted version of X,

$$R = X + W, \quad W \sim \mathcal{N}(0, Q^r). \tag{43}$$

Algorithm 3 can be interpreted as the GAMP procedure from [8] with an additional update of the sparsity rates. Specifically, each iteration t of the algorithm has two stages. The first stage, labeled as the "basic GAMP update," contains the updates from the basic GAMP algorithm [8], which treats the components x_j as independent with sparsity rate $\widehat{\rho}_j(t)$. The second stage of Algorithm 3, labeled as the "sparsity-rate update," updates the sparsity rates $\widehat{\rho}_j(t)$ based on the estimates returned by the first stage.

The second stage of Algorithm 3 has a simple interpretation. The quantities $\widehat{\rho}_j(t)$ and $\widehat{\rho}_{j\to k}(t)$ can be interpreted, respectively, as estimates for the probabilities

$$\rho_j = \Pr\big(\xi_k = 1 \text{ for some } k \in \gamma(j) \,\big|\, \mathbf{y}\big)$$
$$\rho_{j\to k} = \Pr\big(\xi_i = 1 \text{ for some } i \in \{\gamma(j) \setminus k\} \,\big|\, \mathbf{y}\big).$$

That is, $\widehat{\rho}_j(t)$ is an estimate of the probability that the component x_j belongs to at least one active group and $\widehat{\rho}_{j\to k}(t)$ is an estimate of the probability that it belongs to an active group other than G_k. Similarly, the quantities $\mathsf{LLR}_{j\to k}(t)$ and $\mathsf{LLR}_{j\leftarrow k}(t)$ are estimates for the log likelihood ratios

$$\mathsf{LLR}_k = \log \frac{P\big(\xi_k = 1 \,\big|\, \mathbf{y}\big)}{P\big(\xi_k = 0 \,\big|\, \mathbf{y}\big)}.$$

Most of the updates in the second stage are natural conversions from LLR values to estimates of ρ_j and $\rho_{j\to k}$. In line 3, the LLR message is computed as

$$\mathsf{LLR}_{j\to k}(t) = \log\left(\frac{p_R(\widehat{r}_j(t); Q_j^r(t), \widehat{\rho} = 1)}{p_R(\widehat{r}_j(t); Q_j^r(t), \widehat{\rho} = \widehat{\rho}_{j\to k}(t))}\right), \quad (44)$$

where $p_R(r; Q^r, \widehat{\rho})$ is the probability density for the scalar random variable R in (43), where X has the density (42). The message (44) is the ratio of two likelihoods: the likelihood that x_j belongs to an active group and the likelihood that x_j belongs to an active group other than G_k.

To summarize, Algorithm 3 provides a simple and intuitive way to extend the basic GAMP algorithm of [8] to group-structured sparsity.

The HyGAMP algorithm for group sparsity is also extremely general. The algorithm can apply to arbitrary priors and output channels. In particular, the algorithm can incorporate logistic outputs that are often used for group sparse classification problems [40]–[42]; details are provided in [43]. Also, the method can handle arbitrary, even overlapping, groups. In contrast, the extensions of other iterative algorithms to the case of overlapping groups sometimes requires approximations; see, for example, [44]. In fact, the methodology is quite general and likely may be applied to general structured sparsity, including possibly the graphical-model-based sparse structures in image processing considered in [45].

B. Computational Complexity

In addition to its generality, the HyGAMP procedure is among the most computationally efficient for group sparsity. To illustrate this point, consider the special case when there are K non-overlapping groups of d elements each. In this case, the total vector dimension for $\mathbf{x}$ is $n = Kd$. We consider the non-overlapping case since there are many algorithms that apply to this case that we can compare against. For non-overlapping uniform groups, Table I compares the computational cost of the HyGAMP algorithm to other methods.

The computational cost of each iteration of the HyGAMP algorithm, Algorithm 3, is dominated by the matrix multiplications by $\mathbf{A}$ (line 10) and $\mathbf{A}^{\mathsf{T}}$ (line 18) and by the componentwise squares of $\mathbf{A}$ and $\mathbf{A}^{\mathsf{T}}$ (lines 11 and 17). Each of these operations has $O(mn) = O(mdK)$ cost. Note that the multiplications by componentwise-square matrices can be eliminated by using the scalar-variance version of GAMP [8]. Also, the multiplications by $\mathbf{A}$ and $\mathbf{A}^{\mathsf{T}}$ are relatively cheap if the matrix has a fast transform (e.g., FFT). The other per-iteration computations are the m scalar estimates at the output (lines 13 and 14); the n scalar estimates at the input (lines 8 and 9); and the updates of the LLRs. All of these computations are relatively simple.

TABLE I

COMPLEXITY COMPARISON FOR DIFFERENT ALGORITHMS FOR GROUP SPARSITY ESTIMATION OF A SPARSE VECTOR WITH K GROUPS, EACH GROUP OF DIMENSION d. THE NUMBER OF MEASUREMENTS IS m AND THE SPARSITY RATE IS ρ

Method	Complexity
Group-OMP [47]	$O(\rho mn^2)$
Group-Lasso [37], [38], [48]	$O(mn)$ per iteration
Relaxed BP with vector components [46]	$O(mn^2)$ per iteration
HyGAMP with vector components	$O(mnd)$ per iteration
HyGAMP with scalar components	$O(mn)$ per iteration

For the case of non-overlapping groups, the HyGAMP algorithm could also be implemented using vector-valued components. Specifically, the vector $\mathbf{x}$ can be regarded as a block vector with K vector components, each of dimension d. The general HyGAMP algorithm, Algorithm 2, can be applied on the vector-valued components. To contrast this with Algorithm 3, we will call Algorithm 3 HyGAMP with scalar components, and call the vector-valued case HyGAMP with vector components.

The cost is slightly higher for HyGAMP with vector components. In this case, there are no non-trivial strong edges since the block components are independent. However, in the update (29c), each $\mathbf{A}_{ij}$ is $1 \times d$ and $\mathbf{Q}_j^x(t)$ is $d \times d$. Thus, the computation (29c) requires mK computations of d^2 cost each for a total cost of $O(mKd^2) = O(mnd)$, which is the dominant cost. Of course, there may be a benefit in performance for HyGAMP with vector components, since it maintains the complete correlation matrix of all the components in each group. We do not investigate this possible performance benefit in this paper.

Also shown in Table I is the cost of the relaxed BP method from [46], which also uses approximate message passing similar to HyGAMP with vector components. That method, however, performs the same computations as HyGAMP on each of the mK graph edges as opposed to the $m + K$ graph vertices. It can be verified that the resulting cost has an $O(mK^2d^2) = O(mn^2)$ term.

These message passing algorithms can be compared against widely-used group LASSO methods [37], [38], which estimate $\mathbf{x}$ by solving some variant of a regularized least-squares problem of the form

$$\widehat{\mathbf{x}} := \arg\min_{\mathbf{x}} \frac{1}{2}\|\mathbf{y} - \mathbf{A}\mathbf{x}\|^2 + \gamma \sum_{j=1}^{n} \|\mathbf{x}_j\|_2, \quad (45)$$

for some regularization parameter $\gamma > 0$. The problem (45) is convex and can be solved via a number of methods including [48]–[50], the fastest of which is the SpaRSA algorithm of [48]. Interestingly, this algorithm is similar to the GAMP method in that the algorithm is an iterative procedure, where in each iteration there is a linear update followed by a componentwise scalar minimization. Like the GAMP method, the bulk of the cost is the $O(mn)$ operations per iteration for the linear transform. An alternative approach for group sparse estimation is group orthogonal matching pursuit (Group-OMP) of [42], [47], a greedy algorithm that detects one group at a time. Each round of detection requires K correlations of cost md^2. If there are on average ρK nonzero groups, the total complexity will be $O(\rho K^2md^2) = O(\rho mn^2)$. From the complexity estimates summarized in Table I it can be seen that GAMP, despite its

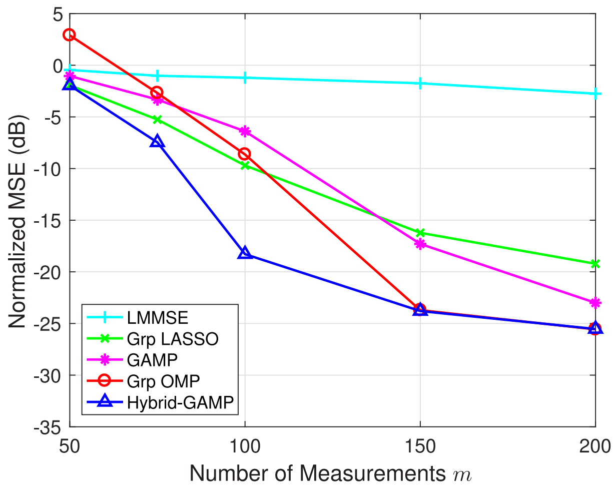


Fig. 5. Comparison of performances of various estimation algorithms for group sparsity with $n = 100$ groups of dimension $d = 4$ with a sparsity fraction of $\rho = 0.1$.

generality, is computationally as simple (per iteration) as some of the most efficient algorithms specifically designed for the group sparsity problem.

Of course, a complete comparison requires that we consider the number of iterations, not just the computation per iteration. This comparison requires further study beyond the scope of this paper. However, it is possible that the HyGAMP procedure will be favorable in this regard. Our simulations below show good convergence after only 10–20 iterations. Moreover, in the case of independent (i.e. non-group) sparsity, the number of iterations for AMP algorithms is typically small and often much less than other iterative methods. Examples in [22] show excellent convergence in 10 to 20 iterations, which is dramatically faster than the iterative soft-thresholding method of [51].

C. Numerical Experiments

Fig. 5 shows a simple simulation comparison of the mean squared error (MSE) of the HyGAMP method (Algorithm 3) along with group OMP, group LASSO, basic GAMP, and the simple linear MMSE estimator. The simulation used a vector $\mathbf{x}$ with $n = 100$ groups of size $d = 4$ and sparsity fraction of $\rho = 0.1$. The matrix was i.i.d. Gaussian and the observations were with AWGN noise at an SNR of 20 dB. The number of measurements m was varied from 50 to 200, and the plot shows the MSE for each of the methods. The HyGAMP method was run with 20 iterations. In group LASSO, at each value of m, the algorithm was simulated with several values of the regularization parameter γ in (45) and the plot shows the minimum MSE. In Group-OMP, the algorithm was run with the true value of the number of nonzero coefficients. It can be seen that the HyGAMP method is consistently as good or better than both other methods. Furthermore, HyGAMP is significantly better than basic GAMP, which exploits sparsity but not group sparsity. All code for the simulations can be found in the GAMPmatlab package [52].

We conclude that, for the problem of group-sparse recovery from AWGN-corrupted measurements, the HyGAMP method is at least comparable in performance and computational complexity to the most competitive algorithms. On top of this, HyGAMP offers a much more general framework that can include more rich modeling in both the output and input.

VII. Application to Multinomial Logistic Regression

In a second example of the HyGAMP method, we apply it to the problem of *multiclass linear classification* using the approach known as *multinomial logistic regression*.

A. Multinomial Logistic Regression

In *multiclass classification* [30], one observes a training set $\{(\mathbf{a}_i, y_i)\}_{i=1}^m$ consisting of m pairs of a feature vector $\mathbf{a}_i \in \mathbb{R}^n$ and a d-ary class label $y_i \in \{1, \ldots, d\}$. The goal is then to infer the unknown d-ary class label y_0 of an observed feature vector $\mathbf{a}_0$. In the *linear* approach to this problem, we design a weight matrix $\widehat{\mathbf{X}} \in \mathbb{R}^{n \times d}$ from the training set. Then, given an unlabeled feature vector $\mathbf{a}_0$, we first generate a vector of linear "scores" $\mathbf{z}_0 := \widehat{\mathbf{X}}^{\mathsf{T}} \mathbf{a}_0 \in \mathbb{R}^d$, and estimate the class label y_0 as the index of the largest score, i.e.,

$$\widehat{y}_0 = \arg\max_k [\mathbf{z}_0]_k. \tag{46}$$

Multinomial linear regression (MLR) [30] is one of the best known methods to design the weight matrix $\mathbf{X}$. There, the labels $\{y_i\}$ are modeled as conditionally independent given the scores $\{\mathbf{z}_i\}$, where $\mathbf{z}_i := \mathbf{X}^{\mathsf{T}} \mathbf{a}_i$. That is,

$$\Pr(\mathbf{y}|\mathbf{X}; \mathbf{A}) = \prod_{i=1}^m p_{\mathsf{mlr}}(y_i | \mathbf{X}^{\mathsf{T}} \mathbf{a}_i), \tag{47a}$$

where $p_{\mathsf{mlr}}(y_i|\mathbf{z}_i)$ is the multinomial logistic pmf,

$$p_{\mathsf{mlr}}(y_i|\mathbf{z}_i) := \frac{\exp\left([\mathbf{z}_i]_{y_i}\right)}{\sum_{k=1}^d \exp\left([\mathbf{z}_i]_k\right)}, \quad y_i \in \{1, \ldots, d\}. \tag{47b}$$

The rows $\mathbf{x}_j^{\mathsf{T}}$ of the weight matrix $\mathbf{X}$ are then modeled as i.i.d.,

$$p(\mathbf{X}) = \prod_{j=1}^n p(\mathbf{x}_j). \tag{48}$$

For log-convex $p(\mathbf{x}_j)$, MAP estimation of $\mathbf{X}$ is a convex problem. The log-convex Laplacian prior

$$p_{\mathsf{lap}}(\mathbf{x}_j) = (\lambda/2)^d \exp\left(-\lambda \|\mathbf{x}_j\|_1\right) \tag{49}$$

is a popular choice for $p(\mathbf{x}_j)$ that promotes sparsity in the designed weight matrix $\widehat{\mathbf{X}}$. Sparsity is essential in the case that the feature dimension n is much larger than the number of training examples m. Fast implementations of sparse MLR were proposed in [31] and refined in [53].

B. HyGAMP Algorithm

Max-sum HyGAMP (MS-HyGAMP) can be directly applied to solve the above optimization problem. To do this, we set $\mathbf{A}_{ij} = [\mathbf{a}_i]_j \mathbf{I}_d \ \forall i, j$ and, recalling (11), we choose $f_i(\mathbf{z}_i) = \log p_{\mathsf{mlr}}(y_i|\mathbf{z}_i) \ \forall i = 1, \ldots, m$, and recalling (12), we choose $f_{m+j}(\mathbf{x}_j) = \log p_{\mathsf{lap}}(\mathbf{x}_j) \, \forall j = 1, \ldots, n$. Then (27a) boils down to

$$\widehat{\mathbf{x}}_j = \arg\min_{\mathbf{x}} \frac{1}{2} \|\mathbf{x} - \widehat{\mathbf{r}}_j\|^2_{\mathbf{Q}_j^r} + \lambda \|\mathbf{x}\|_1, \tag{50}$$

where $\|\mathbf{x}\|^2_{\mathbf{Q}} := \mathbf{x}^{\mathsf{T}} \mathbf{Q}^{-1} \mathbf{x}$, and (34b) boils down to

$$\widehat{\mathbf{z}}_i = \arg\min_{\mathbf{z}} \frac{1}{2} \|\mathbf{z} - \widehat{\mathbf{p}}_i\|^2_{\mathbf{Q}_i^p} - \log p_{\mathsf{mlr}}(y_i|\mathbf{z}). \tag{51}$$

Both problems are convex and can be solved using standard methods, e.g., majorization–minimization or Newton's method in the case of (51). For more details, including the implementation of (27b) and (34c), we refer the reader to [54].

SP-HyGAMP can also be applied to MLR, again using the likelihood (47). However, rather than the Laplacian prior (49), we suggest choosing the Bernoulli-multivariate-Gaussian prior

$$p(\mathbf{X}) = \prod_{j=1}^{n} p_{\mathsf{bg}}(\mathbf{x}_j) \tag{52a}$$

$$p_{\mathsf{bg}}(\mathbf{x}_j) = \beta\delta(\mathbf{x}_j) + (1-\beta)\mathcal{N}(\mathbf{x}_j; \mathbf{0}, q\mathbf{I}) \tag{52b}$$

with $\beta \in [0,1)$, which promotes approximate row-sparsity in $\widehat{\mathbf{X}}$ under sum-product inference. In this case, it can be shown [54] that (28) can be computed in closed form as

$$C_n = 1 + \frac{1-\beta}{\beta} \frac{\mathcal{N}(\mathbf{0}; \widehat{\mathbf{r}}_j, \mathbf{Q}_j^r)}{\mathcal{N}(\mathbf{0}; \widehat{\mathbf{r}}_j, q\mathbf{I} + \mathbf{Q}_j^r)} \tag{53}$$

$$\widehat{\mathbf{x}}_j = \frac{1}{C_n}\left(\mathbf{I} + \frac{1}{q}\mathbf{Q}_j^r\right)^{-1} \widehat{\mathbf{r}}_j \tag{54}$$

$$\mathbf{Q}_j^x = \frac{1}{C_n}\left(\mathbf{I} + \frac{1}{q}\mathbf{Q}_j^r\right)^{-1} \mathbf{Q}_j^r + (C_n - 1)\widehat{\mathbf{x}}_j\widehat{\mathbf{x}}_j^{\mathsf{T}}. \tag{55}$$

Although we are not aware of a closed-form solution to (35), it can be approximated using numerical integration.

C. Numerical Experiments

We will now describe the results of two experiments used to evaluate the application of HyGAMP to sparse MLR. In these experiments, SP-HyGAMP and MS-HyGAMP were compared to two state-of-the-art sparse MLR algorithms: SBMLR from [55] and GLMNET from [53].

1) Synthetic Data: We first performed an experiment on synthetic data with $d = 3$ classes, $n = 500$ features, and $m = 102$ examples. The use of synthetic data allowed us to analytically compute the expected test-error rate associated with the designed weight matrices $\widehat{\mathbf{X}}$.

To generate the synthetic data, we first constructed the set of training labels $\{y_i\}$ such that m/d training samples were dedicated to each class. Then we drew feature vectors $\{\mathbf{a}_i\}$ i.i.d. from the class-conditional density $\mathbf{a}_i|y_i \sim \mathcal{N}(\boldsymbol{\mu}_{y_i}, v\mathbf{I}_n)$. The class means $\{\boldsymbol{\mu}_y\}_{y=1}^d$ were 10-sparse, with support chosen uniformly at random and with non-zero entries chosen uniformly from the columns of a 10×10 random orthonormal matrix. The parameter v was then chosen to achieve a Bayes error rate of 10%. Thus, only 10 of the 500 features were discriminatory. Note that the data-generation model is not matched to the statistical model assumed in the derivation of MS-HyGAMP or SP-HyGAMP.

To test the algorithms, we performed 12 trials, where in each trial we invoked each algorithm-under-test on randomly generated training data and then computed the resulting expected test-error rate. The SP-HyGAMP algorithm used (52) with parameters (β, q) tuned over a 3×5 logarithmically-spaced grid using 5-fold cross-validation (CV). The GLMNET algorithm, which solves the same convex optimization problem as MS-HyGAMP, tuned λ in (49) over 25 logarithmically-spaced values using 5-fold CV. The same CV-optimal λ was then used for MS-HyGAMP. Finally, SBMLR is parameter-free, and thus did not require tuning.

TABLE II
RESULTS FOR THE SYNTHETIC DATA EXPERIMENT

Algorithm	% Error	$\widehat{K}_{99}$	$\widehat{K}_{\ell_0}$
GLMNET	14.787	13.25	25.75
MS-HyGAMP	14.787	13.25	25.75
SBMLR	14.059	15.08	28.92
SP-HyGAMP	**13.981**	16.08	1500

For a designed weight matrix $\widehat{\mathbf{X}} = [\widehat{\mathbf{x}}_1, \ldots, \widehat{\mathbf{x}}_d]$, the expected test-error rate can be analytically computed [54] as

$$\Pr\{\text{err}\} = 1 - \frac{1}{d}\sum_{y=1}^{d} \Pr\{\text{cor}|y\} \tag{56}$$

$$\Pr\{\text{cor}|y\} = \Pr \bigcap_{k \neq y} \left\{ (\widehat{\mathbf{x}}_y - \widehat{\mathbf{x}}_k)^{\mathsf{T}}\mathbf{a} < (\widehat{\mathbf{x}}_y - \widehat{\mathbf{x}}_k)^{\mathsf{T}}\boldsymbol{\mu}_y \right\}, \tag{57}$$

where $\mathbf{a} \sim \mathcal{N}(\mathbf{0}, v\mathbf{I}_n)$ and the multivariate normal cdf in (57) was computed using Matlab's `mvncdf`.

In addition to computing the expected test-error rate, we computed two metrics for the sparsity of the designed weight matrices. The metric $\widehat{K}_{\ell_0} = \|\widehat{\mathbf{X}}\|_0$ quantifies absolute sparsity, i.e., the number of non-zero elements in $\widehat{\mathbf{X}}$. But since the weights returned by SP-HyGAMP are non-zero with probability one, we also computed the "effective sparsity" $\widehat{K}_{99}$, which is defined as the minimum number of elements in $\widehat{\mathbf{X}}$ required to reach 99% of $\|\widehat{\mathbf{X}}\|_F^2$.

Table II shows the expected test-error rate, $\widehat{K}_{99}$, and $\widehat{K}_{\ell_0}$ of each algorithm, averaged over 12 independent trials. From this table, we see that MS-HyGAMP and GLMNET matched on all metrics. This result is expected because the two algorithms aim to solve the same convex problem, and it offers evidence that they do in fact solve the problem. Thus, in the sequel, we report only the results of GLMNET. Next, Table II shows that the SP-HyGAMP achieved the best expected test-error rate of 13.981%, with SBMLR achieving the second best. For comparison, we recall that the Bayes (i.e., minimum) expected error rate was 10% in this experiment. The table also shows that the (average) effective sparsity $\widehat{K}_{99}$ was similar for all algorithms, and smaller than the sparsity of the Bayes' optimal classifier for this dataset, which is $\widehat{K}_{\ell_0} = 30$.

2) Handwritten Digit Classification: In the second experiment, we tested SP-HyGAMP, GLMNET, and SBMLR on the Mixed National Institute of Standards and Technology (MNIST) dataset [56]. The MNIST dataset consists of $m = 70\,000$ total images of handwritten digits 0 through 9, hence $d = 10$. Each image has $n = 784$ pixels. In this experiment we performed 24 trials, where in each trial we randomly partitioned the total dataset into a training and testing portion. Within each trial, we varied the number of image samples in the training partition from $m = 56$ to $m = 1000$. Using the training data, we used each algorithm-under-test to design a weight matrix, which was then used to compute an empirical error-rate on the test partition of the dataset. In this experiment, SP-HyGAMP and GLMNET tuned their associated parameters in a similar manner as in the synthetic experiment. However, they used 2-fold CV instead of 5-fold CV to reduce computation.

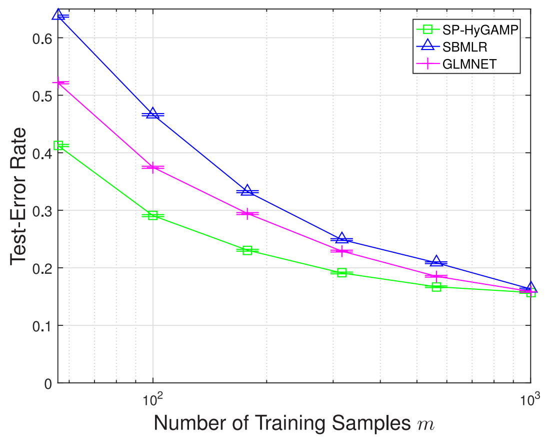


Fig. 6. Classification results for MNIST dataset.

Figure 6 shows the empirical test-error rate versus the number of training samples m, averaged over the 24 random trials. The error bars indicate the standard deviation of the empirical error-rate estimate. The figure shows that, for all m, SP-HyGAMP achieved the best test-error rate and GLMNET achieved the second best. The figure also shows that, for all algorithms, the test-error rate decreased to a common value as the number of training samples m increased. This is not surprising; we expect that, with enough training data, any reasonable approach should recover a close approximation to the Bayes-optimal linear classifier. A much more difficult problem is designing a good linear classifier from limited training data, and, for this problem, Figure 6 shows that SP-HyGAMP beats the competition.

D. Simplified HyGAMP and EM/SURE Tuning

When directly applied to multinomial logistic regression, each iteration of HyGAMP involves the update of $O(m+n)$ multivariate Gaussian pdfs, each of dimension d, for a total complexity of $O((m+n)d^3)$ per iteration. This complexity can be quite large in practice, especially relative to state-of-the-art methods like GLMNET and SBMLR. Furthermore, in its more direct form, HyGAMP assumes knowledge of the statistical parameters of its prior and likelihood. In order to tune these parameters to the data, it was suggested above to use cross-validation (as with GLMNET). But K-fold cross-validation of P parameters using G hypothesized values of each parameter requires the training and evaluation of KG^P classifiers, which can be very expensive in practice.

Fortunately, for multinomial logistic regression, it is possible to modify HyGAMP in such a way that the complexity of the resulting method becomes competitive with GLMNET and SBMLR. The modification consists of two parts: i) a *simplification* of HyGAMP wherein the covariance matrices $\mathbf{Q}^r_j, \mathbf{Q}^x_j, \mathbf{Q}^p_i, \mathbf{Q}^z_i$ are constrained to be diagonal; and ii) an application of EM-based [57] and SURE-based [58] parameter tuning to the priors and likelihoods relevant to multinomial logistic regression. A complete description of EM/SURE-tuned simplified HyGAMP (SHyGAMP) for multinomial logistic regression can be found in [59], with full derivations in [54]. In [59], a detailed numerical study establishes that EM/SURE-tuned SHyGAMP is competitive in both performance and complexity with GLMNET and SBMLR. Due to space limitations, we refer the interested reader to [54] and [59] for more details.

We conclude by saying that, although the "direct" application of HyGAMP from Section V may not lead to a complexity that is always competitive with state-of-the-art methods, it acts as an important *first step* in deriving simplified and/or enhanced version of HyGAMP. This underscores the importance of HyGAMP as stated in Section V.

VIII. Conclusion

A general model for optimization and statistical inference based on graphical models with linear mixing was presented. The linear mixing components of the graphical model account for interactions through aggregates of large numbers of small, linearizable perturbations. Gaussian and second-order approximations are shown to greatly simplify the implementation of loopy BP for these interactions, and the HyGAMP framework presented here enables these approximations to be incorporated in a systematic manner in general graphical models. Simulations were presented for group sparsity and multinomial logistic regression, where the HyGAMP method has equal or superior performance to existing methods. Although we saw that, in multinomial logistic regression, a direct application of HyGAMP does not lead to state-of-the-art computationally complexity, a modification of the HyGAMP presented here suffices to address the complexity issue [54], [59]. The generality of the proposed HyGAMP algorithm also allows its application to many other problems beyond these two examples, such as multiuser detection in massive MIMO [23], [24], inference for neuronal connectivity [25], fitting neural mass spatio-temporal models [26], user activity detection in cloud-radio random access [27], and decoding from pooled data [28]. In addition to pursuing such applications, future work will focus on establishing rigorous theoretical analyses along the lines of [7], [8] for specific instances of HyGAMP.

Appendix A
Derivation of SP-HyGAMP

A. Preliminary Lemma

Before deriving the SP-HyGAMP algorithm, we need the following result. Let $H(\mathbf{w},\mathbf{v})$ be a real-valued function of vectors $\mathbf{w}$ and $\mathbf{v}$ of the form

$$H(\mathbf{w},\mathbf{v}) = H_0(\mathbf{w}) - \frac{1}{2}\|\mathbf{w}-\mathbf{v}\|^2_{\mathbf{Q}^v} \tag{58}$$

for some positive definite matrix $\mathbf{Q}^v$.

Lemma 1: Suppose that $\mathbf{W}$ and $\mathbf{V}$ are random vectors with a conditional probability distribution function of the form

$$p_{\mathbf{W}|\mathbf{V}}(\mathbf{w}|\mathbf{v}) = \frac{1}{Z(\mathbf{v})}\exp\left[uH(\mathbf{w},\mathbf{v})\right],$$

where $H(\mathbf{w},\mathbf{v})$ is given in (58), $u>0$ is some constant and $Z(\mathbf{v})$ is a normalization constant (called the partition function). Then,

$$\frac{\partial}{\partial\mathbf{v}}\widehat{\mathbf{x}}(\mathbf{v}) = \mathbf{D}\mathbf{Q}^{-v} \tag{59a}$$

$$\frac{\partial}{\partial\mathbf{v}}\log Z(\mathbf{v}) = \mathbf{Q}^{-v}(\widehat{\mathbf{x}}(\mathbf{v})-\mathbf{v}) \tag{59b}$$

$$\frac{\partial^2}{\partial\mathbf{v}^2}\log Z(\mathbf{v}) = -\mathbf{Q}^{-v}+\mathbf{Q}^{-v}\mathbf{D}\mathbf{Q}^{-v} \tag{59c}$$

where

$$\widehat{\mathbf{x}}(\mathbf{v}) = \mathbb{E}[\mathbf{W}|\mathbf{V}=\mathbf{v}], \;\; \mathbf{D} = u\mathrm{Cov}(\mathbf{W}|\mathbf{V}=\mathbf{v}).$$

Proof: The relations are standard properties of exponential families [3]. ■

B. SP-HyGAMP Approximation

First partition the objective function $H_{i\to j}(\cdot)$ in (16) as

$$\begin{aligned} &H_{i\to j}(t, \mathbf{x}_{\partial(i)}, \mathbf{z}_i) \\ &\quad = H^{\text{strong}}_{i\to j}(t, \mathbf{x}_{\alpha(i)}, \mathbf{z}_i) + H^{\text{weak}}_{i\to j}(t, \mathbf{x}_{\beta(i)}), \end{aligned} \tag{60}$$

where

$$\begin{aligned} &H^{\text{strong}}_{i\to j}(t, \mathbf{x}_{\alpha(i)}, \mathbf{z}_i) \\ &\quad := f_i(\mathbf{x}_{\alpha(i)}, \mathbf{z}_i) + \sum_{r\,\in\,\{\alpha(i)\backslash j\}} \Delta_{i\leftarrow r}(t, \mathbf{x}_r), \end{aligned} \tag{61a}$$

$$H^{\text{weak}}_{i\to j}(t, \mathbf{x}_{\beta(i)}) := \sum_{r\,\in\,\{\beta(i)\backslash j\}} \Delta_{i\leftarrow r}(t, \mathbf{x}_r). \tag{61b}$$

That is, we have separated the terms in $H_{i\to j}(\cdot)$ between the strong and weak edges.

Then, the marginal distribution $p_{i\to j}(t, \mathbf{x}_j)$ of the distribution $p_{i\to j}(t, \mathbf{x}_{\partial(i)})$ in (19) can be re-written as

$$\begin{aligned} p_{i\to j}(t, \mathbf{x}_j) &= \int p_{i\to j}(t, \mathbf{x}_{\partial(i)}) d\mathbf{x}_{\partial(i)\backslash j} \\ &\propto \int \psi^{\text{strong}}_{i\to j}(t, \mathbf{x}_j, \mathbf{z}_i) \psi^{\text{weak}}_{i\to j}(t, \mathbf{x}_j, \mathbf{z}_i) d\mathbf{z}_i, \end{aligned} \tag{62}$$

where

$$\begin{aligned} &\psi^{\text{strong}}_{i\to j}(t, \mathbf{x}_j, \mathbf{z}_i) \\ &\quad \propto \int_{\mathbf{x}_{\alpha(i)\backslash j}} \exp\left[u H^{\text{strong}}_{i\to j}(t, \mathbf{x}_{\alpha(i)}, \mathbf{z}_i)\right] d\mathbf{x}_{\alpha(i)\backslash j} \end{aligned} \tag{63a}$$

$$\begin{aligned} &\psi^{\text{weak}}_{i\to j}(t, \mathbf{x}_j, \mathbf{z}_i) \\ &\quad \propto \int_{\substack{\mathbf{x}_{\beta(i)\backslash j} \\ \mathbf{z}_i = \mathbf{A}_i \mathbf{x}}} \exp\left[u H^{\text{weak}}_{i\to j}(t, \mathbf{x}_{\beta(i)})\right] d\mathbf{x}_{\beta(i)\backslash j} \end{aligned} \tag{63b}$$

and the integration in (63a) is over the variables $\mathbf{x}_r$ with $r \in \alpha(i) \setminus j$, and and the integration in (63b) is over the variables $\mathbf{x}_r$ with $r \in \beta(i) \setminus j$, and $\mathbf{z}_i = \mathbf{A}_i \mathbf{x}$.

To approximate $p_{i\to j}(t, \mathbf{x}_j)$ in (62), we separately consider the cases when (i, j) is weak edge versus a strong edge. We begin with the weak edge case. That is, $j \in \beta(i)$. Let

$$\widehat{\mathbf{x}}_j(t) := \mathbb{E}[\mathbf{x}_j; \Delta_j(t, \cdot)], \tag{64a}$$

$$\widehat{\mathbf{x}}_{i\leftarrow j}(t) := \mathbb{E}[\mathbf{x}_j; \Delta_{i\leftarrow j}(t, \cdot)], \tag{64b}$$

$$\mathbf{Q}^x_j(t) := u\,\mathrm{Cov}[\mathbf{x}_j; \Delta_j(t, \cdot)] \tag{64c}$$

$$\mathbf{Q}^x_{i\leftarrow j}(t) := u\,\mathrm{Cov}[\mathbf{x}_j; \Delta_{i\leftarrow j}(t, \cdot)], \tag{64d}$$

where we have used the notation $\mathbb{E}[g(\mathbf{x}); \Delta(\cdot)]$ from (14).

Now, using the expression for $H^{\text{weak}}_{i\to j}(t, \mathbf{x}_{\beta(i)})$ in (61b), it can be verified that $\psi^{\text{weak}}_{i\to j}(t, \mathbf{x}_j, \mathbf{z}_i)$ is equivalent to the probability distribution of a random variable

$$\mathbf{z}_i = \mathbf{A}_{ij}\mathbf{x}_j + \sum_{r\,\in\,\{\beta(i)\backslash j\}} \mathbf{A}_{ir}\mathbf{x}_r, \tag{65}$$

with the variables $\mathbf{x}_r$ being independent with probability distribution

$$p(\mathbf{x}_r) \propto \exp(u\Delta_{i\leftarrow r}(\mathbf{x}_r)).$$

Moreover, $\widehat{\mathbf{x}}_{i\leftarrow j}(t)$ and $\mathbf{Q}^x_{i\leftarrow j}(t)/u$ in (64) are precisely the mean and variance of the random variables $\mathbf{x}_j$ under this distribution. Therefore, if the summation in (65) is over a large number of terms, we can then use the CLT to approximate the variable in $\mathbf{z}_i$ in (65) as Gaussian, with distribution $\psi^{\text{weak}}_{i\to j}(t, \mathbf{x}_j, \mathbf{z}_i)$ given by

$$\psi^{\text{weak}}_{i\to j}(t, \mathbf{x}_j, \mathbf{z}_i) \approx \mathcal{N}(\mathbf{A}_{ij}\mathbf{x}_j + \widehat{\mathbf{p}}_{i\to j}(t), \mathbf{Q}^p_{i\to j}(t)/u), \tag{66}$$

where

$$\widehat{\mathbf{p}}_{i\to j}(t) = \sum_{r\,\in\,\{\beta(i)\backslash j\}} \mathbf{A}_{ir}\widehat{\mathbf{x}}_{i\leftarrow r}(t) \tag{67a}$$

$$\mathbf{Q}^p_{i\to j}(t) = \sum_{r\,\in\,\{\beta(i)\backslash j\}} \mathbf{A}_{ir}\mathbf{Q}^x_r(t)\mathbf{A}^*_{ir}. \tag{67b}$$

Substituting this Gaussian approximation into the probability distribution $p_{i\to j}(t, \mathbf{x}_{\partial(i)}, \mathbf{z}_i)$ in (19), and then using the definitions in (61a) and (63a), we obtain the following approximation of the message in (18),

$$\Delta_{i\to j}(t, \mathbf{x}_j) \approx G_i(t, \mathbf{A}_{ij}\mathbf{x}_j + \widehat{\mathbf{p}}_{i\to j}(t), \mathbf{Q}^p_{i\to j}(t)), \tag{68}$$

where

$$\begin{aligned} &G_i(t, \mathbf{p}_i, \mathbf{Q}^p_i) \\ &\quad := \frac{1}{u}\log\int \exp\left[u H^z_i(t, \mathbf{x}_{\alpha(i)}, \mathbf{z}_i, \mathbf{p}_i, \mathbf{Q}^p_i)\right] d\mathbf{x}_{\alpha(i)} d\mathbf{z}_i \end{aligned} \tag{69}$$

and where $H^z_i(\cdot)$ is given in (34a).

Now define

$$\widehat{\mathbf{p}}_i(t) = \sum_{r\,\in\,\beta(i)} \mathbf{A}_{ir}\widehat{\mathbf{x}}_{i\leftarrow r}(t) \tag{70a}$$

$$\mathbf{Q}^p_i(t) = \sum_{r\,\in\,\beta(i)} \mathbf{A}_{ir}\mathbf{Q}^x_r(t)\mathbf{A}^*_{ir}, \tag{70b}$$

so that the expressions in (67) can be re-written as

$$\widehat{\mathbf{p}}_{i\to j}(t) = \widehat{\mathbf{p}}_i(t) - \mathbf{A}_{ij}\widehat{\mathbf{x}}_{i\leftarrow j}(t) \tag{71a}$$

$$\mathbf{Q}^p_{i\to j}(t) = \mathbf{Q}^p_i(t) - \mathbf{A}_{ij}\mathbf{Q}^x_j(t)\mathbf{A}^*_{ij}. \tag{71b}$$

Also, let

$$\widehat{\mathbf{s}}_i(t) = \frac{\partial}{\partial\widehat{\mathbf{p}}} G_i(t, \widehat{\mathbf{p}}_i(t), \mathbf{Q}^p_i(t)) \tag{72a}$$

$$\mathbf{Q}^{-s}_i(t) = -\frac{\partial^2}{\partial\widehat{\mathbf{p}}^2} G_i(t, \widehat{\mathbf{p}}_i(t), \mathbf{Q}^p_i(t)). \tag{72b}$$

Using Lemma 1, one can show that the definitions in (72) agree with the updates (37) where $\widehat{\mathbf{z}}^0_i(t)$ and $\mathbf{Q}^z_i(t)$ are the mean and covariance of the random variable $\mathbf{z}_i$ with the distribution (36).

Applying (72), we can take a second-order approximation of (68) as

$$\begin{aligned}\Delta_{i\to j}(t,\mathbf{x}_j) \approx\ & \text{const} + \widehat{\mathbf{s}}_i(t)^*\mathbf{A}_{ij}(\mathbf{x}_j - \widehat{\mathbf{x}}_j(t)) \\ & - \frac{1}{2}\|\mathbf{A}_{ij}(\mathbf{x}_j - \widehat{\mathbf{x}}_j(t))\|^2_{\mathbf{Q}^s_i(t)} \\ =\ & \text{const} + \left[\mathbf{A}^*_{ij}\mathbf{s}_i(t) + \mathbf{A}^*_{ij}\mathbf{Q}^s_i(t)\mathbf{A}_{ij}\widehat{\mathbf{x}}_j(t)\right]^*\mathbf{x}_j \\ & + \frac{1}{2}\mathbf{x}^*_j\mathbf{A}^*_{ij}\mathbf{Q}^s_i(t)\mathbf{A}_{ij}\mathbf{x}_j \end{aligned} \tag{73}$$

for all weak edges (i,j).

Next consider the case when $j \notin \beta(i)$ so that (i,j) is a strong edge. In this case, $\psi^{\text{weak}}_{i\to j}(t,\mathbf{x}_j,\mathbf{z}_i)$ does not depend on $\mathbf{x}_j$ and a similar calculation as above shows that

$$\psi^{\text{weak}}_{i\to j}(t,\mathbf{x}_j,\mathbf{z}_i) \approx \psi^{\text{weak}}_i(t,\mathbf{z}_i) := \mathcal{N}(\widehat{\mathbf{p}}_i(t),\mathbf{Q}^p_i(t)/u), \tag{74}$$

where $\widehat{\mathbf{p}}_i(t)$ and $\mathbf{Q}^p_i(t)$ are defined in (70). Substituting the Gaussian approximation (74) into (19), and then using the definitions in (61a) and (63a), one can show that the marginal distribution $p_{i\to j}(t,\mathbf{x}_j)$ in (19) is equal to the marginal distribution of $p_{i\to j}(t,\mathbf{x}_{\alpha(i)},\mathbf{z}_i)$ in (33). Therefore, the message $\Delta_{i\to j}(t,\mathbf{x}_j)$ in (18) can be written as (32) for all strong edges (i,j).

We now turn to the variable node update (20) which we partition as

$$\Delta_{i\leftarrow j}(t,\mathbf{x}_j) = \Delta^{\text{weak}}_{i\leftarrow j}(t,\mathbf{x}_j) + \Delta^{\text{strong}}_{i\leftarrow j}(t,\mathbf{x}_j), \tag{75}$$

where

$$\Delta^{\text{strong}}_{i\leftarrow j}(t{+}1,\mathbf{x}_j) = \sum_{\ell\neq i\,:\, j\,\in\,\alpha(\ell)} \Delta_{\ell\to j}(t,\mathbf{x}_j) \tag{76a}$$

$$\Delta^{\text{weak}}_{i\leftarrow j}(t{+}1,\mathbf{x}_j) = \sum_{\ell\neq i\,:\, j\,\in\,\beta(\ell)} \Delta_{\ell\to j}(t,\mathbf{x}_j). \tag{76b}$$

Substituting the approximation (73) into (76b) gives

$$\Delta^{\text{weak}}_{i\leftarrow j}(t{+}1,\mathbf{x}_j) \approx -\frac{1}{2}\|\widehat{\mathbf{r}}_{i\leftarrow j}(t) - \mathbf{x}_j\|^2_{\mathbf{Q}^r_{i\leftarrow j}(t)}, \tag{77}$$

where

$$\mathbf{Q}^{-r}_{i\leftarrow j}(t) = \sum_{\ell\neq i}\mathbf{A}^*_{\ell j}\mathbf{Q}^s_\ell(t)\mathbf{A}_{\ell j} \tag{78a}$$

$$\begin{aligned}\widehat{\mathbf{r}}_{i\leftarrow j}(t) &= \mathbf{Q}^r_{i\leftarrow j}(t)\left[\sum_{\ell\neq i}\mathbf{A}^*_{\ell j}\widehat{\mathbf{s}}_\ell(t) + \mathbf{A}^*_{\ell j}\mathbf{Q}^s_\ell(t)\mathbf{A}_{\ell j}\widehat{\mathbf{x}}_j(t)\right] \\ &= \widehat{\mathbf{x}}(t) + \mathbf{Q}^r_{i\leftarrow j}(t)\sum_{\ell\neq i}\mathbf{A}^*_{\ell j}\widehat{\mathbf{s}}_\ell(t). \end{aligned} \tag{78b}$$

We again consider the case of a weak edge separately from a strong edge. When (i,j) is weak edge, $j \notin \alpha(i)$, so that $\Delta^{\text{strong}}_{i\leftarrow j}(t{+}1,\mathbf{x}_j)$ in (76a) does not depend on i. Combining (75) and (77), we see that

$$\Delta_{i\leftarrow j}(t{+}1,\mathbf{x}_j) \approx H^x_j(t,\mathbf{x}_j,\widehat{\mathbf{r}}_{i\leftarrow j}(t),\mathbf{Q}^r_{i\leftarrow j}(t)), \tag{79}$$

where $H^x_j(\cdot)$ is defined in (26). Also, comparing (38) with (78), we have that

$$\mathbf{Q}^{-r}_{i\leftarrow j}(t) \approx \mathbf{Q}^{-r}_j(t) \tag{80a}$$

$$\widehat{\mathbf{r}}_{i\leftarrow j}(t) \approx \widehat{\mathbf{r}}_j(t) - \mathbf{Q}^r_j(t)\mathbf{A}^*_{ij}\widehat{\mathbf{s}}_i(t). \tag{80b}$$

Substituting (80) into (79) we get

$$\Delta_{i\leftarrow j}(t{+}1,\mathbf{x}_j) \approx H^x_j(t,\mathbf{x}_j,\widehat{\mathbf{r}}_j(t) - \mathbf{Q}^r_j(t)\mathbf{A}^*_{ij}\widehat{\mathbf{s}}_i(t),\mathbf{Q}^r_j(t)). \tag{81}$$

A similar set of calculations shows that $\Delta_j(t{+}1,\mathbf{x}_j)$ in (21) can be approximated as

$$\Delta_j(t{+}1,\mathbf{x}_j) \approx H^x_j(t,\mathbf{x}_j,\widehat{\mathbf{r}}_j(t),\mathbf{Q}^r_j(t)). \tag{82}$$

Thus, the definitions of $\widehat{\mathbf{x}}_j(t{+}1)$ and $\mathbf{Q}^x_j(t{+}1)$ in (64) agree with (28).

Finally, define

$$\Gamma_j(t,\widehat{\mathbf{r}}_j) := \mathbb{E}\left[\mathbf{x}_j; H^x_j(t,\cdot,\widehat{\mathbf{r}}_j,\mathbf{Q}^r_j(t{-}1))\right], \tag{83}$$

where again we are using the notation (14) and $H^x_j(\cdot)$ is defined in (26). It follows from (81), (82) and (64) that

$$\begin{aligned}\widehat{\mathbf{x}}_j(t{+}1) &\approx \Gamma_j(t,\widehat{\mathbf{r}}_j(t)) \\ \widehat{\mathbf{x}}_{i\leftarrow j}(t{+}1) &\approx \Gamma_j(t,\widehat{\mathbf{r}}_j(t) - \mathbf{Q}^r_j(t)\mathbf{A}^*_{ij}\widehat{\mathbf{s}}_i(t)) \\ &\approx \widehat{\mathbf{x}}_j(t) - \frac{\partial\Gamma_j(t,\widehat{\mathbf{r}}_j(t))}{\partial\widehat{\mathbf{r}}_j}\mathbf{Q}^r_j(t)\mathbf{A}^*_{ij}\widehat{\mathbf{s}}_i(t). \end{aligned} \tag{84}$$

From the definition (83), Lemma 1 shows that

$$\frac{\partial\Gamma_j(t,\widehat{\mathbf{r}}_j(t))}{\partial\widehat{\mathbf{r}}_j} \approx \mathbf{Q}^x(t)\mathbf{Q}^{-r}(t), \tag{85}$$

and hence, from (84),

$$\widehat{\mathbf{x}}_{i\leftarrow j}(t{+}1) \approx \widehat{\mathbf{x}}_j(t{+}1) - \mathbf{Q}^x(t{+}1)\mathbf{A}^*_{ij}\widehat{\mathbf{s}}_i(t). \tag{86}$$

Substituting (86) into (70) we obtain

$$\begin{aligned}\widehat{\mathbf{p}}_i(t) &\approx \sum_{j\,\in\,\beta(i)}\mathbf{A}_{ij}\widehat{\mathbf{x}}_j(t) - \sum_{j\,\in\,\beta(i)}\mathbf{A}_{ij}\mathbf{Q}^x(t)\mathbf{A}^*_{ij}\widehat{\mathbf{s}}_i(t{-}1) \\ &\approx \mathbf{z}_i(t) - \mathbf{Q}^p(t)\widehat{\mathbf{s}}_i(t{-}1), \end{aligned}$$

which agrees with the definition in (29).

Appendix B
Derivation of MS-HyGAMP

The derivation of MS-HyGAMP is similar to the derivation of SP-HyGAMP in Appendix A.

A. Preliminary Lemma

We begin by stating the analogue to Lemma 1. For each $\mathbf{v}$, let

$$\widehat{\mathbf{w}}(\mathbf{v}) := \arg\max_{\mathbf{w}} H(\mathbf{w},\mathbf{v}), \tag{87a}$$

$$G(\mathbf{v}) := H(\widehat{\mathbf{w}}(\mathbf{v}),\mathbf{v}) = \max_{\mathbf{w}} H(\mathbf{w},\mathbf{v}), \tag{87b}$$

where $H(\mathbf{w},\mathbf{v})$ was given in (58).

Lemma 2: Assume the maximization in (87) exists and is unique and twice differentiable. Then,

$$\frac{\partial G(\mathbf{v})}{\partial\mathbf{v}} = \mathbf{Q}^{-v}(\widehat{\mathbf{w}}(\mathbf{v}) - \mathbf{v}), \tag{88a}$$

$$\frac{\partial\widehat{\mathbf{w}}(\mathbf{v})}{\partial\mathbf{v}} = -\mathbf{D}^{-1}\mathbf{Q}^{-v}, \tag{88b}$$

$$\frac{\partial^2 G(\mathbf{v})}{\partial\mathbf{v}^2} = -\mathbf{Q}^{-v} - \mathbf{Q}^{-v}\mathbf{D}^{-1}\mathbf{Q}^{-v}, \tag{88c}$$

where

$$\mathbf{D} = \left. \frac{\partial^2 H(\mathbf{w},\mathbf{v})}{\partial \mathbf{w}^2} \right|_{\mathbf{w}=\widehat{\mathbf{w}}(\mathbf{v})}.$$

Proof: Since $\mathbf{w} = \widehat{\mathbf{w}}(\mathbf{v})$ is a maximizer of $H(\mathbf{w},\mathbf{v})$,

$$\frac{\partial H(\widehat{\mathbf{w}}(\mathbf{v}),\mathbf{v})}{\partial \mathbf{w}} = 0. \tag{89}$$

Therefore, (88a) follows from

$$\begin{aligned}\frac{\partial G(\mathbf{v})}{\partial \mathbf{v}} &= \frac{\partial H(\widehat{\mathbf{w}}(\mathbf{v}),\mathbf{v})}{\partial \mathbf{w}\frac{\partial \widehat{\mathbf{w}}(\mathbf{v})}{\partial \mathbf{v}} + \frac{\partial H(\widehat{\mathbf{w}}(\mathbf{v}),\mathbf{v})}{\partial \mathbf{v}}} \\ &= \frac{\partial H(\widehat{\mathbf{w}}(\mathbf{v}),\mathbf{v})}{\partial \mathbf{v}} = \mathbf{Q}^{-v}(\widehat{\mathbf{w}}(\mathbf{v}) - \mathbf{v}),\end{aligned}$$

where the last step is a result of the form of $H(\cdot)$ in (58). The form of $H(\cdot)$ in (58) also shows that for all $\mathbf{w}$ and $\mathbf{v}$

$$\frac{\partial^2 H(\mathbf{w},\mathbf{v})}{\partial \mathbf{w}\partial \mathbf{v}} = \mathbf{Q}^{-v}.$$

Taking the derivative of (89),

$$\frac{\partial^2 H(\widehat{\mathbf{w}},\mathbf{v})}{\partial \mathbf{w}\partial \mathbf{v}} + \frac{\partial^2 H(\widehat{\mathbf{w}},\mathbf{v})}{\partial \mathbf{w}^2}\frac{\partial \widehat{\mathbf{w}}(\mathbf{v})}{\partial \mathbf{v}} = 0,$$

which implies that

$$\frac{\partial \widehat{\mathbf{w}}(\mathbf{v})}{\partial \mathbf{v}} = -\mathbf{D}^{-1}\mathbf{Q}^{-v},$$

which proves (88b). Finally, taking the second derivative of (88a) along with (88b) shows (88c). ■

B. MS-HyGAMP Approximation

Similar to the SPA derivation, we first partition the function $H_{i\to j}(\cdot)$ in (16) as in (60). We can also partition the maximization (17) as

$$\Delta_{i\to j}(t,\mathbf{x}_j) = \max_{\mathbf{z}_i} \left[\Delta^{\text{strong}}_{i\to j}(t,\mathbf{x}_j,\mathbf{z}_i) + \Delta^{\text{weak}}_{i\to j}(t,\mathbf{x}_j,\mathbf{z}_i)\right], \tag{90}$$

where

$$\Delta^{\text{strong}}_{i\to j}(t,\mathbf{x}_j,\mathbf{z}_i) := \max_{\mathbf{x}_{\alpha(i)\setminus j}} H^{\text{strong}}_{i\to j}(t,\mathbf{x}_{\alpha(i)},\mathbf{z}_i), \tag{91a}$$

$$\Delta^{\text{weak}}_{i\to j}(t,\mathbf{z}_i,\mathbf{x}_j) := \max_{\substack{\mathbf{x}_{\beta(i)\setminus j}\\ \mathbf{z}_i=\mathbf{A}_i\mathbf{x}}} H^{\text{weak}}_{i\to j}(t,\mathbf{x}_{\beta(i)}), \tag{91b}$$

with the maximization in (91a) being over all $\mathbf{x}_r$ with $r \in \alpha(i) \setminus j$; and the maximization in (91b) over all $\mathbf{x}_r$ with $r \in \beta(i) \setminus j$ subject to $\mathbf{z}_i = \mathbf{A}_i\mathbf{x}$. The partitioning (90) is valid since the strong and weak edges are distinct. This insures that for all $r \in \delta(i)$, either $r \in \alpha(i)$ or $r \in \beta(i)$, but not both.

The HyGAMP approximation applies to the weak term (91b). For any j and all weak edges (i,j), define:

$$\widehat{\mathbf{x}}_j(t) := \arg\max_{\mathbf{x}_j} \Delta_j(t,\mathbf{x}_j), \tag{92a}$$

$$\widehat{\mathbf{x}}_{i\leftarrow j}(t) := \arg\max_{\mathbf{x}_j} \Delta_{i\leftarrow j}(t,\mathbf{x}_j), \tag{92b}$$

$$\mathbf{Q}^{-x}_j(t) := -\frac{\partial^2}{\partial \mathbf{x}_j^2} \left.\Delta_j(t,\mathbf{x}_j)\right|_{\mathbf{x}_j=\widehat{\mathbf{x}}_j(t)}, \tag{92c}$$

$$\mathbf{Q}^{-x}_{i\leftarrow j}(t) := -\frac{\partial^2}{\partial \mathbf{x}_j^2} \left.\Delta_{i\leftarrow j}(t,\mathbf{x}_j)\right|_{\mathbf{x}_j=\widehat{\mathbf{x}}_{i\leftarrow j}(t)}, \tag{92d}$$

which are the maximum and Hessian of the incoming weak messages. Since the assumption of the HyGAMP algorithm is that $\mathbf{A}_{ir}$ is small for all weak edges (i,r), the values of $\mathbf{x}_r$ in the maximization (91b) will be close to $\widehat{\mathbf{x}}_{i\leftarrow r}(t)$. So, for all weak edges, (i,r), we can approximate each term $\Delta_{i\leftarrow r}(t,\mathbf{x}_r)$ in (61b) with the second-order approximation

$$\Delta_{i\leftarrow r}(t,\mathbf{x}_r) \approx \Delta_{i\leftarrow r}(t,\widehat{x}_{i\leftarrow r}(t)) - \frac{1}{2}\|\mathbf{x}_r - \widehat{\mathbf{x}}_{i\leftarrow r}(t)\|^2_{\mathbf{Q}^x_j(t)}, \tag{93}$$

where we have additionally made the approximation $\mathbf{Q}^x_{i\leftarrow r}(t) \approx \mathbf{Q}^x_r(t)$ for all i. Substituting (93) into (61b), the maximization (91b) reduces to

$$\begin{aligned}&\Delta^{\text{weak}}_{i\to j}(t,\mathbf{x}_j,\mathbf{z}_i)\\ &\approx \text{const} - \max_{\substack{\mathbf{x}_{\beta(i)\setminus j}\\ \mathbf{z}_i=\mathbf{A}_i\mathbf{x}}} \left[\frac{1}{2}\sum_{r\,\in\,\{\beta(i)\setminus j\}} \|\mathbf{x}_r - \widehat{\mathbf{x}}_{i\leftarrow r}(t)\|^2_{\mathbf{Q}^x_r(t)}\right],\end{aligned} \tag{94}$$

where the constant term does not depend on $\mathbf{x}_j$ or $\mathbf{z}_i$.

To proceed, we need to consider two cases separately: when $j \in \beta(i)$ and when $j \notin \beta(i)$. First consider the case when $j \in \beta(i)$. That is, (i,j) is a weak edge. In this case, a standard least-squares calculation shows that (94) reduces to

$$\begin{aligned}&\Delta^{\text{weak}}_{i\to j}(t,\mathbf{x}_j,\mathbf{z}_i)\\ &\approx \text{const} - \frac{1}{2}\|\mathbf{z}_i - \mathbf{A}_{ij}\mathbf{x}_{i\leftarrow j}(t) - \widehat{\mathbf{p}}_{i\leftarrow j}(t)\|^2_{\mathbf{Q}^p_{i\to j}(t)},\end{aligned} \tag{95}$$

where $\widehat{\mathbf{p}}_{i\to j}(t)$ and $\mathbf{Q}^p_{i\to j}(t)$ are given in (67). Also, when $j \in \beta(i)$, the assumption that $\alpha(i)$ and $\beta(i)$ are disjoint implies that $j \notin \alpha(i)$. In this case, $\Delta^{\text{strong}}_{i\to j}(t,\mathbf{x}_j,\mathbf{z}_i)$ in (91a) with the objective function (61a) will not depend on $\mathbf{x}_j$, so we can write

$$\begin{aligned}\Delta^{\text{strong}}_{i\to j}(t,\mathbf{x}_j,\mathbf{z}_i) &= \Delta^{\text{strong}}_i(t,\mathbf{z}_i)\\ &:= \max_{\mathbf{x}}\left[f_i(\mathbf{x}_{\alpha(i)},\mathbf{z}_i) + \sum_{r\,\in\,\alpha(i)} \Delta_{i\leftarrow r}(t,\mathbf{x}_r)\right],\end{aligned} \tag{96}$$

where the maximization is over all $\mathbf{x}_r$ for $r \in \alpha(i)$. Combining (90), (95) and (96), we can write that, for all weak edges (i,j),

$$\Delta_{i\to j}(t,\mathbf{x}_j) \approx G_i(t,\widehat{\mathbf{p}}_{i\to j}(t) + \mathbf{A}_{ij}\mathbf{x}_j), \tag{97}$$

where

$$G_i(t,\widehat{\mathbf{p}}_i) := \max_{\mathbf{x}_{\alpha(i)},\mathbf{z}_i} H^z_i(t,\mathbf{x}_{\alpha(i)},\mathbf{z}_i,\widehat{\mathbf{p}}_i,\mathbf{Q}^p_i(t)) \tag{98}$$

and $H^z_i(\cdot)$ is defined in (34a).

Now define $\widehat{\mathbf{p}}_i(t)$ and $\mathbf{Q}_i^p(t)$ as in (70). Using (71), neglecting terms of order $O(\|\mathbf{A}_{ij}\|^2)$, and taking the approximation that $\widehat{\mathbf{x}}_{i\leftarrow j}(t) \approx \widehat{\mathbf{x}}_j(t)$, (97) can be further approximated as

$$\Delta_{i\to j}(t,\mathbf{x}_j) \approx G_i(t, \widehat{\mathbf{p}}_i(t) + \mathbf{A}_{ij}(\mathbf{x}_j - \widehat{\mathbf{x}}_j(t))),$$

similar to (68). Now, similar to (72), let

$$\widehat{\mathbf{s}}_i(t) = \frac{\partial}{\partial \widehat{\mathbf{p}}} G_i(t, \widehat{\mathbf{p}}_i(t)), \tag{99a}$$

$$\mathbf{Q}_i^{-s}(t) = -\frac{\partial^2}{\partial \widehat{\mathbf{p}}^2} G_i(t, \widehat{\mathbf{p}}_i(t)). \tag{99b}$$

Based on the definition of $G_i(\cdot)$ in (98) with $H_i^z(\cdot)$ defined in (34a), one can apply Lemma 2 to show that (99) agrees with (37). Using a similar approximation as in the derivation of the SPA-HyGAMP, one can then obtain the quadratic approximation in (73) for $\Delta_{i\to j}(t,\mathbf{x}_j)$ for all weak edges (i,j).

Next consider the case when $j \notin \beta(i)$ so that (i,j) is a strong edge. In this case, $\Delta_{i\to j}^{\text{weak}}(t,\mathbf{x}_j,\mathbf{z}_i)$ in (94) does not depend on $\mathbf{x}_j$, so we can write

$$\Delta_{i\to j}^{\text{weak}}(t,\mathbf{x}_j,\mathbf{z}_i) \approx \text{const} + \Delta_i^{\text{weak}}(t,\mathbf{z}_i), \tag{100}$$

where

$$\Delta_i^{\text{weak}}(t,\mathbf{z}_i) := \max_{\mathbf{x}\,:\,\mathbf{z}_i = \mathbf{A}_i\mathbf{x}} H_{i\to j}^{\text{weak}}(t,\mathbf{x}_{\beta(i)}), \tag{101}$$

with the maximization being over $\mathbf{x}$ such that $\mathbf{z}_i = \mathbf{A}_i\mathbf{x}$. Using a similar least-squares calculation as above, $\Delta_i^{\text{weak}}(t,\mathbf{z}_i)$ is given by

$$\Delta_i^{\text{weak}}(t,\mathbf{z}_i) := -\frac{1}{2}\|\mathbf{z}_i - \widehat{\mathbf{p}}_i(t)\|^2_{\mathbf{Q}_i^p(t)}, \tag{102}$$

and $\widehat{\mathbf{p}}_i(t)$ and $\mathbf{Q}_i^p(t)$ are defined in (70). Combining (90), (100) and (102), we can write that, for all strong edges (i,j),

$$\Delta_{i\to j}(t,\mathbf{x}_j) \approx \max_{\mathbf{z}_i}\left[\Delta_{i\to j}^{\text{strong}}(t,\mathbf{x}_j,\mathbf{z}_i) - \frac{1}{2}\|\mathbf{z}_i - \widehat{\mathbf{p}}_i(t)\|^2_{\mathbf{Q}_i^p(t)}\right] + \text{const} \tag{103}$$

From (61a) and (91a), we see that (103) agrees with the factor node update (31) for the strong edges.

We now turn to the variable update steps of the MSA. Since this step is identical to the SPA, one can follow the derivation in Appendix A to show that $\Delta_{i\leftarrow j}(t+1,\mathbf{x}_j)$ and $\Delta_i(t+1,\mathbf{x}_j)$ are given by (81) and (82), respectively and $\widehat{\mathbf{r}}_j(t)$ and $\mathbf{Q}_j^r(t)$ are given in (38). Also, the definitions of $\widehat{\mathbf{x}}_j(t)$ and $\mathbf{Q}_j^x(t)$ in (92) are consistent with (27).

Also, if we let

$$\Gamma_j(t,\widehat{\mathbf{r}}_j) := \arg\max_{\mathbf{x}_j} H_j^x(t,\mathbf{x}_j,\widehat{\mathbf{r}}_j,\mathbf{Q}_j^r(t)),$$

it follows from (92), (81), and (82) that

$$\begin{aligned}\widehat{\mathbf{x}}_j(t+1) &\approx \Gamma_j(t,\widehat{\mathbf{r}}_j(t))\\ \widehat{\mathbf{x}}_{i\leftarrow j}(t+1) &\approx \Gamma_j(t,\widehat{\mathbf{r}}_j(t) - \mathbf{Q}_j^r(t)\mathbf{A}_{ij}^*\widehat{\mathbf{s}}_i(t))\\ &\approx \widehat{\mathbf{x}}_j(t) - \frac{\partial\Gamma_j(t,\widehat{\mathbf{r}}_j(t))}{\partial\widehat{\mathbf{r}}_j}\mathbf{Q}_j^r(t)\mathbf{A}_{ij}^*\widehat{\mathbf{s}}_i(t).\end{aligned} \tag{104}$$

It can be shown from Lemma 2 that

$$\begin{aligned}\frac{\partial\Gamma_j(t,\widehat{\mathbf{r}}_j(t))}{\partial\widehat{\mathbf{r}}_j} &= -\left[\frac{\partial^2}{\partial\mathbf{x}_j^2} H_j^x(t,\widehat{\mathbf{x}}_j(t+1),\widehat{\mathbf{r}}_j(t))\right]^{-1}\mathbf{Q}^{-r}(t)\\ &\approx \mathbf{Q}^x(t)\mathbf{Q}^{-r}(t),\end{aligned}$$

and hence, from (104),

$$\widehat{\mathbf{x}}_{i\leftarrow j}(t+1) \approx \widehat{\mathbf{x}}_j(t+1) - \mathbf{Q}^x(t+1)\mathbf{A}_{ij}^*\widehat{\mathbf{s}}_i(t). \tag{105}$$

The proof now follows identically to the derivation of the SPA-HyGAMP.

REFERENCES

[1] S. Rangan, A. K. Fletcher, V. K. Goyal, and P. Schniter, "Hybrid generalized approximation message passing with applications to structured sparsity," in *Proc. IEEE Int. Symp. Inf. Theory*, Cambridge, MA, USA, Jul. 2012, pp. 1241–1245.

[2] B. J. Frey, *Graphical Models for Machine Learning and Digital Communication*. Cambridge, MA, USA: MIT Press, 1998.

[3] M. J. Wainwright and M. I. Jordan, "Graphical models, exponential families, and variational inference," *Found. Trends Mach. Learn.*, vol. 1, no. 1–2, pp. 1–305, 2008.

[4] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*. Cambridge, MA, USA: MIT Press, 2009.

[5] D. L. Donoho, A. Maleki, and A. Montanari, "Message-passing algorithms for compressed sensing," *Proc. Nat. Acad. Sci.*, vol. 106, no. 45, pp. 18 914–18 919, Nov. 2009.

[6] D. L. Donoho, A. Maleki, and A. Montanari, "Message passing algorithms for compressed sensing I: Motivation and construction," in *Proc. Inf. Theory Workshop*, Jan. 2010, pp. 1–5.

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

[8] S. Rangan, "Generalized approximate message passing for estimation with random linear mixing," in *Proc. IEEE Int. Symp. Inf. Theory*, Saint Petersburg, Russia, Jul./Aug. 2011, pp. 2174–2178.

[9] J. Parker, P. Schniter, and V. Cevher, "Bilinear generalized approximate message passing—Part I: Derivation," *IEEE Trans. Signal Process.*, vol. 62, no. 22, pp. 5839–5853, Nov. 2013.

[10] J. Parker, P. Schniter, and V. Cevher, "Bilinear generalized approximate message passing—Part II: Applications," *IEEE Trans. Signal Process.*, vol. 62, no. 22, pp. 5854–5867, Nov. 2013.

[11] J. Parker and P. Schniter, "Parametric bilinear generalized approximate message passing," *IEEE J. Sel. Topics Signal Process.*, vol. 10, no. 4, pp. 1–14, Jun. 2016.

[12] P. Schniter, "Turbo reconstruction of structured sparse signals," in *Proc. Conf. Inf. Sci. Syst.*, Princeton, NJ, USA, Mar. 2010, pp. 1–6.

[13] P. Schniter, "A message-passing receiver for BICM-OFDM over unknown clustered-sparse channels," *IEEE J. Sel. Topics Signal Process.*, vol. 5, no. 8, pp. 1462–1474, Dec. 2011.

[14] S. Som and P. Schniter, "Compressive imaging using approximate message passing and a Markov-tree prior," *IEEE Trans. Signal Process.*, vol. 60, no. 7, pp. 3439–3448, Jul. 2012.

[15] J. Ziniel and P. Schniter, "Efficient high-dimensional inference in the multiple measurement vector problem," *IEEE Trans. Signal Process.*, vol. 61, no. 2, pp. 340–354, Jan. 2013.

[16] J. Ziniel and P. Schniter, "Dynamic compressive sensing of time-varying signals via approximate message passing," *IEEE Trans. Signal Process.*, vol. 61, no. 21, pp. 5270–5284, Nov. 2013.

[17] M. Nassar, P. Schniter, and B. Evans, "A factor-graph approach to joint OFDM channel estimation and decoding in impulsive noise environments," *IEEE Trans. Signal Process.*, vol. 62, no. 6, pp. 1576–1589, Mar. 2014.

[18] J. Vila, P. Schniter, and J. Meola, "Hyperspectral image unmixing via turbo bilinear approximate message passing," *IEEE Trans. Comput. Imag.*, vol. 1, no. 3, pp. 143–158, Sep. 2015.

[19] E. W. Tramel, A. Drémeau, and F. Krzakala, "Approximated message passing with restricted Boltzmann machine priors," *J. Statist. Mech., Theory Exp.*, vol. 2016, no. 7, 2016, Art. no. 073401.

[20] G. Caire, A. Tulino, and E. Biglieri, "Iterative multiuser joint detection and parameter estimation: A factor-graph approach," in *Proc. IEEE Inf. Theory Workshop*, Cairns, QLD, Australia, Sep. 2001, pp. 36–38.

[21] S. Rangan and R. Madan, "Belief propagation methods for intercell interference coordination," in *Proc. IEEE INFOCOM*, 2011, pp. 2543–2551.

[22] A. Montanari, "Graphical model concepts in compressed sensing," in *Compressed Sensing: Theory and Applications*, Y. C. Eldar and G. Kutyniok, Eds. Cambridge, U.K.: Cambridge Univ. Press, Jun. 2012, pp. 394–438.

[23] S. Wang, Y. Li, and J. Wang, "Multiuser detection in massive spatial modulation MIMO with low-resolution ADCs," *IEEE Trans. Wireless Commun.*, vol. 14, no. 4, pp. 2156–2168, Apr. 2015.

[24] S. Wang and L. Zhang, "Signal processing in massive MIMO with IQ imbalances and low-resolution ADCs," *IEEE Trans. Wireless Commun.*, vol. 15, no. 12, pp. 8298–8312, Dec. 2016.

[25] A. K. Fletcher and S. Rangan, "Scalable inference for neuronal connectivity from calcium imaging," in *Proc. Neural Inf. Process. Syst.*, 2014, pp. 2843–2851.

[26] A. K. Fletcher, J. Viventi, and S. Rangan, "Neural mass spatio-temporal modeling from high-density electrode array recordings," in *Proc. Inf. Theory Appl. Workshop*, 2015, pp. 319–321.

[27] Z. Utkovski, O. Simeone, T. Dimitrova, and P. Popovski, "Random access in c-ran for user activity detection with limited-capacity fronthaul," *IEEE Signal Process. Lett.*, vol. 24, no. 1, pp. 17–21, Jan. 2017.

[28] A. E. Alaoui *et al.*, "Decoding from pooled data: Phase transitions of message passing," arXiv: 1702.02279, Feb. 2017.

[29] A. Dembo and O. Zeitouni, *Large Deviations Techniques and Applications*. New York, NY, USA: Springer-Verlag, 1998.

[30] C. M. Bishop, *Pattern Recognition and Machine Learning (ser. Information Science and Statistics)*. New York, NY, USA: Springer-Verlag, 2006.

[31] B. Krishnapuram, L. Carin, M. Figueiredo, and A. Hartemink, "Sparse multinomial logistic regression: Fast algorithms and generalization bounds," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 6, pp. 957–968, Jun. 2005.

[32] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Mateo, CA, USA: Morgan Kaufmann, 1988.

[33] J. S. Yedidia, W. T. Freeman, and Y. Weiss, "Understanding belief propagation and its generalizations," in *Exploring Artificial Intelligence in the New Millennium*. San Francisco, CA, USA: Morgan Kaufmann, 2003, pp. 239–269.

[34] D. M. Malioutov, J. K. Johnson, and A. S. Willsky, "Walk-sums and belief propagation in Gaussian graphical models," *J. Mach. Learn. Res.*, vol. 7, pp. 2031–2064, Oct. 2006.

[35] G. Elidan, I. McGraw, and D. Koller, "Residual belief propagation: Informed scheduling for asynchronous message passing," in *Proc. Conf. Uncertainty in AI*, Boston, MA, USA, Jul. 2006, pp. 1123–1132.

[36] A. Manoel, F. Krzakala, E. W. Tramel, and L. Zdeborová, "Swept approximate message passing for sparse estimation," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1123–1132.

[37] M. Yuan and Y. Lin, "Model selection and estimation in regression with grouped variables," *J. Roy. Statist. Soc.*, vol. 68, pp. 49–67, 2006.

[38] P. Zhao, G. Rocha, and B. Yu, "The composite absolute penalties family for grouped and hierarchical variable selection," *Ann. Statist.*, vol. 37, no. 6, pp. 3468–3497, 2009.

[39] S. Rangan, A. K. Fletcher, V. K. Goyal, E. Byrne, and P. Schniter, "Hybrid approximate message passing," arXiv:1111.2581, Mar. 2017.

[40] Y. Kim, J. Kim, and Y. Kim, "Blockwise sparse regression," *Statist. Sinica*, vol. 16, pp. 375–390, 2006.

[41] L. Meier, S. van de Geer, and P. Bühlmann, "The group Lasso for logistic regression," *J. Roy. Statist. Soc., B*, vol. 70, no. 1, pp. 53–71, 2008.

[42] A. C. Lozano, G. Świrszcz, and N. Abe, "Group orthogonal matching pursuit for logistic regression," *J. Mach. Learn. Res.*, vol. 15, pp. 452–460, 2011.

[43] J. Ziniel, P. Schniter, and P. Sederberg, "Binary linear classification and feature selection via generalized approximate message passing," *IEEE Trans. Signal Process.*, vol. 63, no. 8, pp. 2020–2032, Apr. 2015.

[44] N. S. Rao, R. D. Nowak, S. J. Wright, and N. G. Kingsbury, "Convex approaches to model wavelet sparsity patterns," in *Proc. IEEE Int. Conf. Image Process.*, 2011, pp. 1917–1920.

[45] V. Cevher, P. Indyk, L. Carin, and R. Baraniuk, "Sparse signal recovery and acquisition with graphical models," *IEEE Signal Process. Mag.*, vol. 27, no. 6, pp. 92–103, Nov. 2010.

[46] J. Kim, W. Chang, B. Jung, D. Baron, and J. C. Ye, "Belief propagation for joint sparse recovery," arXiv:1102.3289, Feb. 2011.

[47] A. C. Lozano, G. Świrszcz, and N. Abe, "Group orthogonal matching pursuit for variable selection and prediction," in *Proc. Neural Inf. Process. Syst.*, Vancouver, BC, Canada, Dec. 2009, pp. 1150–1158.

[48] S. J. Wright, R. D. Nowak, and M. Figueiredo, "Sparse reconstruction by separable approximation," *IEEE Trans. Signal Process.*, vol. 57, no. 7, pp. 2479–2493, Jul. 2009.

[49] M. Figueiredo, S. J. Wright, and R. D. Nowak, "Gradient projection for sparse reconstruction: Application to compressed sensing and other inverse problems," *IEEE J. Sel. Topics Signal Process.*, vol. 1, no. 4, pp. 586–597, Dec. 2007.

[50] S. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinvesky, "An interior point method for large-scale ℓ_1-regularized least squares," *IEEE J. Sel. Topics Signal Process.*, vol. 1, no. 4, pp. 606–617, Dec. 2007.

[51] I. Daubechies, M. Defrise, and C. D. Mol, "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint," *Commun. Pure Appl. Math.*, vol. 57, no. 11, pp. 1413–1457, Nov. 2004.

[52] S. Rangan *et al.*, "Generalized approximate message passing," *SourceForge.net Project GAMPmatlab*. (2011). [Online]. Available: http://gampmatlab.sourceforge.net/

[53] J. Friedman, T. Hastie, and R. Tibshirani, "Regularization paths for generalized linear models via coordinate descent," *J. Statist. Softw.*, vol. 33, no. 1, pp. 1–22, 2010.

[54] E. M. Byrne, "Sparse multinomial logistic regression via approximate message passing," Master's thesis, Dept. Elect. Comput. Eng., Ohio State Univ., Columbus, OH, USA, Jul. 2015.

[55] G. C. Cawley, N. L. C. Talbot, and M. Girolami, "Sparse multinomial logistic regression via Bayesian L1 regularisation," in *Proc. Neural Inf. Process. Syst.*, 2007, pp. 209–216.

[56] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[57] J. P. Vila and P. Schniter, "Expectation-maximization Gaussian-mixture approximate message passing," *IEEE Trans. Signal Process.*, vol. 61, no. 19, pp. 4658–4672, Oct. 2013.

[58] A. Mousavi, A. Maleki, and R. G. Baraniuk, "Parameterless, optimal approximate message passing," arXiv:1311.0035, 2013.

[59] E. M. Byrne and P. Schniter, "Sparse multinomial logistic regression via approximate message passing," *IEEE Trans. Signal Process.*, vol. 64, no. 21, pp. 5485–5498, Nov. 2016.

Sundeep Rangan (S'94–M'98–SM'13–F'16) received the B.A.Sc. degree from the University of Waterloo, Waterloo, ON, Canada, and the M.Sc. and Ph.D. degrees from the University of California, Berkeley, CA, USA, all in electrical engineering. Since 2010, he has been on the faculty of the Department of Electronics and Communication Engineering, New York University Polytechnic School of Engineering.

Alyson K. Fletcher (S'03–M'04) received the B.S. degree in mathematics from the University of Iowa, Iowa City, IA, USA, and the M.S. degree in mathematics and electrical engineering (EE) and the Ph.D. degree in EE, both from the University of California at Berkeley, Berkeley, CA, USA. Since 2016, she has been on the faculty of the Departments of Statistics, Mathematics, EE, and Computer Science at the University of California, Los Angeles.

Vivek K. Goyal (S'92–M'98–SM'03–F'14) received the B.S. degree in mathematics and the B.S.E. degree in electrical engineering (EE) from the University of Iowa, Iowa City, IA, USA, and the M.S. and Ph.D. degrees in EE from the University of California at Berkeley, Berkeley, CA, USA. Since 2014, he has been on the faculty of the Department of Electronics and Communication Engineering, Boston University.

Evan Byrne (S'17) received the B.S. and M.S. degrees in electrical and computer engineering (ECE) from Ohio State University, Columbus, OH, USA, where he is currently working toward the Ph.D. degree in ECE.

Philip Schniter (S'92–M'93–SM'05–F'14) received the B.S. and M.S. degrees from the University of Illinois, Urbana-Champaign, IL, USA, and the Ph.D. degree from Cornell University, Ithaca, NY, USA, all in electrical and computer engineering. Since 2000, he has been on the faculty of the Department of Electrical and Computer Engineering, The Ohio State University.