Boolean Matrix Factorization and Completion via Message Passing

Siamak Ravanbakhsh and Russell Greiner

Department of Computing Science, University of Alberta, Edmonton, AB, Canada

***Abstract*—Boolean factor analysis is the task of decomposing a Binary matrix to the Boolean product of two binary factors. This unsupervised data-analysis approach is desirable due to its interpretability, but hard to perform due its $\mathbb{NP}$-hardness. A closely related problem is low-rank Boolean matrix completion from noisy observations. We treat these problems as maximum a posteriori inference problems, and present message passing solutions that scale linearly with the number of observations and factors. Our empirical study demonstrates that message passing is able to recover low-rank Boolean matrices, in the boundaries of theoretically possible recovery and outperform existing techniques in real-world applications, such as large-scale binary valued collaborative filtering tasks.**

A large body of problems in machine learning, communication theory and combinatorial optimization involve the product form $\mathbf{Z} = \mathbf{X} \odot \mathbf{Y}$ where

$$\mathbf{Z} = \{Z_{\mathrm{m,n}}\}^{\mathrm{M} \times \mathrm{N}}, \mathbf{X} = \{X_{\mathrm{m,k}}\}^{\mathrm{M} \times \mathrm{K}}, \mathbf{Y} = \{Y_{\mathrm{k,n}}\}^{\mathrm{K} \times \mathrm{N}}$$

where one or two components (out of three) are (partially) known and the task is to recover the unknown component(s). Here, $\odot$ operation corresponds to a type of matrix multiplication.

A subset of these problems, which are most closely related to Boolean matrix factorization and matrix completion, can be expressed over the Boolean domain – *i.e.*, $Z_{\mathrm{m,n}},\ X_{\mathrm{m,k}},\ Y_{\mathrm{k,n}} \in \{\mathrm{false}, \mathrm{true}\} \cong \{0, 1\}$. The two most common Boolean matrix products used in such applications are

$$\mathbf{Z} = \mathbf{X} \bullet \mathbf{Y} \Rightarrow Z_{\mathrm{m,n}} = \bigvee_{\mathrm{k}=1}^{\mathrm{K}} X_{\mathrm{m,k}} \wedge Y_{\mathrm{k,n}} \tag{1a}$$

$$\mathbf{Z} = \mathbf{X} * \mathbf{Y} \Rightarrow Z_{\mathrm{m,n}} \equiv \Big(\sum_{\mathrm{k}=1}^{\mathrm{K}} X_{\mathrm{m,k}} \wedge Y_{\mathrm{k,n}}\Big) \mod 2 \tag{1b}$$

where we refer to (1a) simply as *Boolean product* and we distinguish (1b) as *exclusive-OR Boolean product*. One may think of Boolean product as ordinary matrix product where the values that are larger than zero in the product matrix are set to one. Alternatively, in XOR product, the odd (even) numbers are identically set the one (zero) in the product matrix.

When $\mathbf{Z}$ and $\mathbf{Y}$ are column vectors (*i.e.*, $\mathrm{N} = 1$), and $Z_{\mathrm{m,n}} = \mathrm{true}$ for all m, n, the problem of finding $\mathbf{Y}$ from a given $\mathbf{X}$ corresponds to *Boolean satisfiability* (a.k.a. SAT); here message passing has been able to solve hard random SAT instances close to the unsatisfiability transition (Braunstein et al., 2005; Ravanbakhsh and Greiner, 2014a). Using the XOR product of (1b), the problem of recovering $\mathbf{Y}$ becomes exclusive-OR satisfiability (a.k.a. XOR-SAT).

Low density parity check (LDPC) codes use the same modulus algebra, with $N = 1$, where instead of the vector $\mathbf{Z}$, we observe $\mathbf{O}$, after transmitting $\mathbf{Z}$ through a noisy channel with known noise model $p^{\mathbf{O}}(\mathbf{O} \mid \mathbf{Z})$. Here again, message passing decoding has been able to transmit $\mathbf{Z}$ and recover $\mathbf{Y}$ (given the parity checks $\mathbf{X}$) at rates close to the theoretical capacity of the communication channel (Gallager, 1962).

LDPC codes are in turn closely related to the *compressed sensing* (Donoho, 2006) – so much so that successful binary LDPC codes (*i.e.*, matrix $\mathbf{X}$) have been reused for compressed sensing (Dimakis et al., 2012). In this setting, the column-vector $\mathbf{Y}$ is known to be ℓ-sparse (*i.e.*, ℓ non-zero values) which makes it possible to use *approximate message passing* (Donoho et al., 2009) to recover $\mathbf{Y}$ using few noisy measurements $\mathbf{O}$ – that is $\mathrm{M} \ll \mathrm{K}$ and similar to LDPC codes, the *measurement matrix* $\mathbf{X}$ is known.

When the underlying algebra is Boolean, the compressed sensing problem reduces to the problem of (noisy) *group testing*, which has a longer history (Du and Hwang, 1993). The intuition is that the non-zero elements of the vector $\mathbf{Y}$ identify the presence or absence of a rare property (*e.g.*, a rare disease or manufacturing defect), therefore $\mathbf{Y}$ is sparse. The objective is to find these non-zero elements (*i.e.*, recover $\mathbf{Y}$) by screening a few ($\mathrm{M} \ll \mathrm{K}$) “subsets” of elements of $\mathbf{Y}$. Each of these $\mathbf{Y}$-bundles corresponds to a row of $\mathbf{X}$ (in (1a)), and message passing has been successfully applied in this setting as well (Atia and Saligrama, 2012; Sejdinovic and Johnson, 2010). These problems over Boolean domain are special instances of the problem of Boolean factor analysis in which both $\mathbf{X}$ and $\mathbf{Y}$ have a matrix form and neither is given. This paper presents an efficient message passing solution to this general setting.

A. Boolean Factor Analysis

The umbrella term “factor analysis” refers to the unsupervised methodology of expressing a set of observations in terms of unobserved factors (McDonald, 2014).[1] In contrast to the closely related problems that we briefly considered in the previous section, in factor analysis, only (a partial and/or distorted version of) the matrix $\mathbf{Z}$ is observed, and our task is then to find low-dimensional $\mathbf{X}$ and $\mathbf{Y}$ whose product

[1]While some definitions restrict factor analysis to variables over continuous domain or even probabilistic models with Gaussian priors, we take a more general view.

is (approximately) $\mathbf{Z}$. Here, approximate message passing techniques (inspired by their success in compressed sensing) have been successfully extended to matrix factorization, matrix calibration and robust PCA over real domain (Krzakala et al., 2013; Parker et al., 2013; Kabashima et al., 2014).

The Boolean factor analysis has a particularly appealing form. This is because the Boolean values simply indicate the presence of a particular factor (a.k.a. basis (Miettinen et al., 2006)/ concept / category (Belohlavek and Vychodil, 2010; Keprt and Snásel, 2004)) in observations, where each factor is in turn a binary pattern.

The combinatorial representation of this problem is *biclique cover* problem in a bipartite graph $\mathcal{G} = (\mathcal{A} \cup \mathcal{B}, \mathcal{E})$. Here a bipartite graph has two disjoint node sets $\mathcal{A}$ (s.t. $|\mathcal{A}| = \mathrm{M}$) and $\mathcal{B}$ (s.t. $|\mathcal{B}| = \mathrm{N}$) where the only edges are between these two sets – *i.e.*, $\mathcal{E} \subseteq \{(a, b) \mid a \in \mathcal{A},\ b \in \mathcal{B}\}$. Here $\mathbf{Z} \in \{0,1\}^{M \times N}$ represents the incident matrix of $\mathcal{G}$ and the objective of factorization is to cover the edges using K bicliques (*i.e.*, *complete* bipartite sub-graphs of $\mathcal{G}$). Here the k^{th} biclique is identified with a subset of $\mathcal{A}$, corresponding to k^{th} column of $\mathbf{X}$, and a subset of $\mathcal{B}$, corresponding to the k^{th} row of $\mathbf{Y}$.

Boolean factorization is also closely related to the tiling problem (Stockmeyer, 1975). Despite its numerous applications (see Vaidya et al., 2007; Lu et al., 2008; Geerts et al., 2004; Zhang et al., 2010), the non-linearity of Boolean matrix product has confined most techniques to heuristics and local search methods (*e.g.*, Keprt and Snásel, 2004; Belohlavek et al., 2007).

I. Bayesian Formulation

This section formulates both Boolean matrix factorization and Boolean matrix completion as maximum a posteriori (MAP) inference in a probabilistic graphical model. We then proposes simple message passing procedure, whose cost (per iteration) is linear in the number of observed elements of the matrix and the *Boolean rank* (a.k.a. Schein rank), which is the smallest number K for which a decomposition of $\mathbf{Z}$ to $\mathbf{X} \bullet \mathbf{Y}$ is exact (Kim, 1982). For noisy observations, Miettinen and Vreeken (2011) suggest using the minimum description length (MDL) principle to estimate K (also see Tatti et al. (2006) for alternative measures).

Here, we also consider "approximate" decompositions, and assume that the value of K is given as an input. To formalize approximate decompositions for binary data, we use a communication channel, where we assume the product matrix $\mathbf{Z}$ is communicated through a *noisy binary erasure channel* (Cover and Thomas, 2012) to produce the observation $\mathbf{O} \in \{0, 1, \mathrm{null}\}^{\mathrm{M} \times \mathrm{N}}$ where $O_{\mathrm{m,n}} = \mathrm{null}$, means this entry was erased in the channel. Using this erasure channel, we can model **matrix completion** using the same formalism that we use for low-rank factorization. We are unaware of any prior work on Boolean matrix completion. This is despite the fact that it naturally choice for collaborative filtering with binary data.

For simplicity, we assume each element of $\mathbf{Z}$ is independently transmitted (that is erased, flipped or remains intact) through the channel, meaning the following conditional probability completely defines the noise model:

$$p^{\mathbf{O}}(\mathbf{O} \mid \mathbf{Z}) = \prod_{\mathrm{m,n}} {p^{\mathbf{O}}}_{\mathrm{m,n}}(O_{\mathrm{m,n}} \mid Z_{\mathrm{m,n}}) \tag{2}$$

Note that each of these conditional probabilities can be represented using six values – one value per each pair of $O_{\mathrm{m,n}} \in \{0, 1, \mathrm{null}\}$ and $Z_{\mathrm{m,n}} \in \{0, 1\}$. This setting allows *the probability of erasure to depend on the value of* m, n *and* $Z_{\mathrm{m,n}}$.

The objective is to recover $\mathbf{X}$ and $\mathbf{Y}$ from $\mathbf{O}$. However, due to its degeneracy, recovering $\mathbf{X}$ and $\mathbf{Y}$ is only up to a $\mathrm{K} \times \mathrm{K}$ permutation matrix $\mathbf{U}$ – that is $\mathbf{X} \bullet \mathbf{Y} = (\mathbf{X} \bullet \mathbf{U}) \bullet (\mathbf{U}^T \bullet \mathbf{Y})$. A Bayesian approach can resolve this ambiguity by defining non-symmetric priors

$$p^{\mathbf{X}}(\mathbf{X}) = \prod_{\mathrm{m,k}} {p^{\mathbf{X}}}_{\mathrm{m,k}}(X_{\mathrm{m,k}}) \tag{3a}$$

$$p^{\mathbf{Y}}(\mathbf{Y}) = \prod_{\mathrm{k,n}} {p^{\mathbf{Y}}}_{\mathrm{k,n}}(Y_{\mathrm{k,n}}) \tag{3b}$$

where we assume separability.

Now, we can express the problem of recovering $\mathbf{X}$ and $\mathbf{Y}$ as a *maximum a posteriori* (MAP) inference problem $\arg_{\mathbf{X},\mathbf{Y}} \max\ p(\mathbf{X}, \mathbf{Y} \mid \mathbf{O})$, where the posterior is

$$p(\mathbf{X}, \mathbf{Y} \mid \mathbf{O}) \ \propto\ p^{\mathbf{X}}(\mathbf{X})\, p^{\mathbf{Y}}(\mathbf{Y})\, p^{\mathbf{O}}(\mathbf{O} \mid \mathbf{X} \bullet \mathbf{Y}) \tag{4}$$

Finding the maximizing assignment for (4) is $\mathbb{NP}$-hard (Stockmeyer, 1975). Our treatment of SAT as a special case of factorization with $\mathrm{m} = 1$ and fixed $\mathbf{Y}$ proves that even the problem of finding the Boolean matrix $\mathbf{X}$ (when $\mathbf{Y}$ and $\mathbf{Z}$ are given) such that $\mathbf{Z} = \mathbf{X} \bullet \mathbf{Y}$, is $\mathbb{NP}$-hard. Here we introduce a graphical model to represent the posterior and use a simplified form of Belief Propagation (BP Pearl, 1982) to approximate the MAP assignment.

An alternative to finding the MAP assignment is that of finding the marginal-MAP – *i.e.*, $\arg_{X_{\mathrm{m,k}}} \max p(X_{\mathrm{m,k}} \mid \mathbf{O}) = \arg_{X_{\mathrm{m,n}}} \max \sum_{\mathbf{X} \backslash x X_i, \mathbf{Y}} p(\mathbf{X}, \mathbf{Y} \mid \mathbf{O})$. While the MAP assignment seeks the optimal joint assignment to $\mathbf{X}$ and $\mathbf{Y}$, finding the marginal-MAP corresponds to optimally estimating individual assignments for each variables, while the other variable assignments are marginalized. We also provide the message passing solution to this alternative in Appendix B. Here, we focus on the MAP estimate as we find it more useful in practice.

A. The Probabilistic Graphical Model

Figure 1 shows the factor-graph (Kschischang et al., 2001) representation of the posterior (4). Here, variables are circles and squares are factors. The factor-graph is a bipartite graph, connecting each factor (i.e. function) to its relevant variables. This factor-graph has one variable $X_{\mathrm{m,k}} \in \{0, 1\}$ for each element of $\mathbf{X}$, and a variable $Y_{\mathrm{k,n}} \in \{0, 1\}$ for each element of $\mathbf{Y}$. In addition to these $\mathrm{K}(\mathrm{M} + \mathrm{N})$ variables, we have introduced $\mathrm{K\,M\,N}$ auxiliary variables $W_{\mathrm{m,n,k}} \in \{0, 1\}$. For Boolean matrix completion the number of auxiliary variables is $\mathrm{K}|\mathcal{E}|$, where $\mathcal{E} = \{(\mathrm{m,n}) | O_{\mathrm{m,n}} \neq \mathrm{null}\}$ is the set of observed elements (see section II-A).

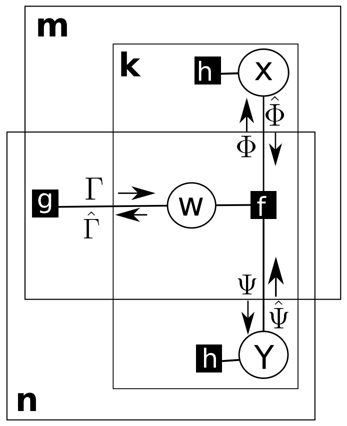


Fig. 1: *The factor-graph and the message exchange between variables and factors.*

We use plate notation (often used with directed models) in representing this factor-graph. Here we have three plates for $1 \leq \mathrm{m} \leq \mathrm{M}$, $1 \leq \mathrm{n} \leq \mathrm{N}$ and $1 \leq \mathrm{k} \leq \mathrm{K}$ (large transparent boxes in fig. 1). In plate notation, all variables and factors on a plate are replicated. For example, variables on the m-plate are replicated for $1 \leq \mathrm{m} \leq \mathrm{M}$. Variables and factors located on more than one plate are replicated for all combinations of their plates. For example, since variable X is in common between m-plate and k-plate, it refers to $\mathrm{M}\,\mathrm{N}$ binary variables – *i.e.*, $X_{\mathrm{m,k}}\ \forall m, \mathrm{n}$.

1) Variables and Factors: The auxiliary variable $W_{\mathrm{m,n,k}}$ represents the Boolean product of $X_{m,\mathrm{k}}$ and $Y_{\mathrm{k,n}}$ – *i.e.*, $W_{\mathrm{m,n,k}} = X_{m,\mathrm{k}} \wedge Y_{\mathrm{k,n}}$. This is achieved through $\mathrm{M}\,\mathrm{N}\,\mathrm{K}$ hard constraint factors

$$\mathsf{f}_{\mathrm{m,n,k}}(X_{\mathrm{m,k}}, Y_{\mathrm{k,n}}, W_{\mathrm{m,n,k}}) \;=\; \mathbb{I}(W_{\mathrm{m,n,k}} = X_{m,\mathrm{k}} \wedge Y_{\mathrm{k,n}})$$

where $\mathbb{I}(.)$ is the identity function on the inference semiring (see Ravanbakhsh and Greiner, 2014b). For the max-sum inference $\mathbb{I}_{\text{max-sum}}(\text{true}) = 0$ and $\mathbb{I}_{\text{max-sum}}(\text{false}) = -\infty$.

Local factors

$$\mathsf{h}_{\mathrm{m,k}}(X_{\mathrm{m,k}}) \;=\; \log(p^{\mathbf{X}}(X_{\mathrm{m,k}}))$$
$$\mathsf{h}_{\mathrm{k,n}}(Y_{\mathrm{k,n}}) \;=\; \log(p^{\mathbf{Y}}(Y_{\mathrm{k,n}}))$$

represent the logarithm of priors over $\mathbf{X}$ and $\mathbf{Y}$ in (4).

Finally, the noise model in (4) is represented by $\mathrm{M}\,\mathrm{N}$ factors over auxiliary variables

$$\mathsf{g}_{\mathrm{m,n}}(\{W_{\mathrm{m,n,k}}\}_{1\leq \mathrm{k}\leq \mathrm{K}}) \;=\; \log\Big({p^{\mathbf{O}}}_{\mathrm{m,n}}(O_{\mathrm{m,n}} \mid \bigvee_{\mathrm{k}} W_{\mathrm{m,n,k}})\Big).$$

The combination of the factors of type g and f, represent the term $p(O_{\mathrm{m,n}} \mid \bigvee_{\mathrm{k}=1}^{\mathrm{K}} X_{\mathrm{m,k}} \wedge Y_{\mathrm{k,n}})$ in (4) and the local factors h, represent the logarithm of the priors. It is easy to see that the sum of all the factors above, evaluates to the logarithm of the posterior

$$\log(p(\mathbf{X},\mathbf{Y} \mid \mathbf{O}) = \sum_{\mathrm{m,k}} \mathsf{h}_{\mathrm{m,k}}(X_{\mathrm{m,k}}) + \sum_{\mathrm{k,n}} \mathsf{h}_{\mathrm{k,n}}(X_{\mathrm{k,n}}) + \sum_{\mathrm{m,n}} \mathsf{g}_{\mathrm{m,n}}(\{X_{m,\mathrm{k}} \wedge Y_{\mathrm{k,n}}\}_{1\leq \mathrm{k}\leq \mathrm{K}})$$

if $W_{\mathrm{m,n,k}} = X_{m,\mathrm{k}} \wedge Y_{\mathrm{k,n}}\ \forall \mathrm{m,n,k}$ and $-\infty$ otherwise. Therefore maximizing the sum of these factors is equivalent to MAP inference for (4).

Algorithm 1: message passing for Boolean matrix factorization/completion

```
Input: 1) observed matrix $\mathbf{O} \in \{0,1\}^{\mathrm{M}\times\mathrm{N}}$; 2) $\mathrm{K} \in \mathbb{N}$;
       3) priors ${p^{\mathbf{X}}}_{\mathrm{m,k}}, {p^{\mathbf{Y}}}_{\mathrm{n,k}}$; 4) noise model
       ${p^{\mathbf{O}}}_{\mathrm{m,n}}\ \forall m, n, k$
Output: $\mathbf{X} \in \{0,1\}^{\mathrm{M}\times\mathrm{K}}$ and $\mathbf{Y} \in \{0,1\}^{\mathrm{K}\times\mathrm{N}}$.
$t := 0$
init $\Phi^{(t)}_{\mathrm{m,n,k}}, \Psi^{(t)}_{\mathrm{m,n,k}}, \hat{\Phi}^{(t)}_{\mathrm{m,n,k}}, \hat{\Psi}^{(t)}_{\mathrm{m,n,k}}, \hat{\Gamma}^{(t)}_{\mathrm{m,n,k}}$ and
$\Gamma^{(t)}_{\mathrm{m,n,k}}\quad \forall \mathrm{m,n,k}$
while $t < T_{\max}$ and not converged do
    $\Phi^{(t+1)}_{\mathrm{m,n,k}} := \big(\Gamma^{(t)}_{\mathrm{m,n,k}} + \hat{\Psi}^{(t)}_{\mathrm{m,n,k}}\big)_+ - \big(\hat{\Psi}^{(t)}_{\mathrm{m,n,k}}\big)_+$   (5a)
    $\Psi^{(t+1)}_{\mathrm{m,n,k}} := \big(\Gamma^{(t)}_{\mathrm{m,n,k}} + \hat{\Phi}^{(t)}_{\mathrm{m,n,k}}\big)_+ - \big(\hat{\Phi}^{(t)}_{\mathrm{m,n,k}}\big)_+$   (5b)
    $\hat{\Phi}^{(t+1)}_{\mathrm{m,n,k}} := \log\left(\frac{{p^{\mathbf{X}}}_{\mathrm{m,k}}(1)}{{p^{\mathbf{X}}}_{\mathrm{m,k}}(0)}\right) + \sum_{\mathrm{n}'\neq\mathrm{n}} \Phi^{(t)}_{\mathrm{m,n',k}}$   (5c)
    $\hat{\Psi}^{(t+1)}_{\mathrm{m,n,k}} := \log\left(\frac{{p^{\mathbf{Y}}}_{\mathrm{n,k}}(1)}{{p^{\mathbf{Y}}}_{\mathrm{n,k}}(0)}\right) + \sum_{\mathrm{m}'\neq\mathrm{m}} \Psi^{(t)}_{\mathrm{m',n,k}}$   (5d)
    $\hat{\Gamma}^{(t+1)}_{\mathrm{m,n,k}} := \min\left\{\hat{\Phi}^{(t)}_{\mathrm{m,n,k}} + \hat{\Psi}^{(t)}_{\mathrm{m,n,k}}, \hat{\Phi}^{(t)}_{\mathrm{m,n,k}}, \hat{\Psi}^{(t)}_{\mathrm{m,n,k}}\right\}$   (5e)
    $\Gamma^{(t+1)}_{\mathrm{m,n,k}} := \min\Big\{\big(-\max_{\mathrm{k}'\neq\mathrm{k}} \hat{\Gamma}^{(t)}_{\mathrm{m,n,k'}}\big)_+,$
    $\sum_{\mathrm{k}'\neq\mathrm{k}} \big(\hat{\Gamma}^{(t)}_{\mathrm{m,n,k'}}\big)_+ + \log\left(\frac{{p^{\mathbf{O}}}_{\mathrm{m,n}}(O_{\mathrm{m,n}} \mid 1)}{{p^{\mathbf{O}}}_{\mathrm{m,n}}(O_{\mathrm{m,n}} \mid 0)}\right)\Big\}$   (5f)
end
calculate log-ratio of the posterior marginals
    $\Xi_{\mathrm{m,k}} := \log\left(\frac{{p^{\mathbf{X}}}_{\mathrm{m,k}}(1)}{{p^{\mathbf{X}}}_{\mathrm{m,k}}(0)}\right) + \sum_{\mathrm{n}} \Phi^{(t)}_{\mathrm{m,n,k}}$   (6a)
    $\Upsilon_{\mathrm{k,n}} := \log\left(\frac{{p^{\mathbf{Y}}}_{\mathrm{k,n}}(1)}{{p^{\mathbf{Y}}}_{\mathrm{k,n}}(0)}\right) + \sum_{\mathrm{m}} \Psi^{(t)}_{\mathrm{m,n,k}}$   (6b)
calculate X and Y
    $X_{\mathrm{m,k}} := \begin{cases} 1, & \text{if } \Xi_{\mathrm{m,k}} > 0 \\ 0, & \text{otherwise} \end{cases}$   (7a)
    $Y_{\mathrm{k,n}} := \begin{cases} 1, & \text{if } \Upsilon_{\mathrm{k,n}} > 0 \\ 0, & \text{otherwise} \end{cases}$   (7b)
return X,Y
```

II. Simplified Form of Belief Propagation

Max-sum Belief Propagation (BP) is a message passing procedure for approximating the MAP assignment in a graphical model. In factor-graphs without loops, max-sum BP is simply an exact dynamic programming approach that leverages the distributive law. In loopy factor-graphs the approximations of this message passing procedure is justified by the fact that it represents the zero temperature limit to sum-product BP, which is in turn a fixed point iteration procedure whose fixed points are the local optima of the Bethe approximation to the free energy (Yedidia et al., 2000); see also (Weiss et al., 2012). For general factor-graphs, it is known that the approximate MAP solution obtained using max-sum BP is optimal within a "neighborhood" (Weiss and Freeman, 2001). Although few applications of max-sum BP to loopy factor-graphs have optimality guarantees (*e.g.*, Bayati et al., 2008; Gamarnik et al., 2012), approximations using BP have been successfully applied to machine learning, coding theory and combinatorial optimization problems.

We apply max-sum BP to approximate the MAP assignment of the factor-graph of fig. 1. This factor-graph is very densely connected and therefore, one expects BP to oscillate or fail to converge to a good solution. However, surprisingly we see in section III that BP performs near-optimally. This can be attributed to the fact that despite dense connectivity, many of the factors have a week influence, often resulting in (close) to uniform messages (see fig. 4). Near-optimal behavior of max-sum BP in dense factor-graph is not without precedence (*e.g.*, Decelle et al., 2011; Ravanbakhsh et al., 2014).

The message passing for MAP inference of (4) involves message exchange between all variables and their neighboring factors in both directions. Here, each message is a Bernoulli distribution. For example $\mathfrak{m}_{X_{\mathrm{m,k}} \to \mathrm{f}_{\mathrm{m,n,k}}}(X_{\mathrm{m,n}}) : \{0,1\} \to \Re^2$ is the message from variable node $X_{\mathrm{m,n}}$ to the factor node $\mathrm{f}_{\mathrm{m,n,k}}$. For binary variables, it is convenient to work with the log-ratio of messages – *e.g.*, we use $\hat{\Phi}_{\mathrm{m,n,k}} = \log\left(\frac{\mathfrak{m}_{X_{\mathrm{m,k}} \to \mathrm{f}_{\mathrm{m,n,k}}}(1)}{\mathfrak{m}_{X_{\mathrm{m,k}} \to \mathrm{f}_{\mathrm{m,n,k}}}(0)}\right)$. In the following, capital Greek letters denote these log-ratios (see also fig. 1).

For a review of max-sum BP and the detailed derivation of the simplified BP updates for this factor-graph, see appendix A. In particular, an important simplification in our derivation, is that of the messages $\Gamma_{\mathrm{m},n}$ from the likelihood factors $\mathrm{g}_{\mathrm{m},n}(\{W_{\mathrm{m,n,k}}\}_{1 \leq \mathrm{k} \leq \mathrm{K}})\ \forall \mathrm{m,n}$ to the auxiliary variables $W_{\mathrm{m,n,k}}$. Since each of these factors depend on K variables, a naive application of BP will have a $\mathcal{O}(2^{\mathrm{K}})$ cost. We reduce this cost to $\mathcal{O}(\mathrm{K})$. algorithm 1 summarizes this simplified message passing algorithm.

At the beginning of the algorithm, $t = 0$, messages are initialized with some random value – *e.g.*, using $\log(U) - \log(1-U)$ where $U \sim \mathrm{Uniform}(0,1)$. Using the short notation $(a)_+ = \max\{0, a\}$, at time $t+1$, the messages are updated using 1) the message values at time t, 2) the priors and 3) the noise model and observation $\mathbf{O}$. The message updates of (5) are repeated until convergence or a maximum number of iterations $T_{\max}$ is reached. We decide the convergence based on the maximum absolute change in one of the message types *e.g.*, $\max_{\mathrm{m,n,k}} |\Phi^{(t+1)}_{\mathrm{m,n,k}} - \Phi^{(t)}_{\mathrm{m,n,k}}| \overset{?}{\leq} \epsilon$.

Once the message update converges, at iteration T, we can use the values for $\Phi^{(\mathrm{T})}_{\mathrm{m,n,k}}$ and $\Psi^{(\mathrm{T})}_{\mathrm{m,n,k}}$ to recover the log-ratio of the marginals $p(X_{\mathrm{m,k}})$ and $p(Y_{\mathrm{n,k}})$. These log-ratios are denoted by $\Xi_{\mathrm{m,k}}$ and $\Upsilon_{\mathrm{k,n}}$ in (6). A positive log-ratio $\Xi_{\mathrm{m,k}} > 0$ means $p(X_{\mathrm{m,k}} = 1) > p(X_{\mathrm{m,k}} = 0)$ and the posterior favors $X_{\mathrm{m,k}} = 1$. In this way the marginals are used to obtain an approximate MAP assignment to both $\mathbf{X}$ and $\mathbf{Y}$.

For better convergence, we also use damping in practice. For this, one type of messages is updated to a linear combination of messages at time t and $t+1$ using a *damping parameter* $\lambda \in (0,1]$. Choosing $\hat{\Phi}$ and $\hat{\Psi}$ for this purpose, the updates of (5c,5d) become

$$\begin{aligned} \hat{\Phi}^{(\mathrm{t}+1)}_{\mathrm{m,n,k}} &:= (1-\lambda)\hat{\Phi}^{(\mathrm{t})}_{\mathrm{m,n,k}} + \\ &\lambda\left(\log\left(\frac{p^{\mathbf{X}}{}_{\mathrm{m,k}}(1)}{p^{\mathbf{X}}{}_{\mathrm{m,k}}(0)}\right) + \sum_{\mathrm{n}' \neq \mathrm{n}} \Phi^{(\mathrm{t})}_{\mathrm{m,n',k}}\right) \\ \hat{\Psi}^{(\mathrm{t}+1)}_{\mathrm{m,n,k}} &:= (1-\lambda)\hat{\Psi}^{(\mathrm{t})}_{\mathrm{m,n,k}} + \\ &\lambda\left(\log\left(\frac{p^{\mathbf{Y}}{}_{\mathrm{n,k}}(1)}{p^{\mathbf{Y}}{}_{\mathrm{n,k}}(0)}\right) + \sum_{\mathrm{m}' \neq \mathrm{m}} \Psi^{(\mathrm{t})}_{\mathrm{m',n,k}}\right) \end{aligned} \tag{8}$$

A. Further Simplifications

Partial knowledge. If any of the priors, $p(X_{\mathrm{m,k}})$ and $p(Y_{\mathrm{n,k}})$, are zero or one, it means that $\mathbf{X}$ and $\mathbf{Y}$ are partially known. The message updates of (5c,5d) will assume $\pm\infty$ values, to reflect these hard constrains. In contrast, for uniform priors, the log-ratio terms disappear.

Matrix completion speed up. Consider the case where $\log\left(\frac{p^{\mathbf{O}}(O_{\mathrm{m,n}}|1)}{p^{\mathbf{O}}(O_{\mathrm{m,n}}|0)}\right) = 0$ in (5f) – *i.e.*, the probabilities in the nominator and denominator are equal. An important case of this happens in matrix completion, when the probability of erasure is independent of the value of $Z_{\mathrm{m,n}}$ – that is $p^{\mathbf{O}}(\mathrm{null} \mid Z_{\mathrm{m,n}} = 0) = p^{\mathbf{O}}(\mathrm{null} \mid Z_{\mathrm{m,n}} = 1) = p^{\mathbf{O}}(\mathrm{null})$ for all m and n.

It is easy to check that in such cases, $\Gamma_{\mathrm{m,n,k}} = \min\big((-\max_{\mathrm{k}' \neq \mathrm{k}} \hat{\Gamma}^{(\mathrm{t})}_{\mathrm{m,n,k}})_+, \sum_{\mathrm{k}' \neq \mathrm{k}} (\hat{\Gamma}^{(\mathrm{t})}_{\mathrm{m,n,k}})_+\big)$ is always zero. This further implies that $\hat{\Phi}_{\mathrm{m,n,k}}$ and $\hat{\Psi}_{\mathrm{m,n,k}}$ in (5c,5d) are also always zero and calculating $\hat{\Gamma}_{\mathrm{m,n,k}}$ in (5f) is pointless. The bottom-line is that we only need to keep track of messages where this log-ratio is non-zero. Let $\mathcal{E} = \{(\mathrm{m,n}) \mid O_{\mathrm{m,n}} \neq \mathrm{null}\}$ denote the observed entries of $\mathbf{O}$. Then in message passing updates of (5) in algorithm 1, wherever the indices m and n appear, we may restrict them to the set $\mathcal{E}$.

Noiseless channel. When $p^{\mathbf{O}}(1 \mid 1) = p^{\mathbf{O}}(0 \mid 0) = 1$, $c^{\mathbf{O}}{}_{m,n}$ evaluates to $-\infty$ or $+\infty$ for $O_{m,n} = 0$ and $O_{m,n} = 1$ respectively. In the former case – *i.e.*, $c^{\mathbf{O}}{}_{m,n} = -\infty$ – the update of (5f) is unnecessary as $\Gamma^{(t+1)}_{\mathrm{m,n,k}} = -\infty$ at all time. Consequently, (5a,5b) further simplify, speeding up the message update for the noiseless Boolean matrix factorization problem.

Belief update. Another trick to reduce the complexity of message updates is in calculating $\{\hat{\Phi}_{\mathrm{m,n,k}}\}_{\mathrm{n}}$ and $\{\hat{\Psi}_{\mathrm{m,n,k}}\}_{\mathrm{m}}$ in (5c,5d). We may calculate the marginals $\Xi_{\mathrm{m,k}}$ and $\Upsilon_{\mathrm{k,n}}$ using (6), and replace the (8) (*i.e.*, the damped version of the (5c,5d)) with

$$\hat{\Phi}^{(\mathrm{t}+1)}_{\mathrm{m,n,k}} := (1-\lambda)\hat{\Phi}^{(\mathrm{t})}_{\mathrm{m,n,k}} + \lambda\big(\Xi^{(\mathrm{t})}_{\mathrm{m,k}} - \Phi^{(\mathrm{t})}_{\mathrm{m,n,k}}\big) \tag{9a}$$

$$\hat{\Psi}^{(\mathrm{t}+1)}_{\mathrm{m,n,k}} := (1-\lambda)\hat{\Psi}^{(\mathrm{t})}_{\mathrm{m,n,k}} + \lambda\big(\Upsilon^{(\mathrm{t})}_{\mathrm{k,n}} - \Psi^{(\mathrm{t})}_{\mathrm{m,n,k}}\big) \tag{9b}$$

where the summation over n' and m' in (5c,5d) respectively, is now performed only once (in producing the marginal) and reused.

Recycling of the max. Finally, using one more computational trick the message passing cost is reduced to linear:

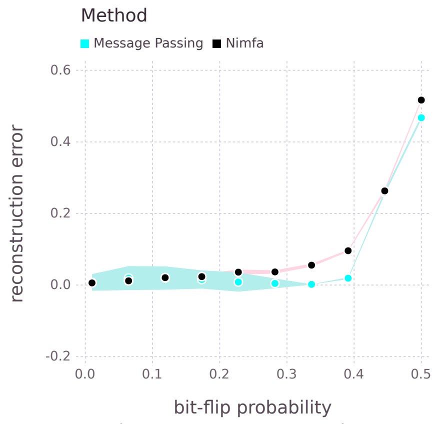


Fig. 3: *Comparison of message passing and NIMFA for Boolean matrix factorization*

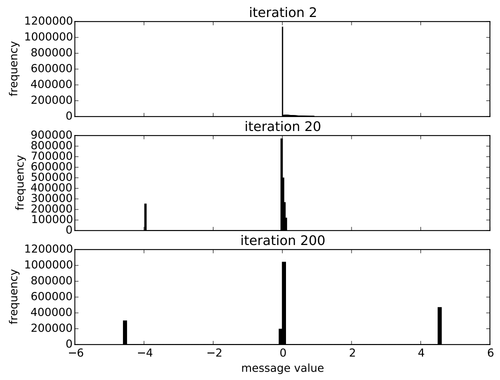


Fig. 4: *Histogram of BP messages* $\{\hat{\Phi}^{(t)}_{\mathrm{m,n}}\}_{\mathrm{m,n}}$ *at* $t \in \{2, 20, 200\}$ *for a random* 1000×1000 *matrix factorization with* $\mathrm{K} = 2$.

in (5e), the maximum of the term $\big(-\max_{\mathrm{k'}\neq\mathrm{k}} \hat{\Gamma}^{(t)}_{\mathrm{m,n,k}}\big)_+$ is calculated for each of K messages $\{\Gamma_{\mathrm{m,n,k}}\}_{\mathrm{k}\in\{1,\ldots,\mathrm{K}\}}$. Here, we may calculate the "two" largest values in the set $\{\hat{\Gamma}^{(t)}_{\mathrm{m,n,k}}\}_{\mathrm{k}}$ only once and reuse them in the updated for all $\{\Gamma_{\mathrm{m,n,k}}\}_{\mathrm{k}}$ – *i.e.*, if the largest value is $\hat{\Gamma}^{(t)}_{\mathrm{m,n,k^*}}$ then we use the second largest value, only in producing $\Gamma_{\mathrm{m,n,k^*}}$.

Computational Complexity. All of the updates in (5a,5b,5f,5e,9) have a constant computational cost. Since these are performed for $\mathrm{K}|\mathcal{E}|$ messages, and the updates in (6a,6b) are $\mathcal{O}(|\mathcal{E}|\mathrm{K})$, the complexity of message update is $\mathcal{O}(|\mathcal{E}|\mathrm{K})$ per iteration.

III. Experiments

We evaluated the performance of message passing on random matrices and real-world data. In all experiments, message passing uses damping with $\lambda = .4$, $T = 200$ iterations and uniform priors $p^{\mathbf{X}}{}_{\mathrm{m,k}}(1) = p^{\mathbf{Y}}{}_{\mathrm{k,n}}(1) = .5$ (*i.e.*, $c^{\mathbf{X}}{}_{\mathrm{m,k}} = c^{\mathbf{Y}}{}_{\mathrm{k,n}} = 0$).[2]

A. Random Matrices

Figure 3 compares the reconstruction error of message passing against state-of-the-art Boolean matrix factorization method of Zhang et al. (2007), which was implemented by NIMFA (Zitnik and Zupan, 2012). Both methods receive the correct K as input. The results are for 1000×1000 random matrices of rank $\mathrm{K} = 5$ where $\mathbf{X}$ and $\mathbf{Y}$ were uniformly sampled from binary matrices. The *reconstruction error* is

$$d(\mathbf{Z}, \widehat{\mathbf{Z}}) \overset{\text{def}}{=} \frac{1}{\mathrm{MN}} \sum_{\mathrm{m,n}} |Z_{\mathrm{m,n}} - \widehat{Z}_{\mathrm{m,n}}|$$

NIMFA uses an alternating optimization method to repeatedly solve a penalized non-negative matrix factorization problem, where the penalty parameters try to enforce the desired binary form.[3]

[2]This also means that if the channel is symmetric – that is $p^{\mathbf{O}}(1 \mid 1) = p^{\mathbf{O}}(0 \mid 0) > .5$ – the approximate MAP reconstruction $\widehat{\mathbf{Z}}$ does not depend on $p^{\mathbf{O}}$, and we could simply use $p^{\mathbf{O}}{}_{\mathrm{m,n}}(1 \mid 1) = p^{\mathbf{O}}{}_{\mathrm{m,n}}(1 \mid 1) = c$ for any $c > .5$.

[3]Both methods use the same number of iterations $T = 200$. For NIMFA we use the default parameters of $\lambda_h = \lambda_w = 1.1$.

The results suggests that two methods are competitive, but message passing performs slightly better at higher noise levels. Here, the experiments were repeated 10 times for each point. The small variance of message passing performance at low noise-levels is due to the multiplicity of symmetric MAP solutions, and could be resolved by performing decimation, albeit at a computational cost. We speculate that the symmetry breaking of higher noise levels help message passing choose a fixed point, which results in lower variance.

Despite being densely connected, at lower levels of noise, BP often converges within the maximum number of iterations. The surprisingly good performance of BP in this setting is due to the fact that most factors have a week influence on many of their neighboring variables. This effectively limits the number of influential loops in the factor-graph. Figure 4 shows the histogram of factor-to-variable messages $\{\hat{\Phi}_{\mathrm{m,n}}\}_{1\leq \mathrm{mM}, 1\leq \mathrm{n} \leq N}$ at different iterations. It suggests that a large portion of messages are close to zero. Since these are log-ratios, the corresponding probabilities are close to uniform.

For the Boolean matrix completion task, and $\mathrm{K} \ll \mathrm{M, N}$, we can lower-bound the *number of observed entries* $\rho = MN(1 - p^{\mathbf{O}}(\mathrm{null}))$ required for recovering $\mathbf{Z}$ by

$$\rho \;>\; \mathrm{K}(\mathrm{M} + \mathrm{N} - \log(\mathrm{K}) + 1) + \mathcal{O}(\log(\mathrm{K})). \qquad (10)$$

To derive this bound, we briefly sketch an information theoretic argument. Note that the total number of ways to define a binary matrix $\mathbf{Z} \in \{0,1\}^{\mathrm{M}\times\mathrm{N}}$ of rank K is $\frac{2^{\mathrm{K(M+N)}}}{\mathrm{K}!}$, where the nominator is the number of different $\mathbf{X}$ and $\mathbf{Y}$ matrices and K! is the irrelevant degree of freedom in choosing the permutation matrix $\mathbf{U}$, such that $\mathbf{Z} = (\mathbf{X} \bullet \mathbf{U}) \bullet (\mathbf{U}^T \bullet \mathbf{Y})$. The logarithm of this number, using Sterling's approximation, is the r.h.s. of (10), lower-bounding the number of bits required to recover $\mathbf{Z}$, in the absence of any noise.

In all panels of fig. 2, each point represents the average reconstruction error for random 1000×1000 Boolean matrices. For each choice of observation sparsity ρ and rank K, the experiments were repeated 10 times. The figure compares the average reconstruction error of Generalized Low-Rank Models (GLRM; Udell et al., 2014) and message passing, where the red line is the information theoretic lower-bound of (10) for perfect recovery. This result suggests that *message passing*

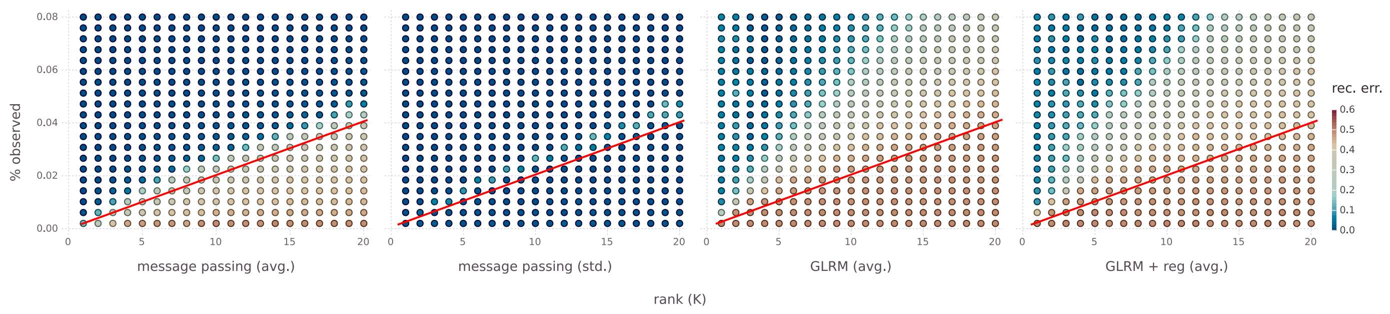


Fig. 2: *The matrix completion error for message passing and GLRM (with real-valued factors) as a function of matrix rank and portion of observed elements* ρ *for* $\mathrm{M} = \mathrm{N} = 1000$. *The red lines indicate the information bottleneck; see (10). From left to right, the 1) average and 2) standard deviation of reconstruction error for message passing; 3) the average error for GLRM with and 4) without regularization.*

is indeed near optimal for Boolean matrix completion. Note that the lower-bound also resembles the $\mathcal{O}(\mathrm{KNpolylog}(\mathrm{N}))$ sample complexity for various real-domain matrix completion tasks (*e.g.*, Candes and Plan, 2010; Keshavan et al., 2010).

GLRM does not produce Boolean factors and it only tries to produce a binary product matrix $\widehat{\mathbf{Z}}$. The choice of GLRM is because of the fact that we could not find any Boolean matrix completion method (with Boolean factors). [4] In our experiments, GLRM is optimizing the *hinge loss*, $\arg_{\mathbf{X},\mathbf{Y}} \min \sum_{\mathrm{m,n}} \left(1 - (\sum_{\mathrm{k}} X_{\mathrm{m,k}} Y_{\mathrm{k,n}})(2O_{\mathrm{m,n}} - 1)\right)_+$, where$(2O_{\mathrm{m,n}} - 1)$ changes the domain of observations to $\{-1, +1\}$.

The better performance of message passing is despite the fact that using real-valued factors gives a higher degree of freedom. Figure 2(message-passing) shows that the transition from recoverability to non-recoverability is sharp, and the variance of the reconstruction error is always close to zero, but in a small neighborhood of the red line.[5] Figure 2(right) demonstrates that the inferior performance of GLRM using real factors is not due to over-fitting as using quadratic regularization (*i.e.*, Gaussian priors over elements of both factors) did not significantly improve its performance.

B. Real-World Applications

This section evaluates message passing on two real-world applications. For application in binary image restoration/completion see Appendix B.

MovieLens. We applied our message passing method to MovieLens-1M dataset[6] as an application in *collaborative filtering*. This benchmark dataset contains 1 million ratings from 6000 users on 4000 movies (*i.e.*, 1/24 of all the ratings are available). The ratings are ordinals 1-5. Here we say a user is "interested" in the movie iff her rating is above the global average of ratings. The task is to predict this single bit by observing a random subset of the available user×movie rating matrix. For this we use $\alpha \in (0, 1)$ portion of the 10^6 ratings to predict the one-bit interest level for the remaining $(1 - \alpha$ portion of the) data-points. Note that here $\rho = \frac{\alpha}{24}$

For Boolean matrix completion, we threshold the training data to $\{0, 1\}$ values from the start using the average rating. For ordinary matrix completion we use GLRM with *ordinal* hinge loss (Rennie and Srebro, 2005) and quadratic regularization to predict the ratings for the unobserved entries.[7] These ratings are then thresholded by the global average rating to predict the interest level.

TABLE I: *Matrix completion performance for MovieLense-1M dataset*

		observed percentage of 1 million ratings					
		1%	5%	10%	20%	50%	95%
K=2	Boolean factorization	**56%**	65%	67%	69%	71%	71%
	ordinal hinge loss	48%	65%	68%	70%	72%	72%
K=4	Boolean factorization	**55%**	**61%**	**63%**	65%	69%	70%
	ordinal hinge loss	45%	50%	57%	62%	62%	71%

Table I reports the test error of the two methods for $K \in \{2, 20\}$, using $\alpha \in \{.01, .05, .1, .2, .5, .95\}$ portion of the available ratings. It is surprising that only using one bit of information per rating, message passing is competitive with GLRM that benefits from the full range of ordinal values. The results also suggest that when only few observations are available (*e.g.*, $\alpha = .01$), message passing performs significantly better.

Davenport et al. (2014) also report a good performance by only observing a single bit, instead of ordinal ratings for the MovieLens dataset, albeit on a small version of this dataset (with 100,000 entries) and for the dense setting of $\alpha = .95$. Similar to GLRM, they assume real factors, and according to their model, the 1-bit observation is obtained by thresholding a function of the product matrix – *i.e.*, $\mathbf{O} = f(\mathbf{Z}) \overset{?}{\geq} c$ for some function f and constant c. Using sigmoid function and $c = .5$, their method is closely related to GLRM with logistic loss (instead of the hinge loss, used in our experiments).

[4]Moreover, GLRM allows different choices of loss function and uses an efficient proximal gradient method that facilitates its application to large matrices.

[5]An issue that is not apparent in fig. 2, is the sparsity of $\mathbf{Z}$. Here, if we generate $\mathbf{X}$ and $\mathbf{Y}$ uniformly at random, as K grows, the matrix $\mathbf{Z} = \mathbf{X} \bullet \mathbf{Y}$ becomes all ones. To avoid this degeneracy, we choose $p^{\mathbf{X}}{}_{\mathrm{m,k}}(X_{\mathrm{m,k}})$ and $p^{\mathbf{Y}}{}_{\mathrm{k,n}}(Y_{\mathrm{k,n}})$ so as to enforce $p(\mathbf{Z} = 1) \approx p(\mathbf{Z} = 0)$. It is easy to check that $p^{\mathbf{X}}{}_{\mathrm{m,k}}(1) = p^{\mathbf{Y}}{}_{\mathrm{k,n}}(1) = \sqrt{1 - \sqrt[\mathrm{K}]{.5}}$ produces this desirable outcome. Note that these probabilities are only used for random matrix generation and the message passing algorithm is using uniform priors $p^{\mathbf{Y}}{}_{\mathrm{k,n}}(1) = p^{\mathbf{X}}{}_{\mathrm{m,k}}(1) = .5$

[6]http://grouplens.org/datasets/movielens/

[7]The results for GLRM are for the regularization parameter in $\{.01, .1, 1, 10\}$, with the best test error. However this choice did not drastically changed the results – *i.e.*, often the test error changed by less than 2%.

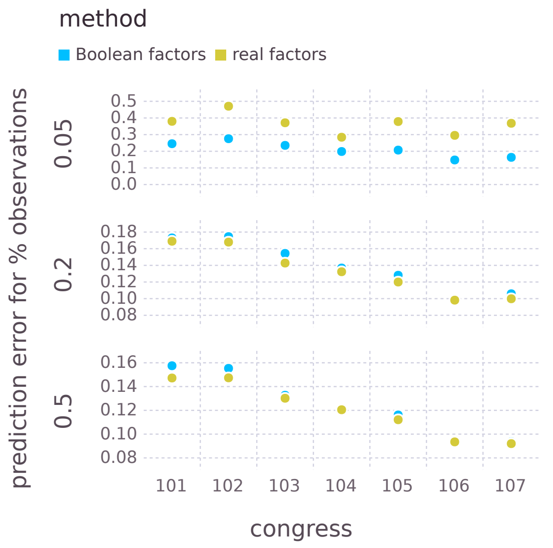


Fig. 5: *the prediction error using Boolean matrix completion (by message passing) versus using GLRM with hinge loss for binary matrix completion using real factors. Each panel has a different observed percentage of entries $\rho \in \{.05, .2, .5\}$. Here the horizontal axis identifies senator×issue matrices and the y-axis is the average error in prediction of the unobserved portion of the (yes/no) votes.*

Reconstructing senate voting records. We applied our noisy completion method to predict the (yes/no) senate votes during 1989-2003 by observing a randomly selected subset of votes.[8] This dataset contains 7 Boolean matrices (corresponding to voting sessions for $101^{st} - 107^{th}$ congress), where a small portion of entries are missing. For example the first matrix is a 634×103 Boolean matrix recording the vote of 102 senators on 634 topics plus the outcome (we ignore the outcome column).

Figure 5 compares the prediction accuracy of message passing and GLRM (with hinge loss or binary predictions) for the best choice of $K \in \{1, \ldots, 10\}$ on each of 7 matrices. GLRM is using quadratic regularization (as it improve its results) while message passing is using uniform priors. In each case we report the prediction accuracy on the unobserved entries, after observing $\rho \in \{5\%, 20\%, 50\%\}$ of the votes. For sparse observations ($\rho = .05$), the message passing error is almost always half of the error when we use real factors. With larger number of observations, the methods are comparable, with GLRM performing slightly better.

Conclusion & Future Work

This paper introduced a simple message passing technique for approximate Boolean factorization and noisy matrix completion. While having a linear time complexity, this procedure achieves near-optimal performance in Boolean matrix completion problem and favorably compares with the state-of-the-art in Boolean matrix factorization. In particular, for matrix completion with few entries, message passing significantly outperforms the existing methods that use real factors. This makes message passing is a useful candidate for collaborative filtering in modern applications involving large datasets of sparse Boolean observations.

Boolean matrix factorization with modular arithmetic (see (1b)) replaces the logical OR operation with exclusive-OR, only changing one of the factor types (*i.e.*, type g) in our graphical model. Therefore both min-sum and sum-product message passing can also be applied to this variation. The similarity of this type of Boolean factorization to LDPC codes (Gallager, 1962), suggests that one may be able to use noisy matrix completion as an efficient method of communication over a noisy channel. This is particularly interesting, as both the code and its parity checks are transferred in the same matrix. We leave this promising direction to future work.

[8]The senate data was obtained from http://www.stat.columbia.edu/~jakulin/Politics/senate-data.zip prepared by Jakulin et al. (2009).

References

George K Atia and Venkatesh Saligrama. Boolean compressed sensing and noisy group testing. *Information Theory, IEEE Transactions on*, 58(3):1880–1901, 2012.

Mohsen Bayati, Devavrat Shah, and Mayank Sharma. Max-product for maximum weight matching: Convergence, correctness, and lp duality. *Information Theory, IEEE Transactions on*, 54(3):1241–1251, 2008.

Radim Belohlavek and Vilem Vychodil. Discovery of optimal factors in binary data via a novel method of matrix decomposition. *Journal of Computer and System Sciences*, 76(1): 3–20, 2010.

Radim Belohlavek, Jiří Dvořák, and Jan Outrata. Fast factorization by similarity in formal concept analysis of data with fuzzy attributes. *Journal of Computer and System Sciences*, 73(6):1012–1022, 2007.

Alfredo Braunstein, Marc Mézard, and Riccardo Zecchina. Survey propagation: An algorithm for satisfiability. *Random Structures & Algorithms*, 27(2):201–226, 2005.

Emmanuel J Candes and Yaniv Plan. Matrix completion with noise. *Proceedings of the IEEE*, 98(6):925–936, 2010.

Thomas M Cover and Joy A Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

Mark A Davenport, Yaniv Plan, Ewout van den Berg, and Mary Wootters. 1-bit matrix completion. *Information and Inference*, 3(3):189–223, 2014.

Aurelien Decelle, Florent Krzakala, Cristopher Moore, and Lenka Zdeborová. Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications. *Physical Review E*, 84(6):066106, 2011.

Alexandros G Dimakis, Roxana Smarandache, and Pascal O Vontobel. Ldpc codes for compressed sensing. *Information Theory, IEEE Transactions on*, 58(5):3093–3114, 2012.

David L Donoho. Compressed sensing. *Information Theory, IEEE Transactions on*, 52(4):1289–1306, 2006.

David L Donoho, Arian Maleki, and Andrea Montanari. Message-passing algorithms for compressed sensing. *Proceedings of the National Academy of Sciences*, 106(45): 18914–18919, 2009.

Ding-Zhu Du and Frank K Hwang. *Combinatorial group testing and its applications*. World Scientific, 1993.

Robert G Gallager. Low-density parity-check codes. *Information Theory, IRE Transactions on*, 8(1):21–28, 1962.

David Gamarnik, Devavrat Shah, and Yehua Wei. Belief propagation for min-cost network flow: Convergence and correctness. *Operations Research*, 60(2):410–428, 2012.

Floris Geerts, Bart Goethals, and Taneli Mielikäinen. Tiling databases. In *Discovery science*, pages 278–289. Springer, 2004.

Aleks Jakulin, Wray Buntine, Timothy M La Pira, and Holly Brasher. Analyzing the us senate in 2003: Similarities, clusters, and blocs. *Political Analysis*, page mpp006, 2009.

Yoshiyuki Kabashima, Florent Krzakala, Marc Mézard, Ayaka Sakata, and Lenka Zdeborová. Phase transitions and sample complexity in bayes-optimal matrix factorization. *arXiv preprint arXiv:1402.1298*, 2014.

Ales Keprt and Václav Snásel. Binary factor analysis with help of formal concepts. In *CLA*, volume 110, pages 90–101, 2004.

Raghunandan H Keshavan, Andrea Montanari, and Sewoong Oh. Matrix completion from a few entries. *Information Theory, IEEE Transactions on*, 56(6):2980–2998, 2010.

K.H. Kim. *Boolean matrix theory and applications*. Monographs and textbooks in pure and applied mathematics. Dekker, 1982. ISBN 9780824717889.

Florent Krzakala, Marc Mézard, and Lenka Zdeborová. Phase diagram and approximate message passing for blind calibration and dictionary learning. In *Information Theory Proceedings (ISIT), 2013 IEEE International Symposium on*, pages 659–663. IEEE, 2013.

Frank R Kschischang, Brendan J Frey, and Hans-Andrea Loeliger. Factor graphs and the sum-product algorithm. *Information Theory, IEEE Transactions on*, 47(2):498–519, 2001.

Haibing Lu, Jaideep Vaidya, and Vijayalakshmi Atluri. Optimal boolean matrix decomposition: Application to role engineering. In *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*, pages 297–306. IEEE, 2008.

Roderick P McDonald. *Factor analysis and related methods*. Psychology Press, 2014.

Pauli Miettinen and Jilles Vreeken. Model order selection for boolean matrix factorization. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 51–59. ACM, 2011.

Pauli Miettinen, Taneli Mielikäinen, Aristides Gionis, Gautam Das, and Heikki Mannila. The discrete basis problem. In *Knowledge Discovery in Databases: PKDD 2006*, pages 335–346. Springer, 2006.

Jason T Parker, Philip Schniter, and Volkan Cevher. Bilinear generalized approximate message passing. *arXiv preprint arXiv:1310.2632*, 2013.

Judea Pearl. Reverend bayes on inference engines: A distributed hierarchical approach. In *AAAI*, pages 133–136, 1982.

Siamak Ravanbakhsh and Russell Greiner. Perturbed message passing for constraint satisfaction problems. *arXiv preprint arXiv:1401.6686*, 2014a.

Siamak Ravanbakhsh and Russell Greiner. Revisiting algebra and complexity of inference in graphical models. *arXiv preprint arXiv:1409.7410*, 2014b.

Siamak Ravanbakhsh, Reihaneh Rabbany, and Russell Greiner. Augmentative message passing for traveling salesman problem and graph partitioning. In *Advances in Neural Information Processing Systems*, pages 289–297, 2014.

Jason DM Rennie and Nathan Srebro. Loss functions for preference levels: Regression with discrete ordered labels. In *Proceedings of the IJCAI multidisciplinary workshop on advances in preference handling*, pages 180–186. Kluwer Norwell, MA, 2005.

Dino Sejdinovic and Oliver Johnson. Note on noisy group testing: asymptotic bounds and belief propagation reconstruction. In *Communication, Control, and Computing (Allerton), 2010 48th Annual Allerton Conference on*, pages 998–1003. IEEE, 2010.

Larry J Stockmeyer. *The set basis problem is NP-complete*. IBM Thomas J. Watson Research Division, 1975.

Nikolaj Tatti, Taneli Mielikainen, Aristides Gionis, and Heikki Mannila. What is the dimension of your binary data? In *Data Mining, 2006. ICDM'06. Sixth International Conference on*, pages 603–612. IEEE, 2006.

Madeleine Udell, Corinne Horn, Reza Zadeh, and Stephen Boyd. Generalized low rank models. *arXiv preprint arXiv:1410.0342*, 2014.

Jaideep Vaidya, Vijayalakshmi Atluri, and Qi Guo. The role mining problem: finding a minimal descriptive set of roles. In *Proceedings of the 12th ACM symposium on Access control models and technologies*, pages 175–184. ACM, 2007.

Yair Weiss and William T Freeman. On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs. *Information Theory, IEEE Transactions on*, 47(2):736–744, 2001.

Yair Weiss, Chen Yanover, and Talya Meltzer. Map estimation, linear programming and belief propagation with convex free energies. *arXiv preprint arXiv:1206.5286*, 2012.

Jonathan S Yedidia, William T Freeman, Yair Weiss, et al. Generalized belief propagation. In *NIPS*, volume 13, pages 689–695, 2000.

Zhong-Yuan Zhang, Tao Li, Chris Ding, Xian-Wen Ren, and Xiang-Sun Zhang. Binary matrix factorization for analyzing gene expression data. *Data Mining and Knowledge Discovery*, 20(1):28–52, 2010.

Zhongyuan Zhang, Chris Ding, Tao Li, and Xiangsun Zhang. Binary matrix factorization with applications. In *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*, pages 391–400. IEEE, 2007.

Marinka Zitnik and Blaz Zupan. Nimfa: A python library for nonnegative matrix factorization. *Journal of Machine Learning Research*, 13:849–853, 2012.

Appendix A
Detailed Derivation of Simplified BP Messages

The sum of the factors in the factor-graph of fig. 1 is

$$\begin{aligned}
&\sum_{\mathrm{m,k}} \mathsf{h}_{\mathrm{m,k}}(X_{\mathrm{m,k}}) + \sum_{\mathrm{n,k}} \mathsf{h}_{\mathrm{n,k}}(Y_{\mathrm{n,k}}) + \\
&\sum_{\mathrm{m,n,k}} \mathsf{f}_{\mathrm{m,n,k}}(X_{\mathrm{m,n,k}}, Y_{\mathrm{m,n,k}}, W_{\mathrm{m,n,k}}) + \\
&\sum_{\mathrm{m,n}} \mathsf{g}_{\mathrm{m,n}}(\{W_{\mathrm{m,n,k}}\}_{\mathrm{k}}) \qquad (11)\\
=&\sum_{\mathrm{m,n}} \log(p^{\mathbf{X}}(X_{\mathrm{m,k}})) + \sum_{\mathrm{n,k}} \log(p^{\mathbf{Y}}(Y_{\mathrm{k,n}})) + \\
&\sum_{\mathrm{m,n,k}} \mathbb{I}(W_{\mathrm{m,n,k}} = X_{m,\mathrm{k}} \wedge Y_{\mathrm{k,n}}) + \\
&\sum_{\mathrm{m,n}} \log\left(p^{\mathbf{O}}{}_{\mathrm{m,n}}(O_{\mathrm{m,n}} \mid \bigvee_{\mathrm{k}} W_{\mathrm{m,n,k}})\right) \qquad (12)\\
=&\sum_{\mathrm{m,n}} \log(p^{\mathbf{X}}(X_{\mathrm{m,k}})) + \sum_{\mathrm{n,k}} \log(p^{\mathbf{Y}}(Y_{\mathrm{k,n}})) + \\
&\sum_{\mathrm{m,n}} \log\left(p^{\mathbf{O}}{}_{\mathrm{m,n}}(O_{\mathrm{m,n}} \mid \bigvee_{\mathrm{k}} X_{m,\mathrm{k}} \wedge Y_{\mathrm{k,n}})\right) \qquad (13)\\
=&\log(p(\mathbf{X}, \mathbf{Y} \mid \mathbf{O})) \qquad (14)
\end{aligned}$$

where in (12) we replaced each factor with its definition. (13) combines the two last terms of (12), which is equivalent to marginalizing out $\mathbf{W}$. The final result of (14) is the log-posterior of (4).

Since the original MAP inference problem of $\arg_{\mathbf{X},\mathbf{Y}} \max\ p(\mathbf{X}, \mathbf{Y} \mid \mathbf{O})$ is equivalent to $\arg_{\mathbf{X},\mathbf{Y}} \max\ \log(p(\mathbf{X}, \mathbf{Y} \mid \mathbf{O}))$, our objective is to perform max-sum inference over this factor-graph, finding an assignment that maximizes the summation of (11)

We perform this max-sum inference using Belief Propagation (BP). Applied to a factor-graph, BP involves message exchange between neighboring variable and factor nodes. Two most well-known variations of BP are sum-product BP for marginalization and max-product or max-sum BP for MAP inference. Here, we provide some details on algebraic manipulations that lead to the simplified form of max-sum BP message updates of (5). (A-A) obtains the updates (5c) and (5d) in our algorithm and (A-B) reviews the remaining message updates of (5)

A. Variable-to-factor messages

Consider the binary variable $X_{\mathrm{m,k}} \in \{0,1\}$ in the graphical model of fig. 1. Let $\mathfrak{m}_{X_{\mathrm{m,k}} \to \mathsf{f}_{\mathrm{m,n,k}}}(X_{\mathrm{m,k}}) : \{0,1\} \to \Re$ be the *message* from variable $X_{\mathrm{m,k}}$ to the factor $\mathsf{f}_{\mathrm{m,n,k}}$ in this factor-graph. Note that this message contains two assignments for $X_{\mathrm{m,k}} = 0$ and $X_{\mathrm{m,k}} = 1$. As we show here, in our simplified updates this message is represented by $\hat{\Phi}_{\mathrm{m,n,k}}$. In the max-sum BP, the outgoing message from any variable to a neighboring factor is the sum of all incoming messages, except for the message from the receiving factor – *i.e.*,

$$\begin{aligned}
\mathfrak{m}_{X_{\mathrm{m,k}} \to \mathsf{f}_{\mathrm{m,n,k}}}(X_{\mathrm{m,k}})^{(\mathrm{t}+1)} &= \mathfrak{m}_{\mathsf{h}_{\mathrm{m,k}} \to X_{\mathrm{m,k}}}(X_{\mathrm{m,k}})^{(\mathrm{t})} \\
+ \sum_{\mathrm{n'} \neq \mathrm{n}} \mathfrak{m}_{\mathsf{f}_{\mathrm{m,n',k}} \to X_{\mathrm{m,k}}}(X_{\mathrm{m,k}})^{(\mathrm{t})} &+ c \qquad (15)
\end{aligned}$$

What matters in BP messages is the difference between the message $\mathfrak{m}_{X_{\mathrm{m,k}} \to \mathsf{f}_{\mathrm{m,n,k}}}(X_{\mathrm{m,k}})$ assignment for $X_{\mathrm{m,k}} = 1$ and $X_{\mathrm{m,k}} = 0$ (note the constant c in (15)). Therefore we can use a singleton message value that capture this difference instead of using a message over the binary domain – *i.e.*,

$$\hat{\Phi}_{\mathrm{m,n,k}} = \mathfrak{m}_{X_{\mathrm{m,k}} \to \mathsf{f}_{\mathrm{m,n,k}}}(1) - \mathfrak{m}_{X_{\mathrm{m,k}} \to \mathsf{f}_{\mathrm{m,n,k}}}(0) \qquad (16)$$

This is equivalent to assuming that the messages are normalized so that $\mathfrak{m}_{X_{\mathrm{m,k}} \to \mathsf{f}_{\mathrm{m,n,k}}}(0) = 0$. We will extensively use this normalization assumption in the following. By substituting (15) in (16) we get the simplified update of (5c)

$$\begin{aligned}
\hat{\Phi}^{(\mathrm{t}+1)}_{\mathrm{m,n,k}} &= \left(\mathfrak{m}_{\mathsf{h}_{\mathrm{m,k}} \to X_{\mathrm{m,k}}}(1)^{(\mathrm{t})} + \sum_{\mathrm{n'} \neq \mathrm{n}} \mathfrak{m}_{\mathsf{f}_{\mathrm{m,n',k}} \to X_{\mathrm{m,k}}}(1)^{(\mathrm{t})}(1)\right) \\
&- \left(\mathfrak{m}_{\mathsf{h}_{\mathrm{m,k}} \to X_{\mathrm{m,k}}}(0)^{(\mathrm{t})} + \sum_{\mathrm{n'} \neq \mathrm{n}} \mathfrak{m}_{\mathsf{f}_{\mathrm{m,n',k}} \to X_{\mathrm{m,k}}}(0)^{(\mathrm{t})}\right) \\
&= \left(\mathfrak{m}_{\mathsf{h}_{\mathrm{m,k}} \to X_{\mathrm{m,k}}}(1)^{(\mathrm{t})} - \mathfrak{m}_{\mathsf{h}_{\mathrm{m,k}} \to X_{\mathrm{m,k}}}(0)^{(\mathrm{t})}\right) \\
&+ \sum_{n' \neq n} \left(\mathfrak{m}_{\mathsf{f}_{\mathrm{m,n',k}} \to X_{\mathrm{m,k}}(1)^{(\mathrm{t})}} - \mathfrak{m}_{\mathsf{f}_{\mathrm{m,n',k}} \to X_{\mathrm{m,k}}}(0)^{(\mathrm{t})}\right) \\
&= \log\left(\frac{p^{\mathbf{X}}{}_{\mathrm{m,k}}(1)}{p^{\mathbf{X}}{}_{\mathrm{m,k}}(0)}\right) + \sum_{\mathrm{n'} \neq \mathrm{n}} \Phi^{(\mathrm{t})}_{\mathrm{m,n',k}}
\end{aligned}$$

and we used the fact that

$$\begin{aligned}
\Phi_{\mathrm{m,n',k}} &= \mathfrak{m}_{\mathsf{f}_{\mathrm{m,n',k}} \to X_{\mathrm{m,k}}}(1)^{(\mathrm{t})} - \mathfrak{m}_{\mathsf{f}_{\mathrm{m,n',k}} \to X_{\mathrm{m,k}}}(0)^{(\mathrm{t})} \\
\log\left(\frac{p^{\mathbf{X}}{}_{\mathrm{m,k}}(1)}{p^{\mathbf{X}}{}_{\mathrm{m,k}}(0)}\right) &= \mathsf{h}_{\mathrm{m,k}}(1) - \mathsf{h}_{\mathrm{m,k}}(0).
\end{aligned}$$

The messages $\hat{\Psi}_{\mathrm{m,n,k}}$ from the variables $Y_{\mathrm{n,k}}$ to $\mathsf{f}_{\mathrm{m,n,k}}$ is obtain similarly. The only remaining variable-to-factor messages in the factor-graph of fig. 1 are from auxiliary variables $W_{\mathrm{m},n,k}$ to neighboring factors. However, since each variable $W_{\mathrm{m,n},k}$ has exactly two neighboring factors, the message from $W_{\mathrm{m,n,k}}$ to any of these factors is simply the incoming message from the other factor – that is

$$\begin{aligned}
\mathfrak{m}_{W_{\mathrm{m,n,k}} \to \mathsf{g}_{\mathrm{m,n}}}(W_{\mathrm{m,n,k}}) &= \mathfrak{m}_{\mathsf{f}_{\mathrm{m,n,k}} \to W_{\mathrm{m,n,k}}}(W_{\mathrm{m,n,k}}) \\
\mathfrak{m}_{\mathsf{g}_{\mathrm{m,n}} \to W_{\mathrm{m,n,k}}}(W_{\mathrm{m,n,k}}) &= \mathfrak{m}_{W_{\mathrm{m,n,k}} \to \mathsf{f}_{\mathrm{m,n,k}}}(W_{\mathrm{m,n,k}}) \qquad (17)
\end{aligned}$$

B. Factor-to-variable messages

The factor-graph of fig. 1 has three types of factors. We obtain the simplified messages from each of these factors to their neighboring variables in the following sections.

1) Local factors: The local factors are $\{\mathsf{h}_{\mathrm{m},k}\}_{\mathrm{m,k}}$ and $\{\mathsf{h}_{n,k}\}_{\mathrm{n,k}}$, each of which is only connected to a single variable. The unnormalized message, leaving these factors is identical to the factor itself. We already used the normalized messages from these local factors to neighboring variables in (17) – *i.e.*, $\mathsf{h}_{\mathrm{m},k}(1) - \mathsf{h}_{\mathrm{m},k}(0)$ and $\mathsf{h}_{\mathrm{n},k}(1) - \mathsf{h}_{\mathrm{n},k}(0)$, respectively.

2) Constraint factors: The constraint factors $\{\mathsf{f}_{\mathrm{m,n,k}}\}_{\mathrm{m,n,k}}$ ensure $\forall_{\mathrm{m,n,k}} W_{\mathrm{m,n,k}} = X_{\mathrm{m,k}} \wedge Y_{\mathrm{n,k}}$. Each of these factors has three neighboring variables. In max-sum BP the message from a factor to a neighboring variable is given by the sum of that factor and incoming messages from its neighboring variables, except for the receiving variable, max-marginalized over the

domain of the receiving variable. Here we first calculate the messages from a constraint factor to $X_{\mathrm{m,k}}$ (or equivalently $Y_{\mathrm{n,k}}$) variables in **(1)**. In **(2)** we derive the simplified messages to the auxiliary variable $W_{\mathrm{m,n,k}}$.
(1) according to max-sum BP equations the message from the factor $\mathsf{f}_{\mathrm{m,n,k}}$ to variable $X_{\mathrm{m,k}}$ is

$$\begin{aligned}&\mathfrak{m}_{\mathsf{f}_{\mathrm{m,n,k}}\to X_{\mathrm{m,k}}}(X_{\mathrm{m,k}})^{(\mathrm{t+1})} = \\ &\max_{W_{\mathrm{m,n,k}},Y_{\mathrm{n,k}}}\Big(\mathsf{f}_{\mathrm{m,n,k}}(X_{\mathrm{m,k}},W_{\mathrm{m,n,k}},Y_{\mathrm{n,k}}) \\ &+\mathfrak{m}_{Y_{\mathrm{n,k}}\mathsf{f}_{\mathrm{m,n,k}}\to(}(Y_{\mathrm{n,k}})^{(\mathrm{t})}+\mathfrak{m}_{W_{\mathrm{m,n,k}}\to\mathsf{f}_{\mathrm{m,n,k}}}(W_{\mathrm{m,n,k}})^{(\mathrm{t})}\Big)\end{aligned}$$

For notational simplicity we temporarily use the shortened version of the above

$$\mathfrak{m}'_1(X)=\max_{W,Y}\mathsf{f}(X,W,Y)\ +\mathfrak{m}_2(Y)+\mathfrak{m}_3(W) \tag{18}$$

where

$$\begin{aligned}\mathfrak{m}_1(X)&=\mathfrak{m}_{X_{\mathrm{m,k}}\to\mathsf{f}_{\mathrm{m,n,k}}}(X_{\mathrm{m,k}})\\ \mathfrak{m}'_1(X)&=\mathfrak{m}_{\mathsf{f}_{\mathrm{m,n,k}}\to X_{\mathrm{m,k}}}(X_{\mathrm{m,k}})\\ \mathfrak{m}_2(Y)&=\mathfrak{m}_{Y_{\mathrm{n,k}}\to\mathsf{f}_{\mathrm{m,n,k}}}(Y_{\mathrm{n,k}})\\ \mathfrak{m}'_2(Y)&=\mathfrak{m}_{\mathsf{f}_{\mathrm{m,n,k}}\to Y_{\mathrm{n,k}}}(Y_{\mathrm{n,k}})\\ \mathfrak{m}_3(W)&=\mathfrak{m}_{W_{\mathrm{m,n,k}}\to\mathsf{f}_{\mathrm{m,n,k}}}(W_{\mathrm{m,n,k}})\\ \mathfrak{m}'_3(W)&=\mathfrak{m}_{\mathsf{f}_{\mathrm{m,n,k}}\to W_{\mathrm{m,n,k}}}(W_{\mathrm{m,n,k}}),\end{aligned}$$

that is we use $\mathfrak{m}(.)$ to denote the incoming messages to the factor and $\mathfrak{m}'(.)$ to identify the outgoing message.

If the constraint $\mathsf{f}(X,Y,W)=\mathbb{I}(W=X\wedge Y)$ is not satisfied by an assignment to X,Y and W, it evaluates to $-\infty$, and therefore it does not have any effect on the outgoing message due to the $\max$ operation. Therefore we should consider the $\max_{W,Y}$ only over the assignments that satisfy $\mathsf{f}(.)$.

Here, X can have two assignments; for $X=1$, if $Y=1$, then $W=1$ is enforced by $\mathsf{f}(.)$, and if $Y=0$ then $W=0$. Therefore (18) for $X=1$ becomes

$$\mathfrak{m}'_1(1)=\max(\mathfrak{m}_2(1)+\mathfrak{m}_3(1),\mathfrak{m}_2(0)+\mathfrak{m}_3(0)) \tag{19}$$

For $X=0$, we have $W=0$, regardless of Y and the update of (18) reduces to

$$\begin{aligned}\mathfrak{m}'_1(0)\ &=\ \max(\mathfrak{m}_2(1)+\mathfrak{m}_3(0),\mathfrak{m}_2(0)+\mathfrak{m}_3(0)\} \\ &=\mathfrak{m}_3(0)+\max\{\mathfrak{m}2(0),\mathfrak{m}2(1)\}\end{aligned} \tag{20}$$

Assuming the incoming messages are normalized such that $\mathfrak{m}_3(0)=\mathfrak{m}_2(0)=0$ and denoting

$$\hat{\Psi}_{\mathrm{m,n,k}}=\mathfrak{m}_{Y_{\mathrm{n,k}}\to\mathsf{f}_{\mathrm{m,n,k}}}(1)-\mathfrak{m}_{Y_{\mathrm{n,k}}\to\mathsf{f}_{\mathrm{m,n,k}}}(0)=\mathfrak{m}_2(1)$$

and

$$\Gamma_{\mathrm{m,n,k}}=\mathfrak{m}_{W_{\mathrm{m,n,k}}\to\mathsf{f}_{\mathrm{m,n,k}}}(1)-\mathfrak{m}_{W_{\mathrm{m,n,k}}\to\mathsf{f}_{\mathrm{m,n,k}}}(0)=\mathfrak{m}_3(1)$$

the difference of (19) and (20) gives the normalized outgoing message of (5a)

$$\begin{aligned}\Phi_{\mathrm{m,n,k}}=&\mathfrak{m}'_1(1)-\mathfrak{m}'_1(0)=\max(\Gamma_{\mathrm{m,n,k}}+\hat{\Psi}_{\mathrm{m,n,k}},0)\\ &-\max(0,\hat{\Psi}_{\mathrm{m,n,k}})\end{aligned} \tag{21}$$

The message of (5b) from the constraint $\mathsf{f}_{\mathrm{m,n,k}}$ to $Y_{\mathrm{n,k}}$ is obtained in exactly the same way.

(2) The max-sum BP message from the constraint factor $\mathsf{f}_{\mathrm{m,n,k}}$ to the auxiliary variable $W_{\mathrm{m,n,k}}$ is

$$\begin{aligned}&\mathfrak{m}_{\mathsf{f}_{\mathrm{m,n,k}}\to W_{\mathrm{m,n,k}}}(W_{\mathrm{m,n,k}})^{(\mathrm{t+1})} = \\ &\max_{X_{\mathrm{m,k}},Y_{\mathrm{n,k}}}\big(\mathsf{f}_{\mathrm{m,n,k}}(X_{\mathrm{m,k}},W_{\mathrm{m,n,k}},Y_{\mathrm{n,k}})+ \\ &\mathfrak{m}_{Y_{\mathrm{n,k}}\to\mathsf{f}_{\mathrm{m,n,k}}}(Y_{\mathrm{n,k}})^{(\mathrm{t})}+\mathfrak{m}_{X_{\mathrm{m,k}}\to\mathsf{f}_{\mathrm{m,n,k}}}(W_{\mathrm{m,n,k}})^{(\mathrm{t})}\big)\end{aligned}$$

Here, again we use the short notation

$$\mathfrak{m}'_3(W)=\max_{X,Y}\mathsf{f}(X,W,Y)\ +\mathfrak{m}_1(X)+\mathfrak{m}_2(Y) \tag{22}$$

and consider the outgoing message $\mathfrak{m}'(W)$ for $W=1$ and $W=0$. If $W=1$, we know that $X=Y=1$. This is because otherwise the factor f evaluates to $-\infty$. This simplifies (22) to

$$\mathfrak{m}'_3(1)=\mathfrak{m}_1(1)+\mathfrak{m}_2(1)$$

For $W=0$, either $X=0$, or $Y=0$ or both. This means

$$\begin{aligned}\mathfrak{m}'_3(0)=\max(&\mathfrak{m}_1(0)+\mathfrak{m}_2(1),\mathfrak{m}+1(1)+\mathfrak{m}_2(0),\\ &\mathfrak{m}_1(0)+\mathfrak{m}_2(0))\end{aligned}$$

Assuming the incoming messages were normalized, such that $\mathfrak{m}_2(0)=\mathfrak{m}_1(0)=0$, the normalized outgoing message $\hat{\Gamma}_{\mathrm{m,n,k}}=\mathfrak{m}_3(1)-\mathfrak{m}_3(0)$ simplifies to

$$\begin{aligned}\hat{\Gamma}_{\mathrm{m,n,k}}&=\mathfrak{m}_1(1)+\mathfrak{m}_2(1)-\max(0,\mathfrak{m}_1(1),\mathfrak{m}_2(1))\\ &=\min(\mathfrak{m}_1(1)+\mathfrak{m}_2(1),\mathfrak{m}_1(1),\mathfrak{m}_2(1))\\ &=\min(\hat{\Phi}_{\mathrm{m,n,k}}+\hat{\Psi}_{\mathrm{m,n,k}},\hat{\Phi}_{\mathrm{m,n,k}},\hat{\Psi}_{\mathrm{m,n,k}})\end{aligned}$$

C. *Likelihood factors*

At this point we have derived all simplified message updates of (5), except for the message $\Gamma_{\mathrm{m,n,k}}$ from factors $\mathsf{g}_{m,n}$ to the auxiliary variables $W_{\mathrm{m,n,k}}$ ((5f)). These factors encode the likelihood term in the factor-graph.

The naive form of max-sum BP for the messages leaving this factor to each of K neighboring variables $\{W_{\mathrm{m,n,k}}\}_{1\leq \mathrm{k}\leq \mathrm{K}}$ is

$$\begin{aligned}&\mathfrak{m}_{\mathsf{g}_{\mathrm{m,n}}\to W_{\mathrm{m,n,k}}}(W_{\mathrm{m,n,k}})^{(\mathrm{t+1})} = \\ &\max_{\{W_{\mathrm{m,n},\ell}\}_{\ell\neq\mathrm{k}}}\Big(\mathsf{g}_{\mathrm{m,n}}(\{W_{\mathrm{m,n},\ell'}\}_{\ell'})+ \\ &\sum_{\mathrm{k}'\neq\mathrm{k}}\mathfrak{m}_{W_{\mathrm{m,n,k}'}\to\mathsf{g}_{\mathrm{m,n}}}(W_{\mathrm{m,n,k}'})^{(\mathrm{t})}\Big)\end{aligned} \tag{23}$$

However, since $\mathsf{g}(.)$ is a high-order factor (*i.e.*, depends on many variables), this naive update has an exponential cost in K. Fortunately, by exploiting the special form of $\mathsf{g}(.)$, we can reduce this cost to linear in K.

In evaluating $\mathsf{g}(\{W_{\mathrm{m,n,k}}\}_{\mathrm{k}})$ two scenarios are conceivable:

1) at least one of $W_{\mathrm{m,n,1}},\ldots,W_{\mathrm{m,n,K}}$ is non-zero – that is $\bigvee_{\mathrm{k}}W_{\mathrm{m,n,k}}=1$ and $\mathsf{g}(W_{\mathrm{m,n,k}})$ evaluates to $p^{\mathbf{O}}{}_{\mathrm{m,n}}(O_{\mathrm{m,n}}\mid 1)$.
2) $\bigvee_{\mathrm{k}}W_{\mathrm{m,n,k}}=0$ and $\mathsf{g}(W_{\mathrm{m,n,k}})$ evaluates to $p^{\mathbf{O}}{}_{\mathrm{m,n}}(O_{\mathrm{m,n}}\mid 0)$.

We can divide the maximization of (23) into two separate maximization operations over sets of assignments depending on the conditioning above and select the maximum of the two.

For simplicity, let $\mathfrak{m}_1(W_1), \ldots, \mathfrak{m}_{\mathrm{K}}(W_{\mathrm{K}})$ denote $\mathfrak{m}_{W_{\mathrm{m,n,1}} \to \mathfrak{g}_{\mathrm{m,n}}}(W_{\mathrm{m,n,1}})^{(\mathrm{t})}, \ldots, \mathfrak{m}_{W_{\mathrm{m,n,K}} \to \mathfrak{g}_{\mathrm{m,n}}}(W_{\mathrm{m,n,K}})^{(\mathrm{t})}$ respectively. W.L.O.G., let us assume the objective is to calculate the outgoing message to the first variable $\mathfrak{m}'_1(W_1) = \mathfrak{m}_{\mathfrak{g}_{\mathrm{m,n}} \to W_{\mathrm{m,n,1}}}(W_{\mathrm{m,n,1}})^{(\mathrm{t+1})}$. Let us rewrite (23) using this notation:

$$\mathfrak{m}'_1(W_1) = \max_{W_2 \ldots W_{\mathrm{K}}} \left(\mathfrak{g}_{\mathrm{m,n}}(\{W_{\mathrm{k}}\}) + \sum_{\mathrm{k}'>1} \mathfrak{m}_{\mathrm{k}'}(W_{\mathrm{k}'}) \right)$$

For $W_1 = 1$, regardless of assignments to $W_2, \ldots, W_{\mathrm{K}}$, we have $\bigvee_{\mathrm{k}} W_{\mathrm{m,n,k}} = 1$ and therefore the maximization above simplifies to

$$\begin{aligned}\mathfrak{m}'_1(1) &= \max_{W_2 \ldots W_{\mathrm{K}}} \left(\log(p^{\mathbf{O}}{}_{\mathrm{m,n}}(O_{\mathrm{m,n}} \mid 1)) \sum_{\mathrm{k}'>1} \mathfrak{m}_{\mathrm{k}'}(W_{\mathrm{k}'}) \right) \\ &= \log(p^{\mathbf{O}}{}_{\mathrm{m,n}}(O_{\mathrm{m,n}} \mid 1)) + \sum_{\mathrm{k}'>1} \max(\mathfrak{m}_{\mathrm{k}'}(0), \mathfrak{m}_{\mathrm{k}'}(1)).\end{aligned}$$

For $W_1 = 0$, if $\forall_{\mathrm{k}'>1} W_{\mathrm{k}'} = 0$ then $\mathfrak{g}(\{W_{\mathrm{k}}\})$ evaluates to $\log(p^{\mathbf{O}}{}_{\mathrm{m,n}}(O_{\mathrm{m,n}} \mid 0)$, and otherwise it evaluates to $\log(p^{\mathbf{O}}{}_{\mathrm{m,n}}(O_{\mathrm{m,n}} \mid 1)$. We need to choose the maximum over these two cases. Note that in the second case we have to ensure at least one of the remaining variables is non-zero – *i.e.*, $\exists_{\mathrm{k}'>1} W_{\mathrm{k}'} = 1$. In the following update to enforce this constraint we use

$$\mathrm{k}^* = \arg_{\mathrm{k}'>1} \max \mathfrak{m}_{\mathrm{k}'}(1) - \mathfrak{m}_{\mathrm{k}'}(0) \tag{24}$$

to get

$$\begin{aligned}\mathfrak{m}'_1(0) = \max \Bigg(& \log(p^{\mathbf{O}}{}_{\mathrm{m,n}}(O_{\mathrm{m,n}} \mid 0) + \sum_{\mathrm{k}'>1} \mathfrak{m}_{\mathrm{k}'}(0) \,, \\ & \log(p^{\mathbf{O}}{}_{\mathrm{m,n}}(O_{\mathrm{m,n}} \mid 1) + \mathfrak{m}_{\mathrm{k}^*} + \\ & \sum_{\mathrm{k}'>1, \mathrm{k}' \neq \mathrm{k}^*} \max(\mathfrak{m}_{\mathrm{k}'}(0), \mathfrak{m}_{\mathrm{k}'}(1)) \Bigg)\end{aligned}$$

where, choosing $W_{\mathrm{k}^*} = 1$ maximizes the second case (where at least one $W_{\mathrm{k}'}$ for $\mathrm{k}' > 1$ is non-zero).

As before, let us assume that the incoming messages are normalized such that $\forall_{\mathrm{k}'} \mathfrak{m}_{\mathrm{k}'}(0) = 0$, and therefore $\hat{\Gamma}_{\mathrm{m,n,k}'} = \mathfrak{m}_{\mathrm{k}'}(1)$. The normalized outgoing message is

$$\begin{aligned}\Gamma_{\mathrm{m,n,1}} &= \mathfrak{m}'_1(1) - \mathfrak{m}'(0) = \log(p^{\mathbf{O}}{}_{\mathrm{m,n}}(O_{\mathrm{m,n}} \mid 1)) \\ &+ \sum_{\mathrm{k}'>1} \max(0, \mathfrak{m}_{\mathrm{k}'}(1)) - \\ &\max \Bigg(\log(p^{\mathbf{O}}{}_{\mathrm{m,n}}(O_{\mathrm{m,n}} \mid 0), \log(p^{\mathbf{O}}{}_{\mathrm{m,n}}(O_{\mathrm{m,n}} \mid 1) + \mathfrak{m}_{\mathrm{k}^*} \\ &+ \sum_{\mathrm{k}'>1, \mathrm{k}' \neq \mathrm{k}^*} \max(0, \mathfrak{m}_{\mathrm{k}'}(1)) \Bigg) \\ &= \min \Bigg(\log(p^{\mathbf{O}}{}_{\mathrm{m,n}}(O_{\mathrm{m,n}} \mid 1)) - \log(p^{\mathbf{O}}{}_{\mathrm{m,n}}(O_{\mathrm{m,n}} \mid 0) \\ &+ \sum_{\mathrm{k}'>1} \max(0, \mathfrak{m}_{\mathrm{k}'}(1)), \max(-\mathfrak{m}_{\mathrm{k}^*}(1), 0) \Bigg) \\ &= \min \Bigg(\sum_{\mathrm{k}'>1} \max(0, \hat{\Gamma}^{(\mathrm{t})}_{\mathrm{m,n,k}'}) \\ &+ \log \left(\frac{p^{\mathbf{O}}{}_{\mathrm{m,n}}(O_{\mathrm{m,n}} \mid 1)}{p^{\mathbf{O}}{}_{\mathrm{m,n}}(O_{\mathrm{m,n}} \mid 0)} \right), \max(0, -\max_{\mathrm{k}>1} \hat{\Gamma}^{(\mathrm{t})}_{\mathrm{m,n,k}'}) \Bigg)\end{aligned}$$

where in the last step we used the definition of factor g and (24) that defines $\mathfrak{m}_{\mathrm{k}^*}(1)$. This produces the simplified form of BP messages for the update (5f) in our algorithm.

Appendix B
Marginal-MAP

While the message passing for MAP inference approximates the "jointly" optimal assignment to $\mathbf{X}$ and $\mathbf{Y}$ in the Bayesian setting, the marginals $p(X_{\mathrm{m,k}} \mid \mathbf{O})$ and $p(X_{\mathrm{k,n}} \mid \mathbf{O})$ are concerned with optimal assignments to "individual" $X_{\mathrm{m,k}}$ and $Y_{\mathrm{k,n}}$ for each $\mathrm{m,n}$ and k. Here again, message passing can approximate the log-ratio of these marginals.

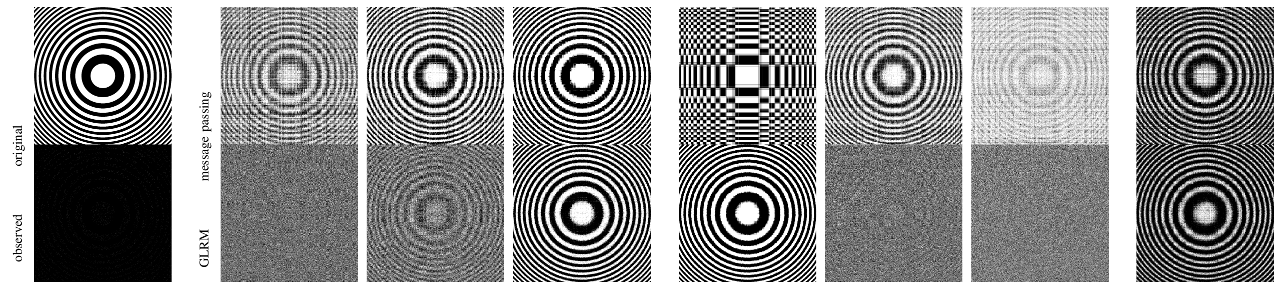
We use the function $\phi(a) = \log(1 + \exp(a))$ and its inverse $\phi^{-1}(b) = \log(\exp(b) - 1)$ in the following updates for marginalization.

$$\begin{aligned}\Phi^{(\mathrm{t+1})}_{\mathrm{m,n,k}} &:= \Gamma^{(\mathrm{t})}_{\mathrm{m,n,k}} + \hat{\Psi}^{(\mathrm{t})}_{\mathrm{m,n,k}} - \\ &\log\left(1 + \exp(\hat{\Psi}^{(\mathrm{t})}_{\mathrm{m,n,k}}) + \exp(\hat{\Phi}^{(\mathrm{t})}_{\mathrm{m,n,k}})\right) \\ \Psi^{(\mathrm{t+1})}_{\mathrm{m,n,k}} &:= \Gamma^{(\mathrm{t})}_{\mathrm{m,n,k}} + \hat{\Phi}^{(\mathrm{t})}_{\mathrm{m,n,k}} - \\ &\log\left(1 + \exp(\hat{\Phi}^{(\mathrm{t})}_{\mathrm{m,n,k}} + \exp(\hat{\Psi}^{(\mathrm{t})}_{\mathrm{m,n,k}})\right) \\ \hat{\Phi}^{(\mathrm{t+1})}_{\mathrm{m,n,k}} &:= \log\left(\frac{p^{\mathbf{X}}{}_{\mathrm{m,k}}(1)}{p^{\mathbf{X}}{}_{\mathrm{m,k}}(0)}\right) + \sum_{\mathrm{n}' \neq \mathrm{n}} \Phi^{(\mathrm{t})}_{\mathrm{m,n',k}} \\ \hat{\Psi}^{(\mathrm{t+1})}_{\mathrm{m,n,k}} &:= \log\left(\frac{p^{\mathbf{Y}}{}_{\mathrm{n,k}}(1)}{p^{\mathbf{Y}}{}_{\mathrm{n,k}}(0)}\right) + \sum_{\mathrm{m}' \neq \mathrm{m}} \Psi^{(\mathrm{t})}_{\mathrm{m',n,k}} \\ \hat{\Gamma}^{(\mathrm{t+1})}_{\mathrm{m,n,k}} &:= \hat{\Phi}^{(\mathrm{t})}_{\mathrm{m,n,k}} + \hat{\Psi}^{(\mathrm{t})}_{\mathrm{m,n,k}} \\ \Gamma^{(\mathrm{t+1})}_{\mathrm{m,n,k}} &:= \sum_{\mathrm{k}' \neq \mathrm{k}} \phi(\hat{\Gamma}^{(\mathrm{t})}_{\mathrm{m,n,k}'}) + \log\left(\frac{p^{\mathbf{O}}{}_{\mathrm{m,n}}(O_{\mathrm{m,n}} \mid 1)}{p^{\mathbf{O}}{}_{\mathrm{m,n}}(O_{\mathrm{m,n}} \mid 0)}\right. \\ &- \phi\Bigg(\phi^{-1}\Big(\sum_{\mathrm{k}' \neq \mathrm{k}} \phi(\hat{\Gamma}^{(\mathrm{t})}_{\mathrm{m,n,k}'})\Big) \\ &+ \log\left(\frac{p^{\mathbf{O}}{}_{\mathrm{m,n}}(O_{\mathrm{m,n}} \mid 1)}{p^{\mathbf{O}}{}_{\mathrm{m,n}}(O_{\mathrm{m,n}} \mid 0)}\right)\Bigg)\end{aligned}$$

Here, again using (6), we can recover $\mathbf{X}$ and $\mathbf{Y}$ from the marginals. However, due to the symmetry of the set of solutions, one needs to perform *decimation* to obtain an assignment to $\mathbf{X}$ and $\mathbf{Y}$. Decimation is the iterative process of running message passing then fixing the most biased variable – *e.g.*, an $X_{\mathrm{m,k}} \in \arg_{\mathrm{m,k}} \max |\Xi_{\mathrm{m,k}}|$ – after each convergence. While a simple randomized initialization of messages is often enough to break the symmetry of the solutions in max-sum inference, in the sum-product case one has to repeatedly fix a new subset of most biased variables.

Appendix C
Image completion

Figure 6 is an example of completing a 1000×1000 black and white image, here using message passing or GLRM. In fig. 6**(a)** we vary the number of observed pixels $\rho \in \{.01, .02, .05\}$ with fixed $\mathrm{K} = 10$ and in fig. 6**(b)** we vary the rank $\mathrm{K} \in \{2, 20, 200\}$, while fixing $\rho = .02$. A visual inspection of reconstructions suggests that, since GLRM is using real factors, it can easily over-fit the observation as we increase the rank. However, the Boolean factorization, despite being expressive, does not show over-fitting behavior for larger rank values – as if the result was regularized. In fig. 6**(c)**, we regularize both methods for $\mathrm{K} = 20$: for GLRM we use Gaussian priors over both $\mathbf{X}$ and $\mathbf{Y}$ and for message passing we use sparsity inducing priors $p^{\mathbf{X}}{}_{\mathrm{m,k}}(0) = p^{\mathbf{X}}{}_{\mathrm{m,k}}(0) = .9$. This improves the performance of both methods. However, note that regularization does not significantly improve the results of GLRM when applied to the matrix completion task, where the underlying factors are known to be Boolean (see fig. 2(right)).



(a) $\rho \in \{.01,\ .02,\ .05\}$, $K = 10$ (b) $K \in \{2,\ 20,\ 200\}$, $\rho = .02$ (c) reg.

Fig. 6: *Comparison of low-rank Boolean matrix completion using 1) message passing (using Boolean factors) and 2) GLRM (using real-valued factors) for* $\mathrm{K} = 10$*. The first column shows the original image (top) and the observation for* $\rho = .01$ *(bottom).* ***(a)*** *increasing numbers of observations* ρ*;* ***(b)*** *increasing rank* K*;* ***(c)*** *using quadratic regularization for GLRM and sparsity inducing priors* $p^{\mathbf{X}}{}_{\mathrm{m,k}}(0) = p^{\mathbf{X}}{}_{\mathrm{m,k}}(0) = .9$ *for message passing. Here* $\mathrm{K} = 20$ *and* $\rho = .02$ *–* i.e.*, similar to the figure (b) middle.*