Spectral Approximate Inference

Sejun Park [1] **Eunho Yang** [2 3 4] **Se-Young Yun** [3 5] **Jinwoo Shin** [1 3 4]

Abstract

Given a graphical model (GM), computing its partition function is the most essential inference task, but it is computationally intractable in general. To address the issue, iterative approximation algorithms exploring certain local structure/consistency of GM have been investigated as popular choices in practice. However, due to their local/iterative nature, they often output poor approximations or even do not converge, e.g., in low-temperature regimes (hard instances of large parameters). To overcome the limitation, we propose a novel approach utilizing the global spectral feature of GM. Our contribution is two-fold: (a) we first propose a fully polynomial-time approximation scheme (FPTAS) for approximating the partition function of GM associating with a low-rank coupling matrix; (b) for general high-rank GMs, we design a spectral mean-field scheme utilizing (a) as a subroutine, where it approximates a high-rank GM into a product of rank-1 GMs for an efficient approximation of the partition function. The proposed algorithm is more robust in its running time and accuracy than prior methods, i.e., neither suffers from the convergence issue nor depends on hard local structures, as demonstrated in our experiments.

1. Introduction

Graphical models (GMs) provide a succinct representation of a joint probability distribution over a set of random variables by encoding their conditional dependencies in graphical structures. GMs have been studied in various fields of machine learning, including computer vision (Freeman et al., 2000), speech recognition (Bilmes, 2004) and deep learning (Salakhutdinov & Larochelle, 2010). Most inference problems arising in GMs, e.g., obtaining desired samples and computing marginal distributions, can be easily reduced to computing their partition function (normalizing constant). However, computing the partition function is #P-hard in general even to approximate (Jerrum & Sinclair, 1993), which is thus a fundamental barrier for inference tasks of GM.

Variational inference is one of the most popular heuristics in practice for estimating the partition function. It is typically achieved via running iterative local message-passing algorithms, e.g., mean-field approximation (Parisi, 1988; Jain et al., 2018) and belief propagation (Pearl, 1982; Wainwright et al., 2005). Markov chain Monte Carlo (MCMC) method (Neal, 2001; Efthymiou et al., 2016) is another popular approach, where it usually samples from GMs via Markov chains with a local transition, e.g., Gibbs sampler (Geman & Geman, 1984), and estimates a target expectation by averaging over samples. Unfortunately, both variational and MCMC methods are hard to guarantee the convergence/mixing under some fixed computation budget and known to output poor approximation in the low-temperature regime, i.e., large parameters of GM, due to the non-existence of the so-called correlation decay (Weitz, 2006; Bandyopadhyay & Gamarnik, 2008). On the other hand, variable elimination (Dechter, 1999; Dechter & Rish, 2003; Liu & Ihler, 2011; Xue et al., 2016; Wrigley et al., 2017; Ahn et al., 2018a;b) is one of popular 'convergence free' methods for approximating the partition function. At each step, it sequentially marginalizes a chosen variable and generates complex high-order factors approximating the marginalized variable and its associated factors. Hence, it guarantees to terminate after marginalizing all variables. However, the performance of variable elimination schemes is also significantly degraded in the low-temperature regime, due to its local/iterative nature of processing variables one by one.

Contribution. In this paper, we propose a completely new approach by investigating the global information of GM, to overcome the limitation of prior methods. To this end, we study the spectral feature of the coupling matrix of GM and propose a partition function approximation algorithm utilizing the eigenvectors and eigenvalues. In particular, if the matrix-rank and parameters of GM are bounded, i.e., $O(1)$, then we prove that the proposed algorithm is a fully

[1] School of Electrical Engineering, KAIST, Daejeon, Korea [2] School of Computing, KAIST, Daejeon, Korea [3] Graduate School of AI, KAIST, Daejeon, Korea [4] AITRICS, Seoul, Korea [5] Department of Industrial & System Engineering, KAIST, Daejeon, Korea. Correspondence to: Jinwoo Shin <jinwoos@kaist.ac.kr>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019. Copyright 2019 by the author(s).

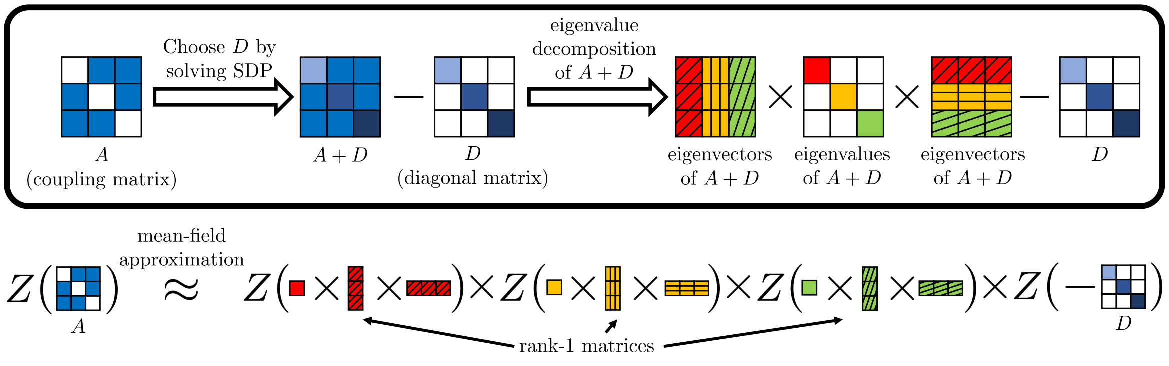


Figure 1. An illustration of the proposed partition function approximation scheme.

polynomial-time approximation scheme (FPTAS), even for GMs with high treewidth. Such polynomial-time approximation schemes have been typically investigated in the literature under certain structured GMs (Temperley & Fisher, 1961; Pearl, 1982; Dechter, 1999; Jerrum et al., 2004), and high-temperature regimes (Zhang et al., 2011; Li et al., 2013; Patel & Regts, 2017) or homogeneity of GM parameters (Jerrum & Sinclair, 1993; Sinclair et al., 2014; Molkaraie, 2016; Liu et al., 2017; Patel & Regts, 2017; Molkaraie & Gómez, 2018). Our theoretical result provides a new class of GMs for the direction.

Despite the theoretical value of the proposed algorithm for low-rank GMs, it is very expensive to run for general high-rank GMs as its complexity grows exponentially with respect to the rank. To address this issue, we decompose the partition function of high-rank GM into a product of those of rank-1 GMs. Then, we run the proposed FPTAS algorithm to compute all rank-1 partition functions and combine them to approximate the original partition function. For improving our approximation, we additionally suggest running a semi-definite programming to discover a better spectral decomposition of the partition function. In a sense, our approach is of mean-field type, but different from the traditional ones decomposing GM itself without spectral pre-processing. We present an illustration of the proposed scheme in Figure 1.

The proposed mean-field scheme can be universally applied to any GMs without the rank restriction. Its computational complexity scales well for large GMs without suffering from the convergence issue. Furthermore, its approximation quality is quite robust against hard GM instances of heterogeneous parameters since the utilized spectral feature grows linearly with respect to the inverse temperature, i.e., scale of parameters. Our experiments demonstrate that the proposed scheme indeed outperforms mean-field approximation, belief propagation and variable elimination, in particular, significantly in the low-temperature regimes where the prior methods fail.

2. Spectral Inference for Low-Rank GMs

We begin with introducing the definition of the pairwise binary graphical model (GM). Given a vector $\boldsymbol{\theta} \in \mathbb{R}^n$ and a symmetric matrix $A \in \mathbb{R}^{n \times n}$, we define GM as the following joint distribution on $\mathbf{x} \in \Omega := \{-1, 1\}^n$:

$$\mathbb{P}(\mathbf{x}) = \frac{1}{Z} \exp\left(\langle \boldsymbol{\theta}, \mathbf{x} \rangle + \mathbf{x}^T A \mathbf{x}\right) \tag{1}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product and Z is the normalizing constant. The above definition of GM coincides with the following conventional definition associating with an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ defined as:

$$\mathbb{P}(\mathbf{x}) \propto \exp\left(\sum_{i \in \mathcal{V}} \theta_i x_i + 2 \sum_{(i,j) \in \mathcal{E}} A_{ij} x_i x_j\right). \tag{2}$$

where $\mathcal{V} = \{1, \ldots, n\}$ and $\mathcal{E} = \{(i,j) : A_{ij} \neq 0,\ i < j\}$.

The normalizing constant Z of (1) is called the partition function defined as follows:

$$Z = Z(\boldsymbol{\theta}, A) := \sum_{\mathbf{x} \in \Omega} \exp\left(\langle \boldsymbol{\theta}, \mathbf{x} \rangle + \mathbf{x}^T A \mathbf{x}\right). \tag{3}$$

Computing Z is one of the most essential inference tasks arising in GMs. However, it is known to be computationally intractable in general, i.e., #P-hard even to approximate (Jerrum & Sinclair, 1993). In particular, the case when the magnitudes of entries of A are large is called, the low-temperature regime (Sykes et al., 1965), where Z is known to be harder to approximate provably (Sly & Sun, 2012; Galanis et al., 2014). This is indeed the regime where known heuristics also fail badly.

In this section, we show that Z is possible to be approximated in polynomial-time if there exists a diagonal matrix D such that the rank of $A + D$ is bounded, i.e., $O(1)$. Just for clarity, we primarily focus on the case when A is of low-rank itself (i.e., $D = 0$) and then describe at the end of this section how our results are extended to the case when $A + D$ is of low-rank for any diagonal matrix D.

2.1. Overall Approach: Approximate Inference via Spectral Decomposition

To design such a polynomial-time algorithm, we first reformulate Z using the eigenvalues/eigenvectors of A as follows:

$$\begin{aligned} Z &= \sum_{\mathbf{x}\in\Omega} \exp\left(\langle \boldsymbol{\theta}, \mathbf{x}\rangle + \mathbf{x}^T A \mathbf{x}\right) \\ &= \sum_{\mathbf{x}\in\Omega} \exp\left(\langle \boldsymbol{\theta}, \mathbf{x}\rangle + \sum_{j=1}^{r} \lambda_j \langle \mathbf{v}_j, \mathbf{x}\rangle^2\right) \qquad (4) \end{aligned}$$

where λ_j and $\mathbf{v}_j$ denote the j-th largest non-zero eigenvalue and its corresponding unit eigenvector of A and r denotes the rank of A. We note that such a decomposition is always possible because A is a real symmetric matrix, i.e., all eigenvalues are real. However, even with a small rank r, a naive computation of Z is still intractable as it is a summation over exponentially many terms. Our main idea is approximating $\lambda_j \langle \mathbf{v}_j, \mathbf{x}\rangle^2$ in (4) to its quantized value in order to drastically reduce the number of summations. Toward this, we rewrite (4) as

$$Z = \sum_{\mathbf{x}\in\Omega} \exp\left(\langle \boldsymbol{\theta}, \mathbf{x}\rangle + \sum_{j=1}^{r} \mathrm{sign}(\lambda_j) \langle \mathbf{u}_j, \mathbf{x}\rangle^2\right)$$

where $\mathrm{sign}(\lambda_j) \in \{-1, 1\}$ denotes the sign of λ_j and $\mathbf{u}_j = \sqrt{|\lambda_j|}\mathbf{v}_j$. Here we deliberately choose some mapping $f_j : \Omega \to \mathbb{Z}$ (it will be explicitly described in Section 2.2) so that $c \cdot f_j(\mathbf{x}) \approx \langle \mathbf{u}_j, \mathbf{x}\rangle$ for some fixed constant $c > 0$ and hence Z can be nicely approximated as

$$\sum_{\mathbf{x}\in\Omega} \exp\left(\langle \boldsymbol{\theta}, \mathbf{x}\rangle\right) \exp\left(\sum_{j=1}^{r} \mathrm{sign}(\lambda_j) \big(c \cdot f_j(\mathbf{x})\big)^2\right). \qquad (5)$$

Note that c decides a quantization interval and $c \cdot f_j(\mathbf{x})$ represents a quantized value of $\langle \mathbf{u}_j, \mathbf{x}\rangle$. Namely, for each $\mathbf{x} \in \Omega$, we will design $\mathbf{f}(\mathbf{x}) = [f_j(\mathbf{x})]_{j=1}^r \in \mathbb{Z}^r$ for approximating $\langle \mathbf{u}_j, \mathbf{x}\rangle$ for all j.

Given such $\mathbf{f}$, we further process (5) as

$$\begin{aligned} &\sum_{\mathbf{x}\in\Omega} \exp\left(\langle \boldsymbol{\theta}, \mathbf{x}\rangle\right) \exp\left(\sum_{j=1}^{r} \mathrm{sign}(\lambda_j) \big(c \cdot f_j(\mathbf{x})\big)^2\right) \\ &= \sum_{\mathbf{k}\in\mathbf{f}(\Omega)} \left(\sum_{\mathbf{x}\in\mathbf{f}^{-1}(\mathbf{k})} \exp\left(\langle \boldsymbol{\theta}, \mathbf{x}\rangle\right)\right) \\ &\qquad\qquad \times \exp\left(\sum_{j=1}^{r} \mathrm{sign}(\lambda_j)(c \cdot k_j)^2\right) \\ &= \sum_{\mathbf{k}\in\mathbf{f}(\Omega)} t(\mathbf{k}) \exp\left(\sum_{j=1}^{r} \mathrm{sign}(\lambda_j)(c \cdot k_j)^2\right). \qquad (6) \end{aligned}$$

In the above, the first equality is from replacing the summation over Ω by that over $\mathbf{f}(\Omega)$, i.e., for $\mathbf{k} = [k_j]_{j=1}^r \in \mathbf{f}(\Omega)$, each k_j represents a possible value of $f_j(\mathbf{x})$. For the second equality, we define $t(\mathbf{k}) := \sum_{\mathbf{x}\in\mathbf{f}^{-1}(\mathbf{k})} \exp\big(\langle \boldsymbol{\theta}, \mathbf{x}\rangle\big)$. Finally, from (5) and (6), one can observe that if $t(\mathbf{k})$ is easy to compute and the cardinality of $\mathbf{f}(\Omega)$ is small, then the partition function Z can be efficiently approximated. In the following section, we provide more details on how to choose $\mathbf{f}$ for the desired property.

2.2. How to Choose f and Compute $t(\mathbf{k})$

Choice of f. A naive choice of $\mathbf{f}$ can be

$$f_j(\mathbf{x}) = \arg\min_{k_j\in\mathbb{Z}} |c \cdot k_j - \langle \mathbf{u}_j, \mathbf{x}\rangle| \qquad (7)$$

for all j. However, with the above choice of $\mathbf{f}$, it is unclear how to compute $t(\mathbf{k})$ efficiently (in polynomial-time). To address the issue, we propose a recursive construction of $\mathbf{f}$ by relaxing (7): we iteratively define $\mathbf{f}(\mathbf{x})$ for $\mathbf{x} \in \mathcal{S}_i \setminus \mathcal{S}_{i-1}$ where $\mathcal{S}_i := \{\mathbf{x} \in \Omega : x_\ell = -1,\ \forall \ell > i\}$ so that

$$\{(-1, \ldots, -1)\} = \mathcal{S}_0 \subset \mathcal{S}_1 \subset \cdots \subset \mathcal{S}_{n-1} \subset \mathcal{S}_n = \Omega.$$

First, we define $\mathbf{f}$ for $\mathcal{S}_0$ following (7):

$$f_j\big((-1, \ldots, -1)\big) := \arg\min_{k_j\in\mathbb{Z}} \left| c \cdot k_j + \sum_{i=1}^{n} u_{ji} \right|, \qquad (8)$$

for all j. The construction of $\mathbf{f}$ for the rest $\mathcal{S}_n \setminus \mathcal{S}_0$ will be done in a recursive manner. Suppose that $\mathbf{f}(\mathbf{x})$ is defined for $\mathbf{x} \in \mathcal{S}_{i-1}$. Then, we define $\mathbf{f}$ for $\mathbf{x} \in \mathcal{S}_i \setminus \mathcal{S}_{i-1}$ as follow:

$$f_j(\mathbf{x}) := f_j(\mathbf{x}') + \widehat{u}_{ji} \qquad (9)$$

where we define $\widehat{u}_{ji} := \arg\min_{\widehat{u}_{ji}\in\mathbb{Z}} |c \cdot \widehat{u}_{ji} - 2u_{ji}|$ and $\mathbf{x}' \in \mathcal{S}_{i-1}$ such that $x'_\ell = x_\ell$ except for $\ell = i$, i.e., $x'_i = -1$. Here, (9) is motivated by the following approximation: $c \cdot f_j(\mathbf{x}') \approx \langle \mathbf{u}_j, \mathbf{x}'\rangle$ and the definition of $\widehat{u}_{ji}$ implies that

$$c \cdot (f_j(\mathbf{x}') + \widehat{u}_{ji}) \approx \langle \mathbf{u}_j, \mathbf{x}'\rangle + 2u_{ji} = \langle \mathbf{u}_j, \mathbf{x}\rangle$$

where the equality is due to $x_i = 1$ and $x'_i = -1$.

In essence, we have so far constructed $\mathbf{f}$ via a dynamic programming to approximate (7), which allows us to compute $t(\mathbf{k})$ efficiently. Furthermore, our choice of $\mathbf{f}$ ensures that $|\mathbf{f}(\Omega)|$ is bounded. Before describing how to compute $t(\mathbf{k})$, let us discuss the bound of $|\mathbf{f}(\Omega)|$. For bounding $|\mathbf{f}(\Omega)|$, we discover a bounded set $\mathcal{B} \subset \mathbb{Z}^r$ so that $\mathbf{f}(\Omega) \subset \mathcal{B}$ instead of characterizing $|\mathbf{f}(\Omega)|$ directly. We explicitly describe such $\mathcal{B}$ as follows.

Claim 1. $\mathbf{f}(\Omega) \subset \mathcal{B}$ *where*

$$\begin{aligned} \mathcal{B} &:= \prod_{j=1}^{r} \{-b_j, -b_j + 1, \ldots, b_j - 1, b_j\}, \\ b_j &:= \big\lceil \|\mathbf{u}_j\|_1/c + (n+1)/2 \big\rceil. \end{aligned}$$

Furthermore, $|\mathcal{B}|$ is bounded by

$$|\mathcal{B}| \leq 2^r \prod_{j=1}^{r} \left(\frac{1}{c}\sqrt{|\lambda_j| n} + \frac{n}{2} + 1 \right).$$

We present the proof of Claim 1 in the supplementary material. Finally, given $t(\mathbf{k})$ and $\mathcal{B}$ as defined in Claim 1, we approximate the partition function Z as follows (see (5) and (6)):

$$Z \approx \sum_{\mathbf{k}\in\mathcal{B}} t(\mathbf{k}) \exp\left(\sum_{j=1}^{r} \operatorname{sign}(\lambda_j)(c \cdot k_j)^2 \right),$$

where $t(\mathbf{k}) = 0$ if $\mathbf{k} \notin \mathbf{f}(\Omega)$.

Computation of $t(\mathbf{k})$. We are now ready to describe how to compute $t(\mathbf{k})$. Since $t(\mathbf{k}) = 0$ for $\mathbf{k} \notin \mathcal{B}$, it suffices to compute $t(\mathbf{k})$ for all $\mathbf{k} \in \mathcal{B}$. Similar to the construction of $\mathbf{f}$, we recursively compute

$$t_i(\mathbf{k}) := \sum_{\mathbf{x}\in\mathbf{f}^{-1}(\mathbf{k})\cap\mathcal{S}_i} \exp\left(\langle \boldsymbol{\theta}, \mathbf{x} \rangle\right),$$

i.e., $t_n(\mathbf{k}) = t(\mathbf{k})$. The recursive computation of $t_i(\mathbf{k})$ is based on the following claim.

Claim 2. $t_i(\mathbf{k}) = t_{i-1}(\mathbf{k}) + \exp(2\theta_i) \cdot t_{i-1}(\mathbf{k} - [\widehat{u}_{ji}]_{j=1}^{r})$.

The proof of Claim 2 is presented in the supplementary material. The above claim implies that once $t_{i-1}(\mathbf{k})$ for $\mathbf{k} \in \mathcal{B}$ is obtained, $t_i(\mathbf{k})$ can be efficiently computed using $t_{i-1}(\mathbf{k})$. Here, we consider $t_{i-1}(\mathbf{k}) = 0$ for $\mathbf{k} \notin \mathcal{B}$. Initially, one can find $t_0(\mathbf{k})$ as follows:

$$t_0(\mathbf{k}) = \begin{cases} \exp\left(-\sum_{i=1}^{n} \theta_i\right) & \text{if } \mathbf{k} = \mathbf{f}\big((-1,\ldots,-1)\big) \\ 0 & \text{otherwise} \end{cases}$$

where $\mathbf{f}\big((-1,\ldots,-1)\big)$ is defined in (8).

2.3. Provable Guarantee

The succinct description of the proposed approximate inference algorithm described in Section 2.1 and 2.2 is given in Algorithm 1. We further prove the following theoretical guarantee of the algorithm.

Theorem 3. *Algorithm 1 outputs $\widehat{Z}$ such that*

$$\left|\log \frac{Z}{\widehat{Z}}\right| \leq \frac{1}{4} r c^2 (n+1)^2 + c\sqrt{n}(n+1) \sum_{j=1}^{r} \sqrt{|\lambda_j|},$$

in $O\big(n 2^r \prod_{j=1}^{r} (\sqrt{|\lambda_j| n}/c + n/2 + 1)\big)$ time.

Algorithm 1 Spectral inference for low-rank GMs

```
Input: $\boldsymbol{\theta}, \lambda_1, \ldots, \lambda_r, \mathbf{v}_1, \ldots, \mathbf{v}_r, c$
Output: $\widehat{Z}$
$\mathbf{u}_j \leftarrow \sqrt{|\lambda_j|}\mathbf{v}_j$ for all $j \in \{1, \ldots, r\}$
$b_j \leftarrow \lceil \|\mathbf{u}_j\|_1/c + (n+1)/2 \rceil$ for all $j \in \{1, \ldots, r\}$
$\mathcal{B} \leftarrow \prod_{j=1}^{r} \{-b_j, \ldots, b_j\}$
$t(\mathbf{k}) \leftarrow 0$ for all $\mathbf{k} \in \mathcal{B}$
$\boldsymbol{\ell} \leftarrow [\ell_j]_{j=1}^{r} : \ell_j \leftarrow \arg\min_{\ell_j\in\mathbb{Z}} |c \cdot \ell_j + \sum_{i=1}^{n} u_{ji}|$
$t(\boldsymbol{\ell}) \leftarrow \exp(-\sum_i \theta_i)$
for $1 \leq i \leq n$ do
    $\widehat{u}_{ji} \leftarrow \arg\min_{\widehat{u}_{ji}\in\mathbb{Z}} |c \cdot \widehat{u}_{ji} - 2u_{ji}|$
    $t'(\mathbf{k}) \leftarrow \exp(2\theta_i) t(\mathbf{k} - [\widehat{u}_{ji}]_{j=1}^{r})$ for all $\mathbf{k} \in \mathcal{B}$[1]
    $t(\mathbf{k}) \leftarrow t(\mathbf{k}) + t'(\mathbf{k})$ for all $\mathbf{k} \in \mathcal{B}$
end for
$\widehat{Z} \leftarrow \sum_{\mathbf{k}\in\mathcal{B}} t(\mathbf{k}) \exp\left(\sum_{j=1}^{r} \operatorname{sign}(\lambda_j)(c \cdot k_j)^2\right)$
```

The proof of Theorem 3 is presented in the supplementary material. As expected, a smaller quantization interval c provides a smaller error bound, but a higher complexity (and vice versa). From Theorem 3, given $\varepsilon \in (0, 1/2)$, one can check that Algorithm 1 guarantees

$$(1-\varepsilon)Z \leq \widehat{Z} \leq (1+\varepsilon)Z,$$

if we choose

$$c = \min\left(\sqrt{\frac{\varepsilon}{r}} \frac{1}{n+1}, \frac{\varepsilon}{4(\sum_j \sqrt{|\lambda_j|})\sqrt{n}(n+1)} \right).$$

Under the choice of c, the algorithm complexity becomes $O\big((\frac{9}{\varepsilon} r \max(\lambda_{\max}, 1))^r n^{2r+1}\big)$ where $\lambda_{\max} = \max_j |\lambda_j|$. Therefore, if the rank and parameters of GM are bounded, i.e., $r, A_{ij} = O(1)$ for all i, j, Algorithm 1 is a fully polynomial-time approximation scheme (FPTAS) for approximating Z.

Finally, we remark that the following simple trick allows a FPTAS for approximating the partition function of a richer class of GMs: for any diagonal matrix D, one can check

$$Z = Z(\boldsymbol{\theta}, A) = \exp\big(-\operatorname{Tr}(D)\big) \cdot Z(\boldsymbol{\theta}, A + D). \quad (10)$$

Namely, if there exists a diagonal matrix D such that the rank of $A + D$ is $O(1)$ (possibly, A is not of low-rank though), then one can run Algorithm 1 to approximate $Z(\boldsymbol{\theta}, A + D)$ and use it to derive $Z(\boldsymbol{\theta}, A)$ from (10).

3. Spectral Inference for High-Rank GMs

In the previous section, we introduced a FPTAS algorithm for approximating the partition function for the special class of low-rank GMs. However, for general (high-rank) GMs, Algorithm 1 is intractable to run as its complexity grows exponentially with respect to the rank. In this section, we

[1] Choose $t(\mathbf{k}) = 0$ for $\mathbf{k} \notin \mathcal{B}$.

address the issue by proposing a new efficient partition function approximation algorithm for general GMs of arbitrary rank. The proposed algorithm utilizes Algorithm 1 as a subroutine. Our main idea is to decompose the partition function of GM into a product of that of rank-1 GMs using the mean-field approximation, and then handle each rank-1 GM via Algorithm 1.

Throughout this section, we assume GMs with $\boldsymbol{\theta} = 0$. Such a restriction does not harm the generality of our method due to the following:

$$\begin{aligned} Z &= \sum_{\mathbf{x}\in\Omega=\{-1,1\}^n} \exp\left(\langle \boldsymbol{\theta}, \mathbf{x}\rangle + \mathbf{x}^T A \mathbf{x}\right) \\ &= \frac{1}{2} \sum_{\mathbf{x}'\in\{-1,1\}^{n+1}} \exp\left((\mathbf{x}')^T A' \mathbf{x}'\right) = \frac{1}{2} Z' \end{aligned}$$

where $A' = \begin{bmatrix} A & \frac{1}{2}\boldsymbol{\theta} \\ \frac{1}{2}\boldsymbol{\theta}^T & 0 \end{bmatrix}$ and $Z' = \sum_{\mathbf{x}'} \exp\left((\mathbf{x}')^T A' \mathbf{x}'\right)$ is the partition function of a GM with A'. Namely, computing the partition function of any GM is easily reducible to computing that of an alternative GM with $\boldsymbol{\theta} = 0$.

3.1. Overall Approach: From High-Rank to Low-Rank

To handle high-rank GMs, we first reformulate the partition function Z by substituting the summation over $\mathbf{x}$ with the expectation over $\mathbf{x}$ drawn from the uniform distribution U_Ω over Ω:

$$Z = \sum_{\mathbf{x}\in\Omega} \exp\left(\mathbf{x}^T A \mathbf{x}\right) = 2^n \mathbb{E}_{\mathbf{x}\sim U_\Omega}\left[\exp\left(\mathbf{x}^T A \mathbf{x}\right)\right]. \tag{11}$$

Then, for approximating the above expectation, we consider the following mean-field approximation via some fully factorized distribution $q(\mathbf{y}) = \prod_{j=1}^n q_j(y_j)$, where $y_j = \langle \mathbf{v}_j, \mathbf{x}\rangle$, $\mathbf{y} = [y_j]_{j=1}^n$:

$$\begin{aligned} &\mathbb{E}_{\mathbf{x}\sim U_\Omega}\left[\exp\left(\mathbf{x}^T A \mathbf{x}\right)\right] = \mathbb{E}_{\mathbf{y}\sim P_\mathcal{Y}}\left[\exp\left(\sum_{j=1}^n \lambda_j y_j^2\right)\right] \\ &\approx \mathbb{E}_{\mathbf{y}\sim q}\left[\exp\left(\sum_{j=1}^n \lambda_j y_j^2\right)\right] = \prod_{j=1}^n \mathbb{E}_{y_j\sim q_j}\left[\exp\left(\lambda_j y_j^2\right)\right], \end{aligned} \tag{12}$$

where $P_\mathcal{Y}(\mathbf{y}) := \sum_{\mathbf{x}\in\Omega\,:\,y_j=\langle\mathbf{v}_j,\mathbf{x}\rangle,\,\forall j} U_\Omega(\mathbf{x})$ for $\mathbf{y} \in \mathcal{Y}$ and $\mathcal{Y} := \left\{\mathbf{y} = [y_j = \langle \mathbf{v}_j, \mathbf{x}\rangle]_{j=1}^n : \mathbf{x} \in \Omega\right\}$. Now, we prove the following claim that the choice of $q_j(y_j) = P_\mathcal{Y}(y_j)$ (the marginal probability of the joint distribution $P_\mathcal{Y}$) is optimal for the mean-field approximation in (12), with respect to the Kullback-Leibler (KL) divergence. The proof of Claim 4 is presented in the supplementary material.

Claim 4. *$KL\big(P_\mathcal{Y}(\mathbf{y}) || \prod_{j=1}^n q_j(y_j)\big)$ is minimized when $q_j(y_j) = P_\mathcal{Y}(y_j)$ for all j.*

In summary, under the choice of $q_j(y_j) = P_\mathcal{Y}(y_j)$, we use the following approximation for Z from (11) and (12):

$$\begin{aligned} Z &= 2^n \mathbb{E}_{\mathbf{x}\sim U_\Omega}\left[\exp\left(\mathbf{x}^T A \mathbf{x}\right)\right] \\ &\approx 2^n \prod_{j=1}^n \mathbb{E}_{y_j\sim q_j}\left[\exp\left(\lambda_j y_j^2\right)\right] \\ &= 2^n \prod_{j=1}^n \mathbb{E}_{\mathbf{x}\sim U_\Omega}\left[\exp\left(\lambda_j \langle \mathbf{v}_j, \mathbf{x}\rangle^2\right)\right], \end{aligned} \tag{13}$$

where it is easy to check that $2^n \mathbb{E}_{\mathbf{x}\sim U_\Omega}\left[\exp\left(\lambda_j \langle \mathbf{v}_j, \mathbf{x}\rangle^2\right)\right]$ is equivalent to the partition function of a rank-1 GM induced by $\lambda_j, \mathbf{v}_j$ and can be efficiently approximated using Algorithm 1. We further remark that the mean-field approximation quality in (13) is expected to be better if variables $y_j = \langle \mathbf{v}_j, \mathbf{x}\rangle$ for all j are closer to independence. Hence, it is quite a reasonable approximation since for $i \neq j$, $\langle \mathbf{v}_i, \mathbf{x}\rangle$, $\langle \mathbf{v}_j, \mathbf{x}\rangle$ are pairwise uncorrelated, i.e., $\mathbb{E}_{\mathbf{x}\sim U_\Omega}[\langle \mathbf{v}_i, \mathbf{x}\rangle \langle \mathbf{v}_j, \mathbf{x}\rangle] = 0$, due to the orthogonality of eigenvectors $\mathbf{v}_i, \mathbf{v}_j$.

We remark that our mean-field approximation (13) is different from the traditional one (Parisi, 1988). The latter addresses to find a mean-field distribution of x_i's minimizing the KL divergence with the original distribution $\mathbb{P}(\mathbf{x})$, while our approach minimizes the KL divergence between $q(y_j)$ and $P_\mathcal{Y}(\mathbf{y})$, i.e., after spectral processing.

3.2. Improving (13) via Controlling the Diagonal of A

It is instructive to remind that varying the diagonal of A only changes the partition function by a constant multiplicative factor, as in (10). In order to fully utilize this, we address to optimize the diagonal of A to improve our mean-field approximation. To this end, we build the following mean-field approximation by introducing the additional freedom of choosing a diagonal matrix D:

$$\begin{aligned} &\mathbb{E}_{\mathbf{x}\sim U_\Omega}\left[\exp\left(\mathbf{x}^T A \mathbf{x}\right)\right] \\ &= \mathbb{E}_{\mathbf{x}\sim U_\Omega}\left[\exp\left(\mathbf{x}^T (A + D)\mathbf{x} - \mathrm{Tr}(D)\right)\right] \\ &= \mathbb{E}_{\mathbf{y}^D\sim P_{\mathcal{Y}^D}}\left[\exp\left(\sum_{j=1}^n \lambda_j^D (y_j^D)^2 - \mathrm{Tr}(D)\right)\right] \\ &\approx \mathbb{E}_{\mathbf{y}^D\sim q^D}\left[\exp\left(\sum_{j=1}^n \lambda_j^D (y_j^D)^2 - \mathrm{Tr}(D)\right)\right] \end{aligned} \tag{14}$$

where $\lambda_j^D, \mathbf{y}^D, q^D, \mathcal{Y}^D, P_{\mathcal{Y}^D}$ are those for $A + D$ (analogous to $\lambda_j, \mathbf{y}, q, \mathcal{Y}, P_\mathcal{Y}$ of A). Since it is intractable to find the optimal selection for D by directly minimizing the approximation gap of (14) (as computing the true expectations is intractable), we propose to set the free parameter D by solving the following semi-definite programming (SDP):

$$\max_{D} \mathrm{Tr}(D) \qquad \text{subject to} \qquad A + D \preceq 0. \tag{15}$$

The intuition behind solving (15) is provided in Section 3.3. We also provide its empirical justification through experimental studies in Section 4.2. We remark that the SDP (15) is equivalent to (the dual of) the popular semi-definite relaxation of the max-cut problem (Goemans & Williamson, 1995) and the maximum eigenvalue minimization problem (Delorme & Poljak, 1993). For the complexity of solving (15), the interior point method (Alizadeh, 1995; Helmberg et al., 1996) has $O(n^{3.5}\log(1/\varepsilon))$ running time and the first order method (Nesterov, 2007) has $O(n^3\sqrt{\log n}/\varepsilon)$ running time where $\varepsilon > 0$ denotes the target precision to the optimum.[2]

From (11), (14) and (15), our final approximation becomes

$$\begin{aligned}
Z &= 2^n \mathbb{E}_{\mathbf{x}\sim U_\Omega}\left[\exp\left(\mathbf{x}^T A \mathbf{x}\right)\right] \\
&\approx 2^n \mathbb{E}_{\mathbf{y}^D \sim q^D}\left[\exp\left(\sum_{j=1}^{n} \lambda_j (y_j^D)^2 - \mathrm{Tr}(D)\right)\right] \\
&= 2^n \exp\left(-\mathrm{Tr}(D)\right) \prod_{j=1}^{n} \mathbb{E}_{y_j^D \sim q_j^D}\left[\exp\left(\lambda_j (y_j^D)^2\right)\right] \\
&= 2^n \exp\left(-\mathrm{Tr}(D)\right) \prod_{j=1}^{n} \mathbb{E}_{\mathbf{x}\sim U_\Omega}\left[\exp\left(\lambda_j^D \langle \mathbf{v}_j^D, \mathbf{x}\rangle^2\right)\right]
\end{aligned}$$

where D is a solution of (15) and $\mathbf{v}_j^D$ is an eigenvector of $A + D$ corresponding to λ_j^D. It is trivial that the above approximation with $D = 0$ reduces to (13). Finally, we formally state the proposed algorithm in Algorithm 2.

3.3. Intuition for (15)

Now, we describe the intuition why we consider the semi-definite programming (15). To this end, let us re-write the approximation error in (14) as the following alternative view:

$$\begin{aligned}
&\left|\log(Z) - \log(\widehat{Z})\right| \\
&= \left|\log\left(\frac{\mathbb{E}_{\mathbf{y}^D \sim P_{\mathbf{y}^D}}\left[\exp\left(\sum_{j=1}^{n} \lambda_j^D (y_j^D)^2 + \mathrm{Tr}(D)\right)\right]}{\mathbb{E}_{\mathbf{y}^D \sim q^D}\left[\exp\left(\sum_{j=1}^{n} \lambda_j^D (y_j^D)^2 + \mathrm{Tr}(D)\right)\right]}\right)\right| \\
&= \left|\log\left(\frac{\mathbb{E}_{\mathbf{y}^D \sim P_{\mathbf{y}^D}}\left[\exp\left(\sum_{j=1}^{n} \lambda_j^D (y_j^D)^2\right)\right]}{\mathbb{E}_{\mathbf{y}^D \sim q^D}\left[\exp\left(\sum_{j=1}^{n} \lambda_j^D (y_j^D)^2\right)\right]}\right)\right|
\end{aligned}$$

where $\widehat{Z}$ denotes the approximated partition function. One can easily check that the approximation error is 0 when $\lambda_1^D = \cdots = \lambda_n^D = 0$. Thus, we can expect a very accurate estimation when all eigenvalues of $A + D$ are close to 0. One can also observe that if there exists $\lambda_j^D > 0$, then the error might be too huge as $\sup_{y \in \mathbb{R}^n} \sum_{j=1}^{n} \lambda_j^D y_j^2 = \infty$ and the supports of P_{y^D} and q^D are different. Under the above intuitions, we suggest to solve the following problem:

Algorithm 2 Spectral inference for high-rank GMs

```
Input: $A, c$
Output: $\widehat{Z}$
$D \leftarrow$ solution of the semi-definite programming (15)
for $1 \le j \le n$ do
    $\lambda_j^D \leftarrow j$-th largest eigenvalue of $A + D$
    $\mathbf{v}_j^D \leftarrow j$-th largest eigenvector of $A + D$
    $E_j \leftarrow 2^{-n} \times$ Algorithm 1$(\theta = 0, \lambda_j^D, \mathbf{v}_j^D, c)$
end for
$\widehat{Z} \leftarrow 2^n \times \exp\left(-\mathrm{Tr}(D)\right) \times \prod_{j=1}^{n} E_j$
```

$$\max_{D} \sum_{j=1}^{n} \lambda_j^D \qquad \text{subject to} \qquad \lambda_j^D \le 0, \quad \text{for all } j. \tag{16}$$

The optimization (16) is equivalent to (15) since $\mathrm{Tr}(D) = \sum_{j=1}^{n} \lambda_j^D - \mathrm{Tr}(A)$ and the condition $\lambda_j^D \le 0$ for all j is equivalent to $A + D \preceq 0$.

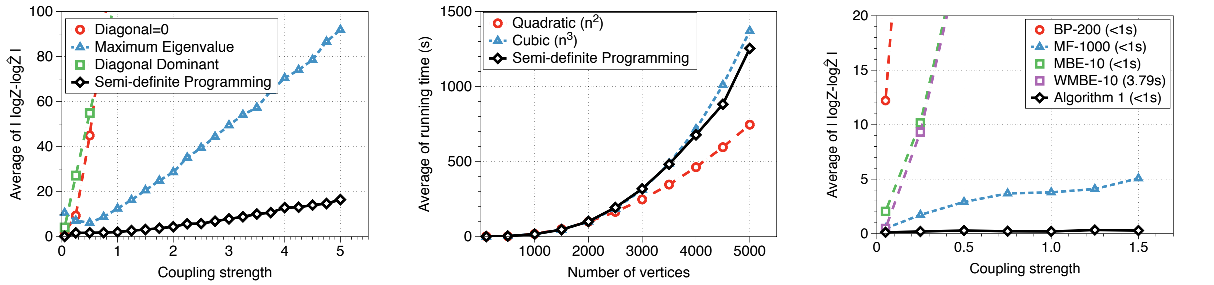
4. Experimental Results

In this section, we evaluate our algorithms on diverse environments including both synthetic and UAI datasets to corroborate our theorem and claims.

4.1. Setups

To begin with, we describe our overall experimental settings. We compare our algorithms against the standard inference schemes dominantly used in most applications: belief propagation (BP) (Pearl, 1982), mean-field approximation (MF) (Parisi, 1988), mini-bucket elimination (MBE) (Dechter & Rish, 2003) and weighted mini-bucket elimination (WMBE) (Liu & Ihler, 2011). Since all baselines are iterative methods and have the trade-off between the computation cost and the performance, we choose 200 iterations for BP, 1000 iterations for MF and 10 ibound for MBE and WMBE, for fair comparisons. Below these are referred to as BP-200, MF-1000, MBE-10 and WMBE-10, respectively. In the case of BP and MF, their performances are saturated with the above choice in most cases and there is no gain by running more iterations. On the other hand, one can improve the approximation quality of MBE and WMBE with a larger ibound. However, its complexity grows exponentially with respect to it. We also report the running times of algorithms in our implementation using round brackets following their names, e.g., BP-200 (2s) means that 200 iterations of BP run in 2 seconds (on average) for tested GMs.

[2] We also refer Section 3 of (Waldspurger et al., 2015) and Section 4 of (Goemans & Williamson, 1995) for more details.



(a) Approximation errors with different D

(b) Running time for solving (15)

(c) Approximation errors for rank-1 GMs

Figure 2. Evaluation to measure (a) the effect and (b) running time of the semi-definite programming (15). (c) Evaluation of Algorithm 1 under rank-1 GMs. In (b), quadratic and cubic denote polynomial regression curves fitted for 100-2000 vertice results.

Throughout our all experiments, we fix $c = \sqrt{|\lambda_j|}/1000$ for Algorithm 1 and Algorithm 2 to bound its running time regardless of eigenvalues. For solving the semi-definite programming (SDP) (15), we use CVX (Grant et al., 2008) with SDPT3 solver (Toh et al., 1999) using MATLAB.

For generating synthetic GMs to evaluate on, we first choose the graph structure (it will be specified in each setting) and randomly sample $\theta_i \sim \text{Unif}[-1, 1]$ on its vertices and $A_{ij} \sim \text{Unif}[-s, s]$ on its edges where Unif denotes the uniform distribution and s indicates the strength of pairwise couplings. For measuring the running time for all experiments, we run algorithms using a single thread of CPU. To reduce experimental noise, we average 100 random GMs for each plot unless otherwise stated.

4.2. Investigating the Semi-Definite Programming (15)

In this section, we investigate empirical effects and running time of the proposed SDP (15).

Effect of solving (15)**.** We first investigate how (15) helps the mean-field approximation (14) used in Algorithm 2 compared to other choices of diagonal matrix D. In particular, we consider three other choices to compare. The first choice is $D = 0$ which does not change the diagonal of A. The second choice is $D = -\max_j \lambda_j \times I$ which chooses entries of D by the maximum eigenvalue of A so that $A + D \preceq 0$. The last choice is $D_{ii} = -\sum_{j=1}^{n} |A_{ij}|$ which forces $A + D$ to be a diagonal dominant matrix, i.e., $A + D \preceq 0$. The second and third choices can be thought as feasible, yet non-optimal solutions of (15). Figure 2a reports the experimental result for measuring the log partition error for GMs on complete graph having 20 vertices. One can observe that solving (15) is important for the approximation performance of Algorithm 2.

Running time for solving (15)**.** Now, we discuss about the empirical complexity of solving (15). Our solver SDPT3 uses the primal-dual interior point method (Toh et al., 1999) for solving (15). To measure the running time of the solver, we generate random GMs on complete graphs by varying the number of vertices from 100 to 5000. Figure 2b illustrates the average running time of our solver where each point is averaged over 10 random GMs. We compare the running time of our solver with quadratic and cubic polynomials with respect to n. One can observe that the empirical running time to solve (15) is between $O(n^2)$ and $O(n^3)$, which is better than the theoretical bound of the interior point method $O(n^{3.5})$ (Helmberg et al., 1996).

4.3. Evaluation of Algorithm 1 under Low-Rank GMs

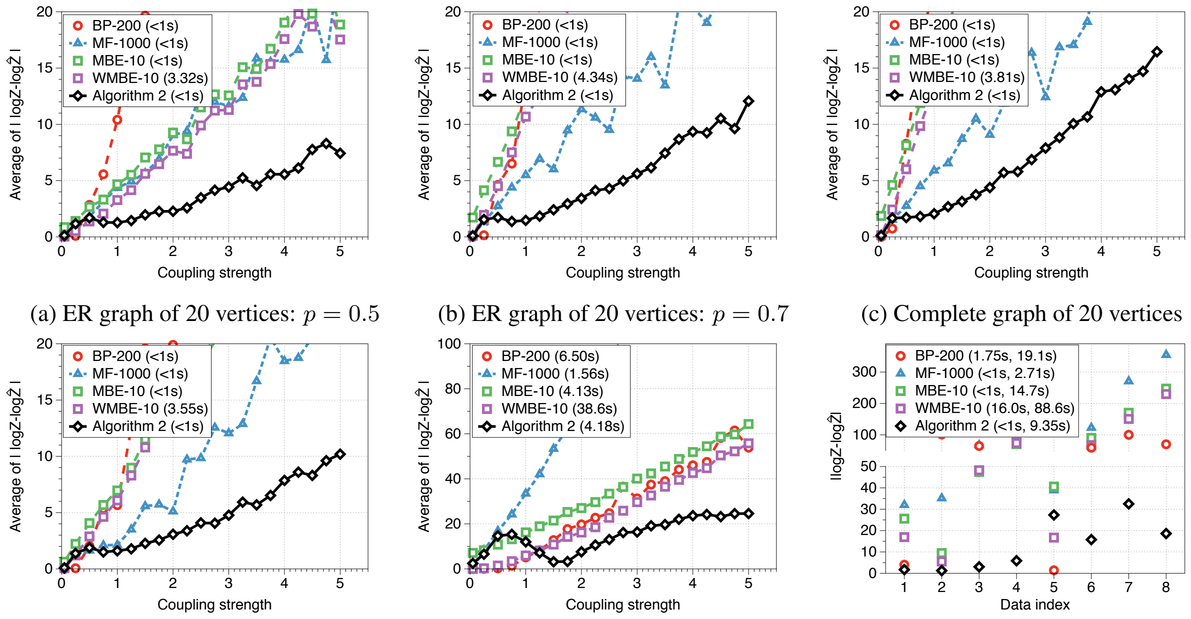
We evaluate Algorithm 1 under rank-1 GMs, which is used as a subroutine of Algorithm 2. We choose a random eigenvalue $\lambda \in \{-1, 1\}$ and a random eigenvector $\mathbf{v} \in \text{Unif}\big(\{\mathbf{v} \in \mathbb{R}^n : \|\mathbf{v}\|_2 = 1\}\big)$ to generate rank-1 GMs by choosing $A = \lambda \mathbf{v}\mathbf{v}^T$ and $\theta_i \sim \text{Unif}[-1, 1]$. Given $\mathbf{v}$, we scale λ to match the average value of $|A_{ij}|$ to be equal to some constant s (coupling strength in Figure 2c), i.e.,

$$\frac{1}{n(n+1)} \sum_{i=1}^{n} \sum_{j=1:i\neq j}^{n} |A_{ij}| = s.$$

We remark that rank-1 GMs has the special property that if its eigenvalue λ is positive (or negative), they are equivalent to ferromagnetic (or antiferromagnetic) models, i.e., $A_{ij} \geq 0$ (or $A_{ij} \leq 0$) for $i \neq j$, respectively. Figure 2c reports the algorithm performances under rank-1 GMs. As expected from our theoretical results (Theorem 3), our algorithm is nearly exact, while other algorithms fail. In particular, BP, MBE and WMBE output very poor approximation since they usually fail in antiferromagnetic cases, i.e., negative eigenvalue. The superior performance of Algorithm 1 under rank-1 GMs implies that the approximation error of Algorithm 2 would mainly come from the mean-field approximation (14).

4.4. Evaluation of Algorithm 2 under High-Rank GMs

We now evaluate the empirical performance of Algorithm 2 under synthetic high-rank GMs and UAI datasets (Gogate, 2014). In all cases, we have checked through simulations



(a) ER graph of 20 vertices: $p = 0.5$ (b) ER graph of 20 vertices: $p = 0.7$ (c) Complete graph of 20 vertices

(d) Complete bipartite graph of 20 vertices (e) Grid graph of 225 vertices (15 × 15) (f) UAI grid GMs

Figure 3. Approximation errors for GMs on various graph structures and UAI datasets. In (a)-(b), p denotes the probability to choose each edge of ER graphs. In (f) for UAI grid GMs, indices 1-4 correspond to GMs of 100 vertices (10 × 10 grid) and 5-8 correspond to GMs of 400 vertices (20 × 20 grid). In (f), average running times of algorithms for indices 1-4 and indices 5-8 are noted in $(\cdot)$, respectively.

that BP and MF do not have better accuracy than BP-200 and MF-1000, respectively, even if we run the algorithms with much longer iterations.

To generate synthetic GMs, we consider Erdős-Rényi (ER) random graphs, complete bipartite graphs, complete graphs, and grid graphs. The experimental results are reported in Figure 3a-3e. In all cases, one can observe that our algorithm significantly outperforms others in the high coupling region, i.e., the low-temperature regime. It is known that MF outputs better approximations than others as the underlying graph structure becomes dense, e.g., complete graph (Ellis & Newman, 1978), however, our algorithm remarkably performs better than MF even in such cases. In particular, MF and BP exhibit high variance on their approximation errors in high coupling regions, while ours does not.

We also evaluate our algorithms with GMs on grid graphs in a dataset for UAI 2014 inference competition. It provides 8 GMs on grid graphs, where 4 of them are of 100 vertices (10×10) and the other 4 are of 400 vertices (20×20). Figure 3f reports the approximation error and the running time of each algorithm. In the experimental results, our algorithm consistently has small errors, while other algorithms often fail badly.

Finally, we compare the running times of algorithms under GMs on complete graphs of 100-500 vertices, which are reported in Figure 4. Here, we do not report WMBE since it is slower than MBE. One can observe that Algorithm 2 scales as well as BP, while MBE does not. MF is the fastest, but it is worst in approximation quality under grid and UAI GMs, as reported in earlier experimental results.

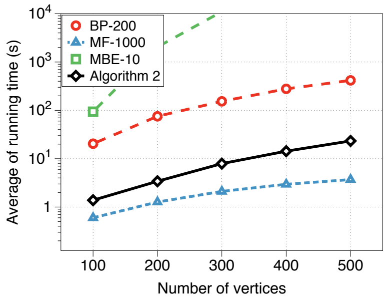


Figure 4. Running time under varying the size of GMs on complete graphs.

5. Conclusion

In this paper, we provide a completely new angle to design approximate inference algorithms for graphical models. The proposed algorithms scale well for large scale models as like prior iterative message-passing schemes, and outperforms them in approximation quality, in particular, significantly for hard instances. For the future work, we plan to extend our spectral approach to estimating the marginal distributions or/and related inference in higher-order or continuous models.

Acknowledgement

This work was supported by IITP grant funded by the Korea government (MSIT) (No.2017-0-01779, XAI). We would like to acknowledge Sungsoo Ahn for helpful discussions and sharing codes.

References

Ahn, S., Chertkov, M., Shin, J., and Weller, A. Gauged mini-bucket elimination for approximate inference. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 10–19, 2018a.

Ahn, S., Chertkov, M., Weller, A., and Shin, J. Bucket renormalization for approximate inference. In *International Conference on Machine Learning (ICML)*, pp. 109–118, 2018b.

Alizadeh, F. Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM Journal on Optimization*, 5(1):13–51, 1995.

Bandyopadhyay, A. and Gamarnik, D. Counting without sampling: Asymptotics of the log-partition function for certain statistical physics models. *Random Structures & Algorithms*, 33(4):452–479, 2008.

Bilmes, J. A. Graphical models and automatic speech recognition. In *Mathematical Foundations of Speech and Language Processing*, pp. 191–245. Springer, 2004.

Dechter, R. Bucket elimination: A unifying framework for reasoning. *Artificial Intelligence*, 113(1-2):41–85, 1999.

Dechter, R. and Rish, I. Mini-buckets: A general scheme for bounded inference. *Journal of the ACM (JACM)*, 50 (2):107–153, 2003.

Delorme, C. and Poljak, S. Laplacian eigenvalues and the maximum cut problem. *Mathematical Programming*, 62 (1-3):557–574, 1993.

Efthymiou, C., Hayes, T. P., Štefankovic, D., Vigoda, E., and Yin, Y. Convergence of MCMC and loopy BP in the tree uniqueness region for the hard-core model. In *IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 704–713. IEEE, 2016.

Ellis, R. S. and Newman, C. M. The statistics of curie-weiss models. *Journal of Statistical Physics*, 19(2):149–161, 1978.

Freeman, W. T., Pasztor, E. C., and Carmichael, O. T. Learning low-level vision. *International Journal of Computer Vision*, 40(1):25–47, 2000.

Galanis, A., Štefankovič, D., and Vigoda, E. Inapproximability for antiferromagnetic spin systems in the tree non-uniqueness region. In *Annual ACM SIGACT Symposium on Theory of computing (STOC)*, pp. 823–831. ACM, 2014.

Geman, S. and Geman, D. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 6(6):721–741, 1984.

Goemans, M. X. and Williamson, D. P. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, 42(6):1115–1145, 1995.

Gogate, V. UAI 2014 inference competition. *See www.hlt.utdallas.edu/vgogate/uai14-competition*, 2014.

Grant, M., Boyd, S., and Ye, Y. CVX: Matlab software for disciplined convex programming, 2008.

Helmberg, C., Rendl, F., Vanderbei, R. J., and Wolkowicz, H. An interior-point method for semidefinite programming. *SIAM Journal on Optimization*, 6(2):342–361, 1996.

Jain, V., Koehler, F., and Mossel, E. The mean-field approximation: Information inequalities, algorithms, and complexity. In *Conference on Learning Theory (COLT)*, pp. 1326–1347, 2018.

Jerrum, M. and Sinclair, A. Polynomial-time approximation algorithms for the Ising model. *SIAM Journal on Computing*, 22(5):1087–1116, 1993.

Jerrum, M., Sinclair, A., and Vigoda, E. A polynomial-time approximation algorithm for the permanent of a matrix with nonnegative entries. *Journal of the ACM (JACM)*, 51(4):671–697, 2004.

Li, L., Lu, P., and Yin, Y. Correlation decay up to uniqueness in spin systems. In *Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 67–84. Society for Industrial and Applied Mathematics, 2013.

Liu, J., Sinclair, A., and Srivastava, P. The Ising partition function: Zeros and deterministic approximation. In *IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 986–997. IEEE, 2017.

Liu, Q. and Ihler, A. T. Bounding the partition function using holder's inequality. In *International Conference on Machine Learning (ICML)*, pp. 849–856, 2011.

Molkaraie, M. An importance sampling algorithm for the Ising model with strong couplings. In *24th International Zurich Seminar on Communications (IZS)*. ETH-Zürich, 2016.

Molkaraie, M. and Gómez, V. Monte carlo methods for the ferromagnetic Potts model using factor graph duality. *IEEE Transactions on Information Theory*, 64(12):7449–7464, 2018.

Neal, R. M. Annealed importance sampling. *Statistics and computing*, 11(2):125–139, 2001.

Nesterov, Y. Smoothing technique and its applications in semidefinite optimization. *Mathematical Programming*, 110(2):245–259, 2007.

Parisi, G. *Statistical field theory*. Addison-Wesley, 1988.

Patel, V. and Regts, G. Deterministic polynomial-time approximation algorithms for partition functions and graph polynomials. *SIAM Journal on Computing*, 46(6):1893–1919, 2017.

Pearl, J. Reverend bayes on inference engines: a distributed hierarchical approach. In *Proceedings of the Second AAAI Conference on Artificial Intelligence*, pp. 133–136. AAAI Press, 1982.

Salakhutdinov, R. and Larochelle, H. Efficient learning of deep Boltzmann machines. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 693–700, 2010.

Sinclair, A., Srivastava, P., and Thurley, M. Approximation algorithms for two-state anti-ferromagnetic spin systems on bounded degree graphs. *Journal of Statistical Physics*, 155(4):666–686, 2014.

Sly, A. and Sun, N. The computational hardness of counting in two-spin models on d-regular graphs. In *IEEE Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 361–369. IEEE, 2012.

Sykes, M., Essam, J., and Gaunt, D. Derivation of low-temperature expansions for the Ising model of a ferromagnet and an antiferromagnet. *Journal of Mathematical Physics*, 6(2):283–298, 1965.

Temperley, H. N. and Fisher, M. E. Dimer problem in statistical mechanics-an exact result. *Philosophical Magazine*, 6(68):1061–1063, 1961.

Toh, K.-C., Todd, M. J., and Tütüncü, R. H. Sdpt3 matlab software package for semidefinite programming, version 1.3. *Optimization Methods and Software*, 11(1-4):545–581, 1999.

Wainwright, M. J., Jaakkola, T. S., and Willsky, A. S. A new class of upper bounds on the log partition function. *IEEE Transactions on Information Theory*, 51(7):2313–2335, 2005.

Waldspurger, I., dâĂŹAspremont, A., and Mallat, S. Phase recovery, maxcut and complex semidefinite programming. *Mathematical Programming*, 149(1-2):47–81, 2015.

Weitz, D. Counting independent sets up to the tree threshold. In *Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pp. 140–149. ACM, 2006.

Wrigley, A., Lee, W. S., and Ye, N. Tensor belief propagation. In *International Conference on Machine Learning (ICML)*, pp. 3771–3779, 2017.

Xue, Y., Ermon, S., Le Bras, R., Selman, B., et al. Variable elimination in the fourier domain. In *International Conference on Machine Learning (ICML)*, pp. 285–294, 2016.

Zhang, J., Liang, H., and Bai, F. Approximating partition functions of the two-state spin system. *Information Processing Letters*, 111(14):702–710, 2011.