Supplementary Material :
Variational Inference over Combinatorial Spaces

Alexandre Bouchard-Côté[*] **Michael I. Jordan**[*,†]
[*]Computer Science Division [†]Department of Statistics
University of California at Berkeley

A Extended derivations and proofs

A.1 Markov random field reformulation

We prove in this section that under the Rich Sufficient Statistics condition (RSS)[1], the log-partition function is the same in the original exponential family and in the bipartite MRF described in Section 2.2. Let us denote the latter log-partition function by $\tilde{A}(\boldsymbol{\theta})$.

We first prove the following identity, introduced in the main paper as Equation (3):

Lemma 1

$$\sum_{s_1\in\{0,1\}}\sum_{s_2\in\{0,1\}}\cdots\sum_{s_J\in\{0,1\}}\prod_{i=1}^{I}\prod_{j=1}^{J}\mathbf{1}[\phi_j(x_i)=s_j]=\begin{cases}1 & \text{if } x_1=x_2=\cdots=x_I\\ 0 & \text{otherwise.}\end{cases}$$

Proof: Suppose first that there are indices i', i'' such that $x_{i'} \neq x_{i''}$. By the RSS condition, this means that there is at least one j_0 such that $\phi_{j_0}(x_{i'}) \neq \phi_{j_0}(x_{i''})$. Since the product $\prod_{i=1}^{I}\prod_{j=1}^{J}\mathbf{1}[\phi_j(x_i)=s_j]$ contains both the factor $\mathbf{1}[\phi_{j_0}(x_{i'})=s_{j_0}]$ and $\mathbf{1}[\phi_{j_0}(x_{i''})=s_{j_0}]$, for any fixed term in the iterated sum, at least one of the two factors will be equal to zero.

Conversely, if $x = x_1 = x_2 = \cdots = x_I$, then only the multi-index $(s_1, s_2, \ldots, s_J) = (\phi_1(x), \phi_2(x), \ldots, \phi_J(x))$ in the iterated sum induces a non-zero term. ■

A slight extension of this argument yields:

Lemma 2 *For all $\boldsymbol{x} = (x_1, \ldots, x_I)$, where $x_i \in \boldsymbol{\mathcal{X}}$, we have:*

$$\begin{aligned}\Psi(\boldsymbol{x}) &= \sum_{s_1\in\{0,1\}}\sum_{s_2\in\{0,1\}}\cdots\sum_{s_J\in\{0,1\}}\left\{\prod_{i=1}^{I}\prod_{j=1}^{J}\Psi_{i,j}(x_i,s_j)\right\}\left\{\prod_{j=1}^{J}\Psi_j(s_j)\right\}\left\{\prod_{i=1}^{I}\Psi_i(x_i)\right\}\\ &= \begin{cases}\exp\{\langle\boldsymbol{\phi}(x_1),\boldsymbol{\theta}\rangle\}\,\nu(x_1) & \text{if } x_1=x_2=\cdots=x_I\\ 0 & \text{otherwise.}\end{cases}\end{aligned}$$

Using this lemma, we can prove the main proposition:

Proposition 3 *Under RSS, $\tilde{A}(\boldsymbol{\theta}) = A(\boldsymbol{\theta})$.*

[1]We make the observation in passing that the RSS condition can also be described tersely as the requirement that the σ-algebra generated by the sufficient statistics be equal to the base σ-algebra: $\sigma(\phi_1, \ldots, \phi_J) = \mathcal{F}$.

MFMF($\boldsymbol{\theta}, A_1, \ldots, A_I$)

1: $\zeta_{i,j}^{(1)} = 0$
2: **for** $t = 1, 2, \ldots, T$ **do**
3: $\quad \boldsymbol{\xi}^{(t)} = \boldsymbol{\theta} + \sum_i \boldsymbol{\zeta}_i^{(t-1)}$
4: $\quad \boldsymbol{\zeta}_i^{(t)} = \text{logit}\left(\nabla A_i\left(\boldsymbol{\xi}^{(t)}\right)\right)$
5: **end for**
6: **return** $\hat{\boldsymbol{\mu}} = \text{logistic}(\boldsymbol{\xi})$

TRWMF($\boldsymbol{\theta}, A_1, \ldots, A_I, \boldsymbol{\rho}$)

1: $\zeta_{i,j}^{(1)} = 0$
2: **for** $t = 1, 2, \ldots, T$ **do**
3: $\quad \lambda_{i,j}^{(t)} = \theta_j + \sum_{i':i' \neq i} \rho_{i' \to j} \zeta_{i',j}^{(t-1)} - \rho_{i \to j} \zeta_{i,j}^{(t-1)}$
4: $\quad \bar{\xi}_{i,j}^{(t)} = \rho_{j \to i} \lambda_{i,j}^{(t)}$
5: $\quad \delta_{i,j}^{(t)} = (1 - 2\rho_{j \to i}) \lambda_{i,j}^{(t)}$
6: $\quad \boldsymbol{\zeta}_i^{(t)} = \text{logit}\left(\nabla A_i\left(\bar{\boldsymbol{\xi}}_i^{(t)}\right)\right) - \boldsymbol{\delta}_i^{(t)}$
7: **end for**
8: **return** $\hat{\boldsymbol{\mu}} = \text{logistic}\left(\boldsymbol{\theta} + \sum_i \boldsymbol{\zeta}_i^{(T)}\right)$

Figure 4: Pseudocode for Mean Field Measure Factorization and Tree-Reweighted Measure Factorization. The vector $\boldsymbol{\rho}$ is the collection of marginals of a distribution of spanning trees over $K_{I,J}$. Note that these marginals can also be updated, see [20] for details.

Proof: We have:

$$\begin{aligned}
\exp \tilde{A}(\boldsymbol{\theta}) &= \sum_{x_1 \in \boldsymbol{\mathcal{X}}_1} \sum_{x_2 \in \boldsymbol{\mathcal{X}}_2} \cdots \sum_{x_I \in \boldsymbol{\mathcal{X}}_I} \Psi(\boldsymbol{x}) \\
&= \sum_{x_1 \in \boldsymbol{\mathcal{X}}_1} \sum_{x_2 \in \boldsymbol{\mathcal{X}}_2} \cdots \sum_{x_I \in \boldsymbol{\mathcal{X}}_I} \begin{cases} \exp\left\{\langle \boldsymbol{\phi}(x_1), \boldsymbol{\theta} \rangle\right\} \nu(x_1) & \text{if } x_1 = x_2 = \cdots = x_I \\ 0 \quad \text{otherwise} \end{cases} \\
&= \sum_{x \in \boldsymbol{\mathcal{X}}} \exp\left\{\langle \boldsymbol{\phi}(x), \boldsymbol{\theta} \rangle\right\} \nu(x) \\
&= \exp A(\boldsymbol{\theta}).
\end{aligned}$$

■

A.2 Algorithms

In this appendix, we provide more information regarding the derivation of the variational algorithms discussed in the paper.

BPMF

The BPMF algorithm maintains at each iteration the quantities $\boldsymbol{\zeta}_i, \bar{\boldsymbol{\xi}}_i$, and super-partition functions $A_i(\bar{\boldsymbol{\xi}}_i)$. Starting with $\zeta_{i,j}^{(0)} = 0$, we use the following updates at each iteration $t = 1, 2, \ldots, T$:

$$\begin{aligned}
\bar{\boldsymbol{\xi}}_i^{(t)} &= \boldsymbol{\theta} + \sum_{i':i' \neq i} \boldsymbol{\zeta}_{i'}^{(t-1)} \\
\boldsymbol{\zeta}_i^{(t)} &= \text{logit}\left(\nabla A_i\left(\bar{\boldsymbol{\xi}}_i^{(t)}\right)\right) - \bar{\boldsymbol{\xi}}_i^{(t)},
\end{aligned}$$

where the logit function of a vector $\text{logit}\, \boldsymbol{v}$ is the vector of the logit function applied to each entry of the vector $\boldsymbol{v}$, and we use the convention $(\pm\infty) - (\pm\infty) = \pm\infty$.

The approximation of the moments $\mu_j = \nabla_j A(\boldsymbol{\theta})$ is proportional to the product of all the incoming messages at the last iteration T, times the local potential, $\prod_j m_{i \to j}(s) \Psi_j(s)$:

$$\frac{\hat{\mu}_j}{1 - \hat{\mu}_j} = \frac{\prod_j m_{i \to j}^{(T)}(1) e^{\theta_j}}{\prod_j m_{i \to j}^{(T)}(0) e^0}.$$

Using the notation $\text{logistic}(\boldsymbol{v})_j = (1 + \exp(-v_j))^{-1}$, this is equivalent to:

$$\hat{\boldsymbol{\mu}} = \text{logistic}\left(\boldsymbol{\theta} + \sum_i \boldsymbol{\zeta}_i^{(T)}\right).$$

TRWMF

We now derive Equation (7) in the paper. We start from the explicit TRW updates, and show how to make the large messages implicit:

$$m_{i\to j}(s) \propto \sum_{x\in\boldsymbol{\mathcal{X}}} \mathbf{1}[\phi_j(x)=s]\nu_i(x)\frac{\prod_{j':j'\neq j}\left(M_{j'\to i}(x)\right)^{\rho_{j'\to i}}}{\left(M_{j\to i}(x)\right)^{1-\rho_{i\to j}}},$$

where $\rho_{i\to j}$ are marginals of a spanning tree distribution over $K_{I,J}$.

Again, the idea is to find a parameter vector $\boldsymbol{\xi}_{i,j}\in\mathbb{R}^J$ such that

$$\frac{\prod_{j':j'\neq j}\left(M_{j'\to i}(x)\right)^{\rho_{j'\to i}}}{\left(M_{j\to i}(x)\right)^{1-\rho_{i\to j}}} \propto \exp\langle\boldsymbol{\phi}(x),\boldsymbol{\xi}_{i,j}\rangle. \tag{8}$$

To do this, we start by rewriting the numerator of the left hand side of Equation (8):

$$\begin{aligned}
\prod_{j':j'\neq j}\left(M_{j'\to i}(x)\right)^{\rho_{j'\to i}} &= \prod_{j':j'\neq j}\left(\frac{e^{\theta_{j'}\phi_{j'}(x)}\prod_{i':i\neq i}\left(m_{i'\to j'}(\phi_{j'}(x))\right)^{\rho_{i'\to j'}}}{\left(m_{i\to j'}(\phi_{j'}(x))\right)^{\rho_{i\to j'}}}\right)^{\rho_{j'\to i}} \\
&= \exp\left\{\sum_{j':j'\neq j}\rho_{j'\to i}\Big(\theta_{j'}\phi_{j'}(x)+\sum_{i':i'\neq i}\rho_{i'\to j'}\log m_{i'\to j'}(\phi_{j'}(x)) \right.\right. \\
&\qquad \left.\left. -\rho_{i\to j'}\log m_{i\to j'}(\phi_{j'}(x))\Big)\right\} \\
&\propto \exp\left\{\sum_{j':j'\neq j}\rho_{j'\to i}\phi_{j'}(x)\Big(\theta_{j'}+\sum_{i':i'\neq i}\rho_{i'\to j'}\zeta_{i',j'}-\rho_{i\to j'}\zeta_{i,j'}\Big)\right\},
\end{aligned}$$

where we have used in the last step the assumption that ϕ_j has domain $\{0,1\}$, which implies that $m_{i\to j}(\phi_j(x)) = \exp\{\phi_j(x)\log m_{i\to j}(1)+(1-\phi_j(x))\log m_{i\to j}(0)\}\propto\exp\{\phi_j(x)\zeta_{i,j}\}$.

A similar argument on the denominator of the left hand side of Equation (8) yields:

$$\left(M_{j\to i}(x)\right)^{1-\rho_{i\to j}} \propto \exp\left\{(1-\rho_{i\to j}\phi_j(x)\Big(\theta_j+\sum_{i':i'\neq i}\rho_{i'\to j}\zeta_{i',j}-\rho_{i\to j}\zeta_{i,j}\Big)\right\}.$$

Combining these gives the update:

$$\left(\boldsymbol{\xi}_{i,j}\right)_{j'} = \left(\theta_{j'}+\sum_{i':i'\neq i}\rho_{i'\to j'}\zeta_{i',j'}-\rho_{i\to j'}\zeta_{i,j'}\right)\cdot\begin{cases}\rho_{j'\to i} & \text{if } j'\neq j\\ (1-\rho_{i\to j}) & \text{otherwise.}\end{cases}$$

Finally, applying the argument introduced in Section 2.4 yields the reparameterized updates shown in Figure 4.

A.3 Matching factorizations

In this section, we show how to compute efficiently the super-partition functions described in the matching examples of Section 2.1 and 3.1.

Proposition 4 *For perfect bipartite matchings in SBM, computing one super-partition function $A_i(\boldsymbol{\theta})$ takes time $O(N^2)$.*

Proof: For this type of super-partition function, we claim that computation simply involves renormalizing rows or columns of a matrix. We first introduce some notation: let the sufficient statistic coordinate j corresponds to the indicator $x_{m,n}$, and $\boldsymbol{X}_i$ denote a random variable distributed according to the member indexed by $\boldsymbol{\theta}$ in the exponential family with base measure ν_i and sufficient

statistics $\boldsymbol{\phi}$. Note that in the case of SBM, this corresponds to a distribution over functions of the form $f : \{1, 2, \ldots, N\} \to \{1, 2, \ldots, N\}$.

With this notation, we can write:

$$\nabla_j A_1(\boldsymbol{\theta}) = \mathbb{E}[\phi_j(\boldsymbol{X}_1)] \tag{9}$$
$$= \mathbb{P}(\boldsymbol{X}_1(m, n) = 1) \tag{10}$$
$$= \frac{\exp \theta_{m,n}}{\sum_{n'=1}^{N} \exp \theta_{m,n'}}, \tag{11}$$

and similarly:

$$\nabla_j A_2(\boldsymbol{\theta}) = \frac{\exp \theta_{m,n}}{\sum_{m'=1}^{N} \exp \theta_{m',n}}. \tag{12}$$

Therefore by caching the normalizations, it is possible to compute all the gradient in time $O(N^2)$. ■

Proposition 5 *For perfect bipartite matchings in HBM, computing one super-partition functions $A_i(\boldsymbol{\theta})$ takes time $O(N^3)$.*

Proof: As described in the paper, at a high level, the technique we use to compute $\nabla_j A_i(\boldsymbol{\theta})$ involves constructing an *auxiliary exponential family* with associated graphical model given by a chain of length N, and where the state space of each node in this graph is $\{1, 2, \ldots, N\}$. The basic sufficient statistic coordinates are encoded as node potentials, and the augmented ones, as edge potentials in the chain.

To make this precise, let us introduce some notation. Let $\boldsymbol{\tau}$, $B(\boldsymbol{\tau})$ and $\boldsymbol{\varphi}(y)$ denote the parameters, log-partition function and sufficient statistics of the auxiliary exponential family, $y = (y_1, \ldots, y_N), y_n \in \{1, \ldots, N\}$. We construct the sufficient statistic vectors such that that they have the same dimensionality as $\boldsymbol{\phi}$. The coordinates of $\boldsymbol{\varphi}$ correspond naturally to those of $\boldsymbol{\phi}$: if $\phi_j(x) = x_{n,m}$, then $\boldsymbol{\varphi}_j(y) = \mathbf{1}[y_n = m]$, and if $\phi_j(x) = x_{n,m}x_{n+1,m+1}$, then $\boldsymbol{\varphi}_j(y) = \mathbf{1}[y_n = m]\mathbf{1}[y_{n+1} = m + 1]$. With this construction and by setting $\boldsymbol{\tau} = \boldsymbol{\theta}$, we have $\nabla_j A_i(\theta) = \nabla_j B(\boldsymbol{\tau})$. This computation can be done with forward-backward on chain of length N and state space of size N, hence a total running time of $O(N^3)$. ■

A.4 Multiple sequence alignment factorization

We start by defining formally the state space, sufficient statistic and the measure factors involved. The state space is the collection of all pairwise alignment indicators, and we use the notation $x_{m,n}^{k,k'}$ to denote the indicator function on the alignment between character m of sequence k and character n of sequence k'. In this section, we will assume for simplicity that the sufficient statistic coordinates have the form $\phi_j(x) = x_{m,n}^{k,k'}$, but higher order statistics were added for the experiments of Section 4.2. Handling those is no more complicated than what was demonstrated for matchings in Section 3.1.

There are two types of factors in the measure decomposition:

Monotonicity: each pair of components k, k' forms a pairwise alignment:

$$\nu_i(x) = \prod_{m=1}^{N_k} \prod_{m'=1}^{N_{k'}} \mathbf{1}\left[x_{m,n}^{k,k'} = 1, x_{m',n'}^{k,k'} = 1 \Longrightarrow (m > m', n > n') \text{ or } (m < m', n < n')\right].$$

Transitivity: for each triplet of sequences k, k', k'' and positions m, n, p, transitivity holds:

$$\nu_i(x) = \mathbf{1}\left[x_{m,n}^{k,k'} = 1, x_{n,p}^{k',k''} = 1 \Longrightarrow x_{m,p}^{k,k''} = 1\right].$$

Proposition 6 *Each monotonicity factor can be computed in time $O(N_k N_{k'})$, where $N_k, N_{k'}$ are the lengths of the sequences involved.*

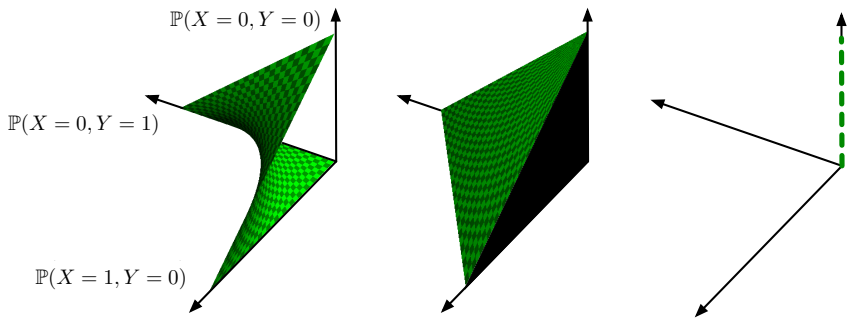

Figure 5: Left: the mean field realizable moments surface $\mathscr{M}_{\text{MF}}$, center, the realizable moments volume $\mathscr{M}$ without a structured base measure, right, the realizable moment line $\mathscr{M}$ with a structured base measure (in green, parallel to the z-axis).

Proof: The idea is to use a non-homogeneous pair HMM, or weighted transducer [27]. In the pair HMM terminology, weights of matching two symbols in this transducer are given by $\exp \theta^{k,k'}_{m,n}$, while the weights of deletions and insertions are set to one.[2]

Once the weighted transducer is constructed, there are standard polynomial-time algorithms for finding its partition function and natural parameter gradient [27]. ■

Proposition 7 *Each transitivity factor can be computed in constant time, and the super-partition functions take the form:*

$$A_i(\boldsymbol{\theta}) = 1 + \exp\left(\theta^{k,k'}_{m,n} + \theta^{k',k''}_{n,p} + \theta^{k,k''}_{m,p}\right) + \exp\left(\theta^{k,k'}_{m,n}\right) + \exp\left(\theta^{k',k''}_{n,p}\right) + \exp\left(\theta^{k,k''}_{m,p}\right)$$

Proof: The eight possible cases to consider are shown in Figure 6, and the ones in the support of the factor are boxed. They each correspond to a term in the sum above by inspection. ■

A.5 Linearization of partial orders factorization

Proposition 8 *The partition function and gradient of the factors proposed in Section 3.3 can be computed in time $O(N^3)$.*

Proof:

Let $G_i = (V, E_i)$ be the current forest in the factorization. We now introduce a new auxiliary family: let $\boldsymbol{\tau}$, $B(\boldsymbol{\tau})$ and $\boldsymbol{\varphi}(y)$ denote its parameters, log-partition function and sufficient statistics of the auxiliary exponential family, $y = (y_1, \ldots, y_N), y_n \in \{1, \ldots, N\}$. We construct the sufficient statistic vectors in the same way as in the proof of Proposition 5. The base measure μ of the auxiliary family enforces the portions of $\leq_p$ that are in E_i. Formally, it is defined as:

$$\mu(y) = \prod_{(n \to n') \in E_i} \mathbf{1}[y_n < y_{n'}].$$

This computation can be done with the sum product algorithm on a forest of size N and state space of size N, hence a total running time of $O(N^3)$. ■

A.6 MFMF does not guarantee a log partition lower bound

In contrast to what one would expect with a mean field algorithm, MFMF is not guaranteed to lower bound the log partition function. In this section, we show why the argument used in [5] to prove the

[2] Special costs for deletion and insertion are encoded in the matching costs as a ratio, and long gap/hydrophobic core modeling are encoded by augmenting the state of the transducers.

bound in the case of standard mean field does not apply to MFMF, and then show a simple counter example. As one would expect, the difference comes from the structured base measure.

We first review the argument of [5], Section 5.4, specializing it to our situation, where the graphical model it described in Section 2.2, and the tractable subgraph is the fully disconnected graphical model on $S_1, S_2, \ldots, S_J, B_1, B_2, \ldots, B_I$ (the *naive mean field*). We let $\mathscr{M} = \nabla A(\mathbb{R}^J \times \boldsymbol{\mathcal{X}} \times \cdots \times \boldsymbol{\mathcal{X}})$ denote the set of *realizable moments*. We will also use the following definition:

Definition 9 *For an extended real-valued function f, the Legendre-Fenchel transformation is defined as:*

$$f^*(x) = \sup\{\langle x, y\rangle - f(y) : y \in \mathrm{dom}(f)\}.$$

When f is convex and lower semi-continuous, $f = f^{**}$, we can use convexity of A to obtain:

$$A(\boldsymbol{\theta}) = \sup\{\langle \boldsymbol{\theta}, \boldsymbol{\mu}\rangle - A^*(\boldsymbol{\mu}) : \boldsymbol{\mu} \in \mathscr{M}\}. \tag{13}$$

Formulation (13) is no more tractable than the definition of A, but gives a constrained optimization problem that can be relaxed. Mean field methods can be seen as a particular type of relaxation where the sup is taken over the set of realizable moment induced by a simpler exponential family. In the case of naive mean field on our graphical model, the simpler family is defined as

$$\text{NMF} = \left\{ p_{\boldsymbol{\gamma}}(s_1, \ldots, s_J, b_1, \ldots, b_I) = \exp\left(\sum_i \sum_{x \in \boldsymbol{\mathcal{X}}} \mathbf{1}[b_i = x]\gamma_{i,x} + \sum_j s_j \gamma_j \right) : \gamma_{i,x}, \gamma_j \in \mathbb{R} \right\},$$

from which we define

$$\mathscr{M}_{\text{MF}} = \left\{ \boldsymbol{\mu} \in \mathbb{R}^J : \exists p \in \text{NMF with } \boldsymbol{\mu} = \mathbb{E}[\boldsymbol{\phi}(\boldsymbol{X})], \boldsymbol{X} \sim p \right\}.$$

With this notation, the mean field objective function is:

$$A_{\text{MF}}(\boldsymbol{\theta}) = \sup\{\langle \boldsymbol{\theta}, \boldsymbol{\mu}\rangle - A^*(\boldsymbol{\mu}) : \boldsymbol{\mu} \in \mathscr{M}_{\text{MF}}\}.$$

Without a structured base measure (meaning, when the base measure is the uniform counting measure over the full state space), we have $\mathscr{M}_{\text{MF}} \subseteq \mathscr{M}$, and it follows that the mean field estimate is a lower bound. On the other hand, since the edge potentials are deterministic, this inclusion does not hold in our case.

To see why, we show a simple counter-example in Figure 5: a graphical model on a pair of binary random variables, X, Y. One can check easily that if an indicator edge potential $\mathbf{1}[X = Y]$ is added, then $\mathscr{M}$ is neither included nor enclosing $\mathscr{M}_{\text{MF}}$.

B More examples of factorizations

B.1 Plane partitions

Counting *plane partitions* is a classical problem in statistical physics, combinatorics and probability theory [3]. A *plane partition* is an array of non-negative integers, $p_{n,m}, 0 \leq n, m \leq N$ such that $p_{n+1,m} \geq p_{n,m}$, $p_{n,m+1} \geq p_{n,m}$. There is a well-known connection between these arrays and a certain type of *routing*, exemplified in Figure 6. We will describe the factorization in the routing formulation, which represents plane partitions as a collection of N non-crossing integer paths, each of length $2N + 1$. Path n starts and ends at position n, and its transitions are either the identity, an increase by one (only allowed in the first N transitions), or a decrease by one (in the last N transitions).

We propose an approximation based on $N - 1$ factors. Each factor relaxes the problem to enforcing non-crossing only for two consecutive paths. With this relaxation, the partition function can be computed in $O(N^3)$ by using forward-backward on a chain of length $2N + 1$ with state space $O(N^2)$ that keeps track of the position of the two consecutive paths.

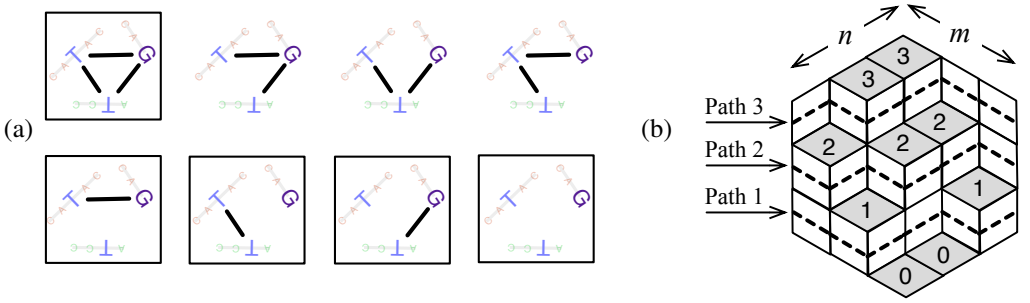

Figure 6: (a) Pictorial representation of the terms involved in the transitivity super-partition computation. The boxed alignment triplets correspond to the transitive cases. (b) The 3D representation of a plane partition with $N = 3$. The value $p_{n,m}$ is the height of the pile at (n, m), for example $p_{3,3} = 3$. The equivalent *routing* representation is shown as a dashed line. For example, path 1 is $(1, 1, 1, 2, 2, 1, 1)$.

B.2 Traveling salesman problem

The method is not limited to #P problems derived from decision problems in P: we show in this section for example that the counting version of the traveling salesman problem, which is NP in its decision version [28], can be attacked with the same tools.

We consider a set of N cities $\{c_1, \ldots, c_N\}$, where each pair of cities has an associated parameter $\theta(c_n, c_m) \in \mathbb{R}$. A *tour* t is a list of cities, $t = t_1, t_2, \ldots, t_N : t_n \in \{c_1, \ldots, c_N\}$ where each city is visited exactly once, i.e. $\{t_1, \ldots, t_N\} = \{c_1, \ldots, c_N\}$. The *weight* of a tour is the product of the weights of the pairs of consecutive cities in the tour (modulo N): $w(t) = \exp\{\sum_{n=1}^{N-1} \theta(t_n, t_{n+1}) + \theta(t_1, t_N)\}$. By normalization, this yields a probability model:

$$\mathbb{P}(T = t) = \exp\left\{\sum_{n=1}^{N-1} \theta(t_n, t_{n+1}) + \theta(t_1, t_N) - A(\boldsymbol{\theta})\right\}$$
$$A(\boldsymbol{\theta}) = \log \sum_{\text{tour } t} w(t),$$

and also an exponential family indexed by $\boldsymbol{\theta}$.

Fix without loss of generality an arbitrary city c_1 as the starting and ending point, and take $\boldsymbol{\mathcal{X}}$ to be the set of all paths of length N that starts and ends at c_1, but without the coverage restriction. One factorization for this problem can be constructed by looping over the $N - 1$ other cities, $c_n \neq c_1$, and building for each one a factor that enforces that c_n be visited exactly once. Computation over a single factor can be computed using dynamic programming (by maintaining the number of steps left and whether c_n was visited or not). Moreover, a state that satisfies all factors is a valid tour.

C More information on the experiments

C.1 Handling extended real parameters

Note that in order to handle the cases where a canonical parameter coordinate is $+\infty$, we need to slightly redefine the super-partition functions as follows:

$$A_i(\theta) = \sum_{x \in \mathcal{C}} \exp\left\{\sum_{j=1}^{J} \mathbf{1}[\theta_j < +\infty]\theta_j \phi_j(x)\right\} \nu_i(x) \prod_{j=1}^{J} \mathbf{1}[\theta_j = +\infty \Rightarrow \phi_j(x) = 1].$$

We also use the convention $(\pm\infty) - (\pm\infty) = \pm\infty$.

C.2 Matching experiments

The generative model used in the third experiment works as follows: first, generate a bipartite perfect matching according to the exponential family HBM $M \sim \text{HBM}(\boldsymbol{\theta})$, next, generate a noisy observation for each edge, $Y_{m,n}|M \sim N(\mathbf{1}(e_{m,n} \in M), \sigma^2)$. The observations $Y_{m,n}$ and parameters $\boldsymbol{\theta}, \sigma^2$ are given to the algorithm, but not the value of M, which is reconstructed using the minimum Bayes risk estimator $\min_m \mathbb{E}[l(m, M)|Y]$ over the 0-1 loss l. The coefficients of this objective are approximated using BPMF. One can check that forming the objective function involves computing moments over HBM with parameters θ_j for the higher order sufficient statistic coordinates j, and with parameters $\theta_j + 1/\sigma^2$ for the basic sufficient statistic coordinates j. We then optimized the objective using the Hungarian algorithm [29]. The zero-one loss is computed against the true (generating) matching, and averaged over 100 random noisy generated datasets.

C.3 Multiple sequence alignment

We used the following experimental protocol: first, we trained parameters for HMMs using EM ran on all pairs of sequences in the test1/ref1 directory, without using the gold alignment information. Second, we ran BPMF with the factorization described in Section 3.2, with an annealing exponent of 1/10 on the consistency messages to avoid convergence problems. Third, we decoded (i.e. transformed the marginals into a single multiple sequence alignment) using the minimum Bayes risk approximation of [30]. Finally, we computed the standard SP (Sum of Pairs) metric on the annotated core blocks (SP is an edge recall score, see [25] for instance for the details).

References

[1] Alexander Karzanov and Leonid Khachiyan. On the conductance of order Markov chains. *Order*, V8(1):7–15, March 1991.

[2] Mark Jerrum, Alistair Sinclair, and Eric Vigoda. A polynomial-time approximation algorithm for the permanent of a matrix with non-negative entries. In *Proceedings of the Annual ACM Symposium on Theory of Computing*, pages 712–721, 2001.

[3] David Wilson. Mixing times of lozenge tiling and card shuffling Markov chains. *The Annals of Applied Probability*, 14:274–325, 2004.

[4] Adam Siepel and David Haussler. Phylogenetic estimation of context-dependent substitution rates by maximum likelihood. *Mol Biol Evol*, 21(3):468–488, 2004.

[5] Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1:1–305, 2008.

[6] Julie Thompson, Frédéric Plewniak, and Olivier Poch. BAliBASE: A benchmark alignments database for the evaluation of multiple sequence alignment programs. *Bioinformatics*, 15:87–88, 1999.

[7] David A. Smith and Jason Eisner. Dependency parsing by belief propagation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 145–156, Honolulu, October 2008.

[8] David Burkett, John Blitzer, and Dan Klein. Joint parsing and alignment with weakly synchronized grammars. In *North American Association for Computational Linguistics*, Los Angeles, 2010.

[9] Bert Huang and Tony Jebara. Approximating the permanent with belief propagation. *ArXiv e-prints*, 2009.

[10] Yusuke Watanabe and Michael Chertkov. Belief propagation and loop calculus for the permanent of a non-negative matrix. *J. Phys. A: Math. Theor.*, 2010.

[11] Ben Taskar, Dan Klein, Michael Collins, Daphne Koller, and Christopher Manning. Max-margin parsing. In *EMNLP*, 2004.

[12] Ben Taskar, Simon Lacoste-Julien, and Dan Klein. A discriminative matching approach to word alignment. In *EMNLP 2005*, 2005.

[13] John Duchi, Daniel Tarlow, Gal Elidan, and Daphne Koller. Using combinatorial optimization within max-product belief propagation. In *Advances in Neural Information Processing Systems*, 2007.

[14] Aron Culotta, Andrew McCallum, Bart Selman, and Ashish Sabharwal. Sparse message passing algorithms for weighted maximum satisfiability. In *New England Student Symposium on Artificial Intelligence*, 2007.

[15] Percy Liang, Ben Taskar, and Dan Klein. Alignment by agreement. In *North American Association for Computational Linguistics (NAACL)*, pages 104–111, 2006.

[16] Percy Liang, Dan Klein, and Michael I. Jordan. Agreement-based learning. In *Advances in Neural Information Processing Systems (NIPS)*, 2008.

[17] Leslie G. Valiant. The complexity of computing the permanent. *Theoret. Comput. Sci.*, 1979.

[18] Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. Generalized belief propagation. In *Advances in Neural Information Processing Systems*, pages 689–695, Cambridge, MA, 2001. MIT Press.

[19] Carsten Peterson and James R. Anderson. A mean field theory learning algorithm for neural networks. *Complex Systems*, 1:995–1019, 1987.

[20] Martin J. Wainwright, Tommi S. Jaakkola, and Alan S. Willsky. Tree-reweighted belief propagation algorithms and approximate ML estimation by pseudomoment matching. In *Proceedings of the International Conference on Articial Intelligence and Statistics*, 2003.

[21] Alexandre Bouchard-Côté and Michael I. Jordan. Optimization of structured mean field objectives. In *Proceedings of Uncertainty in Artifical Intelligence*, 2009.

[22] Graham Brightwell and Peter Winkler. Counting linear extensions. *Order*, 1991.

[23] Lars Eilstrup Rasmussen. Approximating the permanent: A simple approach. *Random Structures and Algorithms*, 1992.

[24] Des G. Higgins and Paul M. Sharp. CLUSTAL: a package for performing multiple sequence alignment on a microcomputer. *Gene*, 73:237–244, 1988.

[25] Chuong B. Do, Mahathi S. P. Mahabhashyam, Michael Brudno, and Serafim Batzoglou. PROBCONS: Probabilistic consistency-based multiple sequence alignment. *Genome Research*, 15:330–340, 2005.

[26] David B. Searls and Kevin P. Murphy. Automata-theoretic models of mutation and alignment. In *Proc Int Conf Intell Syst Mol Biol.*, 1995.

[27] Mehryar Mohri. *Handbook of Weighted Automata*, chapter 6, pages 213–254. Monographs in Theoretical Computer Science. Springer, 2009.

[28] Richard M. Karp. Reducibility among combinatorial problems. *Complexity of Computer Computations*, page 85103, 1972.

[29] Harold W. Kuhn. The Hungarian Method for the assignment problem. *Naval Research Logistics Quarterly*, 1955.

[30] Ariel Schwartz and Lior Pachter. Multiple alignment by sequence annealing. *Bioinformatics*, 23:e24–e29, 2006.