Marginal Inference in MRFs using Frank-Wolfe

David Belanger, Dan Sheldon, Andrew McCallum
School of Computer Science
University of Massachuseets, Amherst
{belanger,sheldon,mccallum}@cs.umass.edu

Abstract

We introduce an algorithm, based on the Frank-Wolfe technique (conditional gradient), for performing marginal inference in undirected graphical models by repeatedly performing MAP inference. It minimizes standard Bethe-style convex variational objectives for inference, leverages known MAP algorithms as black boxes, and offers a principled means to construct sparse approximate marginals for high-arity graphs. We also offer intuition and empirical evidence for a relationship between the entropy of the true marginal distribution of the model and the convergence rate of the algorithm. We advocate for further applications of Frank-Wolfe to marginal inference in Gibbs distributions with combinatorial energy functions.

1 Introduction

Recently, two different algorithms have been proposed that reduce marginal inference in a Markov Random Field (MRF) to multiple instances of MAP inference [1, 2]. This is desirable because MAP is better understood and often easier than the counting-style problem of marginal inference. In this paper, we propose a third reduction based on the Frank-Wolfe algorithm [3].

Our work has several potential advantages over the previous approaches: (1) The work of Hazan and Jaakkola (2012) computes marginals as an average over t independent MAP solutions obtained by sampling, and thus the accuracy of the marginals converges as $O(\frac{1}{\sqrt{t}})$, while Frank-Wolfe has the potential to achieve a convergence rate of $O(\frac{1}{t})$ [1]. (2) Unlike the work of Ermon et al. (2013), in which parity constraints are added to the model, our work retains the original constraint structure and thus allows application of known black-box MAP solvers [2]. (3) In our reduction, the approximate marginals retain a sparsity structure that may lead to significant memory savings for high-arity graphs. (See Section 3.2).

However, we also identify several weaknesses of our Frank-Wolfe reduction: (1) The apparent convergence rate of $O(\frac{1}{t})$ buries a constant factor that diverges when the marginals approach the boundary of the set of feasible marginals, where the marginals have low entropy. This is a numerical concern if the algorithm's iterates approach the boundary, which is necessarily the case if the true marginals are low-entropy . (2) Since our algorithm modifies the model parameters in each step, it does not always retain structure in the MRF potentials, such as log-submodularity, that make MAP inference tractable, (3) Many MAP solvers maximize over the *local polytope* rather than the *marginal polytope*, and using such a solver for Frank-Wolfe is a reasonable way to solve a relaxation of the original marginal inference problem. However, existing message passing algorithms solve this relaxation efficiently. Regarding this third drawback, our presentation and experiments focus on graphical models, but the techniques generalize trivially to marginal inference in Gibbs distributions given by alternative combinatorial energy functions. We encourage future work on cases such as matchings, where MAP is tractable but we do not have available marginal inference algorithms.

We first provide background on Frank-Wolfe and inference in MRFs. We then present our algorithm and two desirable characteristics: a maneuver based on sparsity allows us to perform exact line search efficiently at each iteration, and we can store sparse tables of clique marginals. We then present experiments supporting the relationship between the underlying entropy of the marginals and the convergence behavior of the algorithm.

2 Background

2.1 Frank-Wolfe Algorithm

Following [3], we minimize convex function $f(\mathbf{x})$ over convex set $\mathbf{X}$ with the following update rule:

$$\mathbf{y}_t = \arg\min_{\mathbf{x}\in\mathbf{X}} \langle \mathbf{x}, -\nabla f(\mathbf{x}^{t-1})\rangle \tag{1}$$

$$\mathbf{x}_t = (1-\gamma_t)\mathbf{x}^t + \gamma_t \mathbf{y}_t, \tag{2}$$

where γ_t is either selected using line search or fixed at $\frac{2}{2+t}$.

2.2 Inference In Markov Random Fields

A joint distribution over $\mathbf{x} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ is defined via a graph $\mathcal{G}$ on $\mathbf{x}$ and an *energy function* $\Phi_{\boldsymbol{\theta}}(\mathbf{x}) = \sum_{c\in\mathcal{C}} \boldsymbol{\theta}_c(\mathbf{x}_c)$, where $\mathcal{C}$ denotes the set of all cliques of $\mathcal{G}$ (including single-node cliques) and $\mathbf{x}_c$ denotes the subvector of $\mathbf{x}$ for a clique c. The joint distribution is given by $P(\mathbf{x}) = \exp\left(\Phi_{\boldsymbol{\theta}}(\mathbf{x})\right)/\log(Z)$. For discrete $\mathbf{x}$, $\Phi_{\boldsymbol{\theta}}$ can always be expressed as a linear function $\langle \boldsymbol{\theta}, \boldsymbol{\mu}\rangle$ of an indicator vector $\boldsymbol{\mu}$ for settings of the cliques.

We seek to perform *marginal inference*, which returns the marginal distribution $P_c(\mathbf{x}_c)$ for every clique. We concatenate these vectors of marginals into one vector, $\boldsymbol{\mu}_{\text{MARG}}$. Following [4], $\boldsymbol{\mu}_{\text{MARG}}$ can be identified as the solution to:

$$\boldsymbol{\mu}_{\text{MARG}} = \arg\max_{\boldsymbol{\mu}\in\mathcal{M}} \langle \boldsymbol{\mu}, \boldsymbol{\theta}\rangle + H_{\mathcal{M}}(\boldsymbol{\mu}), \tag{3}$$

where $\mathcal{M}$ denotes the *marginal polytope*, the set of all marginal distributions realizable from some joint distribution over $\mathbf{x}$ encoded by some $\boldsymbol{\theta}$, and $H_{\mathcal{M}}$ is the positive entropy. A standard approximation is to replace the entropy with some Bethe-style approximation $H_B(\boldsymbol{\mu})$ that factorizes conveniently over the components of $\boldsymbol{\mu}$: $H_B(\boldsymbol{\mu}) = \sum_{c\in\mathcal{C}} W_c H(\boldsymbol{\mu}_c)$, where W_c are counting numbers, designed to maintain the concavity of H_B, but yield a good approximation to $H_{\mathcal{M}}(\boldsymbol{\mu})$ [5]. Here, $H(\boldsymbol{\mu}_c)$ is the standard entropy on the unit simplex: $-\sum_i \boldsymbol{\mu}_i \log(\boldsymbol{\mu}_i)$. $F(\boldsymbol{\mu}) = -\boldsymbol{\mu}\cdot\boldsymbol{\theta} - H_B(\boldsymbol{\mu})$ is called the negative *variational free energy*. Furthermore, a common relaxation of the problem is to replace $\mathcal{M}$ with $\mathcal{L}$, the set of *locally consistent* marginals (i.e. where two distinct clique marginals always agree on their overlap, and all clique marginals are properly normalized).

An alternative problem is MAP inference, the task of finding the assignment to $\mathbf{x}$ with highest probability, i.e. that maximizes $\Phi_{\boldsymbol{\theta}}(\mathbf{x})$. This is equivalent to finding the maximum-energy marginals that assign unit mass to a single possible value for each clique, which the vertices of $\mathcal{M}$ satisfy.

Since any discrete energy function can be formulated as a linear function, MAP inference can conveniently be written as

$$\boldsymbol{\mu}_{MAP} = \arg\max_{\boldsymbol{\mu}\in\mathcal{M}} \langle \boldsymbol{\mu}, \boldsymbol{\theta}\rangle. \tag{4}$$

Many standard MAP algorithms, often based on message passing, relax this constraint to $\boldsymbol{\mu}\in\mathcal{L}$. Therefore, any linear optimization problem over the local polytope can be solved as a MAP problem for some parameter vector $\tilde{\boldsymbol{\theta}}$, provided we use one of many available black-box MAP algorithms that optimize over the local polytope.

3 Minimizing the Variational Free Energy Using Frank-Wolfe

Minimizing the variational free energy using Frank-Wolfe requires solving a minimization problem at every iteration given by: $\boldsymbol{\mu}_t = \arg\min_{\boldsymbol{\mu}\in\mathcal{M}} \langle \boldsymbol{\mu}, -\nabla F(\boldsymbol{\mu}^{t-1})\rangle$, which can be expressed as MAP inference with parameter vector $\tilde{\boldsymbol{\theta}}_t = -\nabla F(\boldsymbol{\mu}_t) = \boldsymbol{\theta} + \nabla H_B(\boldsymbol{\mu})$. Let $\tilde{\boldsymbol{\theta}}_{c,t}$ denote the subvector of $\tilde{\boldsymbol{\theta}}_t$ for clique c at iteration t. We have $\tilde{\boldsymbol{\theta}}_{c,t} = \boldsymbol{\theta}_c + W_c\left(1 + \log(\boldsymbol{\mu}_{c,t-1})\right)$, where $\log(\boldsymbol{\mu}_{c,t-1})$ is taken coordinate-wise.

Algorithm 1 Frank-Wolfe for Marginal Inference

input $\boldsymbol{\theta}$, a vector of MRF parameters
set $\boldsymbol{\mu}_0$ to some interior point of $\mathcal{M}$ // We exp-normalize local potentials.
while !CONVERGED($\boldsymbol{\mu}^t$,$\boldsymbol{\mu}^{t-1}$) **do**
  $\tilde{\boldsymbol{\theta}}_{c,t} = \boldsymbol{\theta}_c + W_c \left(1 + \log(\boldsymbol{\mu}_{c,t-1})\right), \ \forall c, t$
  $\tilde{\boldsymbol{\mu}}_t = \text{MAP-ORACLE}(\tilde{\boldsymbol{\theta}})$
  $\gamma^* = \arg\min_{\gamma \in [0,1]} F\left((1-\gamma)\boldsymbol{\mu}_{t-1} + \gamma\tilde{\boldsymbol{\mu}}_t\right)$ // We use Newton's method.
  $\boldsymbol{\mu}_t = (1-\gamma)\boldsymbol{\mu}_{t-1} + \gamma^*\tilde{\boldsymbol{\mu}}_t$
end while
return $\boldsymbol{\mu}_t$

3.1 Efficient Line Search

We found that line search was very important for improving the convergence speed of the algorithm and preventing oscillatory behavior in early iterations. Let $\tilde{\boldsymbol{\mu}}$ be a 0-1 vector of pseudomarginals returned by the MAP oracle at iteration t. Line search chooses a γ that minimizes the one dimensional function $G(\gamma)$ given by

$$F\left((1-\gamma)\boldsymbol{\mu}^t + \gamma\tilde{\boldsymbol{\mu}}\right) = -\langle\boldsymbol{\theta} \cdot ((1-\gamma)\boldsymbol{\mu}^t + \gamma\tilde{\boldsymbol{\mu}}\rangle + \sum_n W_n H((1-\gamma)\boldsymbol{\mu}_n^t + \gamma\tilde{\boldsymbol{\mu}}_n) + \sum_{e \in E} W_e H((1-\gamma)\boldsymbol{\mu}_e^t + \gamma\tilde{\boldsymbol{\mu}}_e).$$

Evaluating this function requires looping over every entry of every table of values for every node and edge potential in the MRF. However, we can pre-compute certain quantities for a given $\boldsymbol{\mu}_t$ such that evaluating $G(\gamma)$ has computational cost that scales merely with the number of nodes and edges, not the number of entries in the potentials (which have size $O(k^t)$, where k is the number of possible values that each node can take on and t is the size of the largest clique in the MRF). We exploit the fact that $\tilde{\boldsymbol{\mu}}$ corresponds to a corner of $\mathcal{M}$. For a given node marginal $\tilde{\boldsymbol{\mu}}_n$ of the MAP assignment to variable n, let $\tilde{i}_n$ equal the single index that is nonzero. Consider just the node entropy term $\sum_n W_n H((1-\gamma)\boldsymbol{\mu}_n + \gamma\tilde{\boldsymbol{\mu}})$. This is equal to:

$$\begin{aligned}
&\sum_n W_n \left[(1-\gamma)\left(\sum_{i \neq \tilde{i}_n} \boldsymbol{\mu}_n(i)\log((1-\gamma)\boldsymbol{\mu}_n(i))\right) + ((1-\gamma)\boldsymbol{\mu}_n(\tilde{i}_n) + \gamma)\log((1-\gamma)\boldsymbol{\mu}_n(\tilde{i}_n) + \gamma)\right] \\
&= A(1-\gamma)\log(1-\gamma) + B(1-\gamma) + \sum_n W_n((1-\gamma)\boldsymbol{\mu}_n(\tilde{i}_n)) + \gamma)\log((1-\gamma\boldsymbol{\mu}_n(\tilde{i}_n)) + \gamma)
\end{aligned}$$

Here, A and B are constants independent of γ. The edge-wise entropy can be decomposed similarly. This is a smooth function of γ and can be minimized in a few iterations using Newton's method.

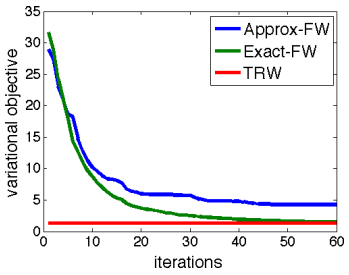
3.2 Sparse Storage of Marginals

Every iterate $\boldsymbol{\mu}_t$ is a convex combination of at most t distinct vertices of the polytope $\mathcal{M}$ and the initial iterate $\boldsymbol{\mu}_0$. Therefore, every clique marginal $\boldsymbol{\mu}_{c,t}$ is a mixture of at most t 0-1 distributions and $\boldsymbol{\mu}_{c,0}$. This means that we can store the clique marginals in terms of sparse vectors, with no more that t nonzero components, and reconstruct them by adding this sparse vector to $\boldsymbol{\mu}_{c,0}$. We save memory if we can store $\boldsymbol{\mu}_{c,0}$ in small space. This is possible, for example, if we choose $\boldsymbol{\mu}_{c,0}$ to be the uniform distribution, or a 'cross-product distribution,' where $\boldsymbol{\mu}_{(a,b),0}(s_a, s_b) = \boldsymbol{\mu}_{a,0}(s_a)\boldsymbol{\mu}_{b,0}(s_b)$.

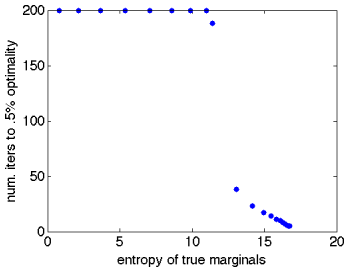
3.3 Convergence Rate and Optimality Guarantee

Marginal inference in general is #P-hard, but we are solving an approximation, due to the convex entropy approximation $H_B(\boldsymbol{\mu})$. The Frank-Wolfe algorithm has been shown to have suboptimality decaying as $F(\boldsymbol{\mu}_t) - F(\boldsymbol{\mu}^*) \leq \frac{2C_F}{t+2}(1+\delta)$, where the additive suboptimality of MAP at iteration t is $\frac{\delta C_f}{t+2}$, and C_F is the *curvature* of F over $\mathcal{M}$ [3].

MAP over the marginal polytope is NP-hard in general [6], as is approximating it within an additive or multiplicative factor. However, there are many MAP solvers that work well in practice. One solution is to simply use one of these and hope for a high-quality solution to the marginal inference problem, as was done by [1]. We could have focused on graph structures (trees) where MAP inference is tractable using dynamic programming, but dynamic programming can also be used for marginal inference in these. For certain loopy graph structures, there are conditions on $\boldsymbol{\theta}$



(a) Convex variational objective vs. iteration # for Algorithm 1 using exact MAP (Junction Tree) and approximate MAP. Horizontal line is the true optimum.



(b) # iterations to 0.5% relative duality gap vs. entropy of true marginal distribution. Problem structure and potentials are fixed, entropy is varied by changing temperature. Num iterations is capped at 200.

such that MAP is also tractable, such as when the potentials are submodular [7]. However, we shift the parameters at every iteration of Algorithm 1, and we can not guarantee the submodularity of $\tilde{\boldsymbol{\theta}}_t = -\nabla F(\boldsymbol{\mu}_t) = -\boldsymbol{\theta} - \nabla H_B(\boldsymbol{\mu}_t)$. If we relax the marginal inference problem to be over the local polytope, then we can perform MAP over the local polytope, which is a polynomial-sized LP. Many message passing algorithms approximately solve this LP very efficiently [8].

Curvature C_F quantifies how much F can differ from its linearization, and is defined formally in [3]. Unfortunately, in our case it is unbounded as one approaches the boundary of the local polytope. In the expression for $\nabla F(\boldsymbol{\mu}_t)$ above, we see that $W_c(1 + \log(\boldsymbol{\mu}_{c,t-1}))$ has arbitrarily large magnitude when the marginal probability of certain clique assignements is small.

Observe, however, that $\log(\boldsymbol{\mu}_{c,t-1})$ becomes unmanageable only for $\boldsymbol{\mu}_{c,t-1}$ with components quite close to 0 (in a Gibbs distribution, no clique assignment ever has zero probability, so $\nabla F(\boldsymbol{\mu}_t)$ is always well-defined). If the iterates $\boldsymbol{\mu}_t$ never get too close to the boundary of $\mathcal{M}$, then the 'effective' curvature term will reasonable. Therefore, the worst-case convergence rate of Algorithm 1 is unbounded, but this may not be a concern in practice. Of course, if the true marginals of the distribution encoded by $\boldsymbol{\theta}$ are close to the boundary of $\mathcal{M}$, then C_F will be large in the neighborhood of the solution, and we should not expect fast convergence. In our experiments, we demonstrate that the algorithm converges faster when the true marginal distribution encoded by $\boldsymbol{\theta}$ has higher entropy.

4 Experiments

We perform inference over the marginal polytope for synthetic grid-structured binary MRFs where each component of $\boldsymbol{\theta}$ is drawn independently from a mean-0 Gaussian. The counting numbers W_c in our convex entropy approximation are given by a randomly-generated tree decomposition, used, for example, by the TRW algorithm [9]. This is useful because we can use convergent TRW message passing to compute a lower bound on the target variational objective. For the sake of convenience, we also perform approximate MAP inference using a MAP version of TRW with the same tree decomposition [10], though we could have used any black-box solver. TRW performs MAP over the local polytope, but we use a standard method for rounding its solution to a vertex of the marginal polytope. We use the TRW and junction tree implementations in the UGM toolkit [11].

In Figure 1a, we demonstrate the correctness and convergence of our algorithm on a 10-by-10 grid. We compare using exact MAP (junction tree) in the inner loop, which is obviously too slow to use in practice, with using approximate MAP. In general, we find that approximate MAP does not change the overall scale of iterations until convergence, but it converges to a suboptimal objective.

In the previous section, we suggested a relationship between the entropy of the underlying marginal distribution and the empirical convergence time of the algorithm. In Figure 1b, we take a fixed 5-by-5 grid and vary the entropy by mapping $\boldsymbol{\theta} \rightarrow \boldsymbol{\theta}/T$ for $0.1 \leq T \leq 4$. MAP is performed using the junction-tree algorithm, in order to avoid complications from approximations in the inner loop. We plot the number of iterations to obtain a solution within 0.5% relative optimality gap, where the suboptimality is with respect to a lower bound on the objective computed using TRW. In this example, and in general, we find a distinct phase transition where the algorithm suddenly starts converging much faster. After this transition, increasing the temperature further continues to increase convergence speed, in a seemingly linear fashion. We leave further exploration of this entropy-convergence relationship to future work.

4.1 Acknowledgement

This work was supported in part by the Center for Intelligent Information Retrieval and in part by DARPA under agreement number FA8750-13-2-0020. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

References

[1] Tamir Hazan and Tommi S Jaakkola. On the Partition Function and Random Maximum A-Posteriori Perturbations. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pages 991–998, 2012.

[2] Stefano Ermon, Carla Gomes, Ashish Sabharwal, and Bart Selman. Taming the Curse of Dimensionality: Discrete Integration by Hashing and Optimization. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 334–342, 2013.

[3] Martin Jaggi. Revisiting Frank-Wolfe: Projection-Free Sparse Convex Optimization. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 427–435, 2013.

[4] Martin J Wainwright and Michael I Jordan. Graphical Models, Exponential Families, and Variational Inference. *Foundations and Trends® in Machine Learning*, 1(1-2):1–305, 2008.

[5] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT press, 2009.

[6] Solomon Eyal Shimony. Finding MAPs for Belief Networks is NP-hard. *Artificial Intelligence*, 68(2):399–410, 1994.

[7] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast Approximate Energy Minimization via Graph Cuts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(11):1222–1239, 2001.

[8] Amir Globerson and Tommi S Jaakkola. Fixing max-product: Convergent message passing algorithms for map lp-relaxations. In *Advances in neural information processing systems*, pages 553–560, 2007.

[9] Martin J Wainwright, Tommi S Jaakkola, and Alan S Willsky. A New Class of Upper Bounds on the Log Partition Function. *Information Theory, IEEE Transactions on*, 51(7):2313–2335, 2005.

[10] Vladimir Kolmogorov. Convergent Tree-Reweighted Message Passing for Energy Minimization. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(10):1568–1583, 2006.

[11] Mark Schmidt. UGM: Matlab code for undirected graphical models.