Frank-Wolfe based Dual Decomposition for MAP Inference

Shell X. Hu
EECS, Oregon State University
huxu@onid.orst.edu

Abstract

This paper presents a new approach to MAP inference of a general graphical model. We cast MAP inference as a linear programming problem, and focus on solving its equivalent dual problem, since the latter can be formulated as a constrained convex optimization, whose objective function decomposes as a sum of convex subproblems. Such a formulation fits well within the Frank-Wolfe optimization framework. We call our new MAP inference as Frank-Wolfe based dual decomposition (FWDD). Specifically, in each iteration, the subgradient descent involves solving subproblems and a linear minimization of the dual problem. By specifying the underlying structure as a spanning subgraph with bounded treewidth, each subproblem can be solved efficiently by junction tree algorithm. The linear minimization problem turns out that can be further decomposed into smaller linear programs, thus can also be solved efficiently. Our experiments on image denoising and large scale scene labeling demonstrate that our FWDD performs on par with standard dual decomposition methods in terms of convergence speed, duality gap and classification accuracy.

Introduction

Many problems with structured inputs and outputs can be formulated as graphical models. In such problems, predicting structured outputs often equal to solve a maximum a posteriori (MAP) inference. The goal of MAP inference is to find the most probable assignment to each factor, which is usually cast as a combinatorial optimization problem.

The objective function of MAP inference (or energy minimization) typically factorizes into a sum of potential functions over subsets of variables. However, MAP inference involving discrete variables is typically NP-hard (Koller and Friedman 2009).The intractability of MAP inference requires approximation algorithms. One common approach is the loopy belief propagation (LBP) (Yedidia, Freeman, and Weiss 2005), which is a generalization of the forward-backward algorithm (Rabiner 1989) and its relation with Bethe free energy has been well studied (Yedidia, Freeman, and Weiss 2005; Wainwright and Jordan 2008). LBP typically provides good approximate solutions in practice, but there is no theoretical guarantee of its convergence to a local optimum. Another approach to MAP inference relaxes the original optimization problem to a linear programming (LP) problem, and approximates the marginal polytope constraints of the original problem by using a local polytope (Sontag et al. 2008). It has been found that the quality of LP relaxation of MAP inference depends mostly on the tightness of marginal polytope approximation (Yanover, Meltzer, and Weiss 2006; Sontag et al. 2008). Although LP is tractable, general LP solvers typically do not exploit the structure of the original problem, thus usually slower than LBP. The connection between message passing algorithms and LP relaxation has been investigated by (Werner 2007; Globerson and Jaakkola 2007; Kolmogorov 2006). These method interpret message passing as coordinate descent over dual variables with convergence guarantee, however they are generally not promised to converge to a global minimum (Meshi, Jaakkola, and Globerson 2012).

The above issues with LP relaxation of MAP inference have been addressed by decomposing the original problem into several tractable subproblems, each of which can be solved efficiently by dual decomposition (Komodakis, Paragios, and Tziritas 2011; Domke 2011; Sontag, Globerson, and Jaakkola 2010; Martins et al. 2014). Dual decomposition is a standard optimization method that decouples the original objective function into independent subproblems by introducing Lagrangian multipliers. The multipliers are then re-adjusted in the master problem to ensure that all solutions of the subproblems agree. For example, (Komodakis, Paragios, and Tziritas 2011) and (Domke 2011) decompose the master problem into a set of spanning trees. (Sontag, Globerson, and Jaakkola 2010) provide an alternative decomposition in terms of factors and show the connection between dual decomposition and the dual formulation of LP relaxation.

We present a new approach for MAP inference. Our work also make use of the dual problem of LP relaxation. However, unlike previous dual decomposition methods (Komodakis, Paragios, and Tziritas 2011; Sontag, Globerson, and Jaakkola 2010), we do not establish explicit connections between dual variables and messages. Instead, the dual variables for each subproblem can be viewed as components of the global potential parameter vector. Accordingly, the original high-treewidth graphical model is decomposed into a set of spanning subgraphs with bounded treewidth. Each resulting subgraph gives rise to a simpler optimization subproblem, which can be solved efficiently by running max-product message passing on the junction tree.

The master problem is formulated as a constrained convex optimization problem and thus fit well within the Frank-Wolfe optimization framework (Franke and Wolfe 1956; Jaggi 2013). Comparing to projection or proximal methods (Komodakis, Paragios, and Tziritas 2011), Frank-Wolfe algorithm requires only linear computation in each iteration, thus provides better convergence speed. Moreover, we show that the linear minimization oracle can be further decomposed into smaller linear programs and thus can be efficiently solved. We also analyze the convergence rate of the proposed algorithm based on the convergence analysis of the general Frank-Wolfe optimization framework (Jaggi 2013). We call the proposed algorithm as Frank-Wolfe based Dual Decomposition (FWDD). We evaluate our FWDD on two challenging structured prediction tasks: image denoising and scene labeling. The results demonstrate that our FWDD performs on par with standard dual decomposition methods in terms of convergence speed, duality gap and yields better classification accuracy on the scene labeling task.

Frank-Wolfe Algorithm

The Frank-Wolfe algorithm was originally proposed by (Franke and Wolfe 1956) and revisited by (Jaggi 2013). It aims at solving a constrained convex optimization in the form of $\min_{\mathcal{D}} f(\mathbf{x})$, where D is a compact convex set, and f is a continuously differentiable function. There are two main steps in the main loop of Frank-Wolfe algorithm:

- Solve a linearization of f at $\mathbf{x}^k$, also called linear minimization oracle (LMO), $\mathbf{s} = \arg\min_{\mathbf{x}\in\mathcal{D}} \langle x, \nabla f(x^k)\rangle$
- Update variable as $\mathbf{x}^{k+1} = (1-\gamma)\mathbf{x}^k + \gamma\mathbf{s}$, where $\mathbf{s} - \mathbf{x}^k$ is the descent direction, and γ is the step size.

Comparing to projection or proximal methods, Frank-Wolfe requires only to run a linear oracle at each iteration instead of a quadratic optimization. Moreover, note that each call of LMO chooses a corner of $\mathcal{D}$, thus the final solution is a sparse convex combination of these corners, which makes Frank-Wolfe a memory efficient algorithm.

For measuring the convergence, a important quantity is the linearized duality gap between f and its linear lower bound $f(\mathbf{x}) + \langle \nabla f(\mathbf{x}), \mathbf{s} - \mathbf{x}\rangle$, which is defined as

$$\mathrm{gap}(\mathbf{x}) = \max_{\mathbf{s}\in\mathcal{D}} \langle \nabla f(\mathbf{x}), \mathbf{x} - \mathbf{s}\rangle. \tag{1}$$

The definition of $\mathrm{gap}(\mathbf{x})$ implies that $\mathrm{gap}(\mathbf{x}) \geq f(\mathbf{x}) - f(\mathbf{x}^*)$, in other words, the linearized duality gap is always an upper bound of the primal objective error.

MAP Inference in Graphical Models

A probabilistic graphical model (PGM) is a probabilistic model for which the conditional independence between random variables is denoted by a directed/undirected graph.

Consider a graph $G = (V, E)$ with discrete random variables $\mathbf{x} = (x_1, \ldots, x_n)$ associated with each node, and the random variables are further grouped in terms of a set of clusters of variables $C(G)$ ($\cup_{c\in C(G)} c = G$). A graphical model is usually represented as an exponential family:

$$p_{\boldsymbol{\theta}}(\mathbf{x}) = \frac{1}{Z(\boldsymbol{\theta})} \exp\Big(\sum_{c\in C(G)} \langle \boldsymbol{\theta}_c, \boldsymbol{\phi}(\mathbf{x}_c)\rangle\Big), \tag{2}$$

where $\boldsymbol{\theta}$ is the vector of canonical parameters and $\boldsymbol{\phi}$ denotes the stack of indicator functions as sufficient statistics. $Z(\boldsymbol{\theta}) = \sum_{\mathbf{x}\in\mathcal{X}^n} \exp(\sum_{c\in C(G)} \langle \boldsymbol{\theta}_c, \boldsymbol{\phi}(\mathbf{x}_c)\rangle)$ is the partition function. In addition, we assume that $\boldsymbol{\phi}$ is greater than zero. This is easy to obtain by adding a big constant to all elements of $\boldsymbol{\phi}$, which will not affect the results of inference.

The maximum a posterior (MAP) inference is posed as finding the most probable assignment $\mathbf{x}^*$ that maximize $p_{\boldsymbol{\theta}}(\mathbf{x})$. To slightly abuse the notation, the potential function for each cluster is defined as $\boldsymbol{\theta}_c(\mathbf{x}_c) = \langle \boldsymbol{\theta}_c, \boldsymbol{\phi}(\mathbf{x}_c)\rangle$. Since the partition function $Z(\boldsymbol{\theta})$ is independent of $\boldsymbol{\theta}$, the MAP inference is equivalently cast as a discrete optimization

$$\max_{\mathbf{x}\in\mathcal{X}^n} \sum_{c\in C(G)} \langle \boldsymbol{\theta}_c, \boldsymbol{\phi}(\mathbf{x}_c)\rangle. \tag{3}$$

In general, the MAP inference is NP-hard (Koller and Friedman 2009). Exact inference such as belief propagation with junction tree representation is possible only when the underlying graph structure has low treewidth.

To address the above issue, a standard approximation is to relax the integer programming to a linear programming (Yanover, Meltzer, and Weiss 2006) with auxiliary variables $\boldsymbol{\mu}$, where each cluster potential function $\boldsymbol{\theta}_c(\mathbf{x}_c)$ is replaced by $\sum_{\mathbf{x}_c} \boldsymbol{\mu}_c(\mathbf{x}_c)\boldsymbol{\theta}_c(\mathbf{x}_c)$. It is interesting to see that $\boldsymbol{\mu}_c$ can be interpreted as the marginal probability of cluster c, that is, $\boldsymbol{\mu}_c(\mathbf{x}_c) = \mathbb{E}_{\boldsymbol{\theta}}[\boldsymbol{\phi}(\mathbf{x}_c)]$. The set of valid $\boldsymbol{\mu}$ is called marginal polytope (Wainwright and Jordan 2008), which is denoted by $\mathcal{M}_G = \{\boldsymbol{\mu} \mid \exists\boldsymbol{\theta} \text{ s.t. } \boldsymbol{\mu} = \mathbb{E}_{\boldsymbol{\theta}}[\boldsymbol{\phi}(\mathbf{x})]\}$. It is usually impossible to explicitly represent the marginal polytope $\mathcal{M}_G$, since there are exponentially large number of vertices $\boldsymbol{\phi}(\mathbf{x})$. A further relaxation is to use an outer bound of $\mathcal{M}_G$ by considering only local consistencies. The outer bound is called local polytope, given by the following definition:

$$\mathcal{L}_G = \Big\{\boldsymbol{\mu} \succeq 0 \mid \sum_{\mathbf{x}_c\setminus\mathbf{x}_a} \boldsymbol{\mu}_c(\mathbf{x}_c) = \boldsymbol{\mu}_a(\mathbf{x}_a) \quad \forall c \in C(G), \forall a \subseteq c, \\ \sum_{\mathbf{x}_c} \boldsymbol{\mu}_c(\mathbf{x}_c) = 1 \quad \forall c \in C(G)\Big\}. \tag{4}$$

Thus, the linear programming (LP) relaxation of MAP inference (called MAP-LPR) can be formulated as

$$\max_{\boldsymbol{\mu}\in\mathcal{L}_G} \langle \boldsymbol{\theta}, \boldsymbol{\mu}\rangle. \tag{5}$$

where $\langle \boldsymbol{\theta}, \boldsymbol{\mu}\rangle = \sum_c \sum_{\mathbf{x}_c} \boldsymbol{\theta}_c(\mathbf{x}_c)\boldsymbol{\mu}_c(\mathbf{x}_c)$. However, Eq.(5) cannot be efficiently solved by standard LP solvers due to the exponential size of $\mathcal{L}_G$ (Yanover, Meltzer, and Weiss 2006). The state-of-the-art algorithms for MAP-LPR can be classified into message passing algorithms (Kolmogorov 2006; Werner 2007; Globerson and Jaakkola 2007) and dual decomposition algorithms (Komodakis, Paragios, and Tziritas 2011; Sontag, Globerson, and Jaakkola 2010; Domke 2011), both of which make use of the dual problem of LP relaxation. The former updates dual variables iteratively as a generalization of loopy belief propagation (LBP). Different message passing strategies can be interpreted as different block coordinate descents. On the other hand, dual decomposition algorithms are more general in the sense that the intractable original graph is decomposed into tractable subgraphs, while subproblems induced by these subgraphs are tied by dual variables. Moreover, subproblems can be solved in parallel to gain additional speed-up.

Dual Decomposition Using Frank-Wolfe

The basic idea of dual decomposition is to decompose the problem into several simpler subproblems and constrain them to agree with each other. However, the decomposition usually results in an upper bound of the original problem. The tightness of the upper bound depends on the quality and the quantity of the subproblems, since the strong duality holds if and only if when all subproblems agree on a maximizing assignment.

Previous methods proposed decomposing as spanning trees (Komodakis, Paragios, and Tziritas 2011; Domke 2011) or individual factors (Sontag, Globerson, and Jaakkola 2010; Meshi et al. 2010). A spanning tree decomposition is good for finding consentaneous assignments among subproblems, but it considers only pairwise consistencies, which could be a loose solution to the original MAP problem. Conversely, an individual factor decomposition may result in an equivalent problem, but its coordinate descent nature usually ends up with a local optima that the agreement is not attained.

In this work, we propose to use more complex subproblems while their corresponding subgraphs have bounded treewidth. Hence, each subproblem can be solved by junction tree belief propagation exactly in polynomial time. We first define a new graph structure for a subproblem– the *cluster preserving spanning subgraph* (CPSG), which can be viewed as a spanning tree augmented with certain clusters.

Definition 1. *(Cluster Preserving Spanning Subgraph) Given a graph $G = (V, E)$ and a subset of clusters $\mathbf{c} \subseteq C(G)$, a CPSG $T(\mathbf{c}) = (V_T, E_T)$ is a spanning subgraph of G induced by $\mathbf{c}$, such that $T(\mathbf{c}) = \mathbf{c} \cup F$, where F is the spanning forest of $G \setminus \mathbf{c}$.*

Apparently, the treewidth of a CPSG $T(\mathbf{c})$ is less or equal than the treewidth of $\mathbf{c}$. Thus, it is convenient to control the subproblem complexity by choosing $\mathbf{c}$ with a small treewidth. Correspondingly, a CPSG decomposition outputs a set of CPSGs $\mathcal{T}(G)$ given the input graph G. An example of graph decomposition in terms of CPSGs is shown in Fig. 1. Note that, in a valid CPSG decomposition, every cluster of the original factorization is covered by at least one CPSG. In the following, we will also use T to denote a CPSG for simplicity.

The dual decomposition formulation of our work is shown in Theorem 1, which is different from standard dual decomposition for MAP inference, such as (Komodakis, Paragios, and Tziritas 2011; Sontag, Globerson, and Jaakkola 2010). In FWDD, dual variables are interpreted as components of canonical parameter vector $\boldsymbol{\theta}$, rather than messages, thus dual variable updates are not message passing. In this sense, FWDD is not a message passing algorithm. Similar formulations have been proposed in (Wainwright, Jaakkola, and Willsky 2005; Domke 2011), but their derived results are used for other purposes.

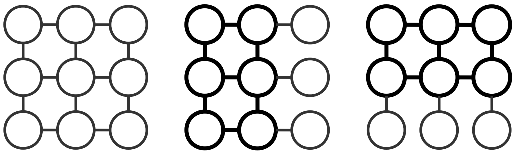

Figure 1: **Left**: a 3×3 grid graph. **Middle and Right**: CPSGs where preserved clusters are highlighted in bold.

Theorem 1. *Given a valid CPSG decomposition $\mathcal{T}(G)$ of the graph G, the MAP-LPR posed in Eq. (5) can be formulated as an equivalent convex optimization*

$$\begin{aligned} \min_{\boldsymbol{\omega}} \quad & \sum_{T \in \mathcal{T}(G)} \max_{\boldsymbol{\nu}_T \in \mathcal{L}_T} \langle \boldsymbol{\omega}_T, \boldsymbol{\nu}_T \rangle \\ \textit{s.t.} \quad & \sum_{T \in \mathcal{T}(G)} \boldsymbol{\omega}_T = \boldsymbol{\theta} \end{aligned} \tag{6}$$

where $\boldsymbol{\omega}_T$ and $\boldsymbol{\nu}_T$ are potentials and beliefs corresponding to CPSG T. They are of the same length as $\boldsymbol{\theta}$ with zeros filled to corresponding elements where edges are not present in G. $\mathcal{L}_T \subseteq \mathcal{L}_G$ is the subset of the local polytope corresponding to the clusters present in T.

Proof. Given the set of CPSGs $\mathcal{T}(G)$, we can construct a potential $\boldsymbol{\theta}_T$ for each CPSG T, such that $\sum_{T:c \in C(T)} \boldsymbol{\theta}_{Tc} = \boldsymbol{\theta}_c$, for all $c \in C(G)$. Then the original optimization in Eq. (5) is equivalent to

$$\max_{\boldsymbol{\mu} \in \mathcal{L}_G} \sum_c \langle \sum_{T:c \in C(T)} \boldsymbol{\theta}_{Tc}, \boldsymbol{\mu}_c \rangle \tag{7}$$

By introducing a copy of $\boldsymbol{\mu}_c$ in each CPSG which has the cluster c, and constrain them to be the same as $\boldsymbol{\mu}_c$, we decompose the original problem into subproblems in CPSGs:

$$\begin{aligned} \max_{\boldsymbol{\mu} \in \mathcal{L}_G} \max_{\boldsymbol{\nu}} \quad & \sum_c \sum_{T:c \in C(T)} \langle \boldsymbol{\theta}_{Tc}, \boldsymbol{\nu}_{Tc} \rangle \\ \text{s.t.} \quad & \boldsymbol{\nu}_{Tc} = \boldsymbol{\mu}_c, \quad \forall T, T : c \in C(T) \\ & \boldsymbol{\nu}_T \in \mathcal{M}_T, \quad \forall T \in \mathcal{T}(G) \end{aligned} \tag{8}$$

where $\boldsymbol{\nu} = (\boldsymbol{\nu}_T^\top)_{T \in \mathcal{T}(G)}^\top$ is the collection of beliefs w.r.t. CPSGs. The above optimization can be rewritten in an equivalent form (Domke 2011) by eliminating $\boldsymbol{\mu}$

$$\begin{aligned} \max_{\boldsymbol{\nu}} \quad & \sum_c \sum_{T:c \in C(T)} \langle \boldsymbol{\theta}_{Tc}, \boldsymbol{\nu}_{Tc} \rangle \\ \text{s.t.} \quad & \boldsymbol{\nu}_{Tc} = \frac{1}{N_c} \sum_{T':c \in C(T')} \boldsymbol{\nu}_{T'c}, \quad \forall T, T : c \in C(T) \\ & \boldsymbol{\nu}_T \in \mathcal{L}_T, \quad \forall T \in \mathcal{T}(G) \end{aligned} \tag{9}$$

where $N_c = |T \colon T : c \in C(T)|$, *i.e.*, the number of CPSGs that contain the cluster c.

The Lagrangian is given by

$$\begin{aligned} L(\boldsymbol{\nu}, \boldsymbol{\lambda}) = & \sum_c \sum_{T:c \in C(T)} \langle \boldsymbol{\theta}_{Tc}, \boldsymbol{\nu}_{Tc} \rangle \\ & + \sum_c \sum_{T:c \in C(T)} \lambda_{Tc} \Big(\boldsymbol{\nu}_{Tc} - \frac{1}{N_c} \sum_{T':c \in C(T')} \boldsymbol{\nu}_{T'c} \Big) \end{aligned} \tag{10}$$

and the dual problem is

$$\begin{aligned} \min_{\boldsymbol{\lambda}} \max_{\boldsymbol{\nu}} \quad & L(\boldsymbol{\nu}, \boldsymbol{\lambda}) \\ \text{s.t.} \quad & \boldsymbol{\nu}_T \in \mathcal{M}_T, \quad \forall T \in \mathcal{T}(G) \end{aligned} \tag{11}$$

Now define $\boldsymbol{\omega}_{Tc} = \boldsymbol{\theta}_{Tc} + \lambda_{Tc} - \frac{1}{N_c} \sum_{T':c \in C(T')} \lambda_{T'c}$. Observing that

$$\begin{aligned} \sum_{T:c \in C(T)} \boldsymbol{\omega}_{Tc} &= \sum_{T:c \in C(T)} \boldsymbol{\theta}_{Tc} + \lambda_{Tc} - \frac{1}{N_c} \sum_{T':c \in C(T')} \lambda_{T'c} \\ &= \sum_{T:c \in C(T)} \boldsymbol{\theta}_{Tc} = \boldsymbol{\theta}_c \end{aligned} \tag{12}$$

Thus, we can transfer Lagrangian multipliers λ_{Tc} into $\boldsymbol{\omega}_{Tc}$, by substituting $\boldsymbol{\theta}_{Tc} = \boldsymbol{\omega}_{Tc} - \lambda_{Tc} + \frac{1}{N_c} \sum_{T':c \in C(T')} \lambda_{T'c}$ into Eq. (10). The dual problem can be rewritten accordingly as

$$\begin{aligned} \min_{\boldsymbol{\omega}} \max_{\boldsymbol{\nu}} \quad & \sum_{c} \sum_{T:c \in C(T)} \langle \boldsymbol{\omega}_{Tc}, \boldsymbol{\nu}_{Tc} \rangle \\ \text{s.t.} \quad & \sum_{T:c \in C(T)} \boldsymbol{\omega}_{Tc} = \boldsymbol{\theta}_c, \quad \forall c \\ & \boldsymbol{\nu}_T \in \mathcal{M}_T, \quad \forall T \in \mathcal{T}(G) \end{aligned} \tag{13}$$

which can be simplified as the same form as Eq. (6), where $\boldsymbol{\omega} = (\boldsymbol{\omega}_T^\top)_{T \in \mathcal{T}(G)}^\top$. By strong duality, the optimum of Eq. (6) is equal to the optimal primal objective. □

Note that Eq. (6) is a constrained convex optimization consisting of subproblems with respect to each CPSG of the form $S_T(\boldsymbol{\omega}_T) = \max_{\boldsymbol{\nu}_T \in \mathcal{L}_T} \langle \boldsymbol{\omega}_T, \boldsymbol{\nu}_T \rangle$, which can be solved exactly by running max-product message passing on the junction tree of T. Accordingly, the objective function of Eq. (6) (master problem) is written as $M(\boldsymbol{\omega}) = \sum_{T \in \mathcal{T}(G)} S_T(\boldsymbol{\omega}_T)$. Note that S_T is a piecewise linear function, which is not differentiable, but it has subdifferential everywhere in the compact convex set. The subgradient of S_T as well as M can be computed by Danskin's min-max theoerm. In next section, we show that the standard convergence results can also be applied to the min-max optimization problem formulated as in Eq. (6).

The following theorem shows that LMO of Eq. (6) can be solved greedily.

Theorem 2. *The linear minimization oracle (LMO) of Eq. (6):*

$$\begin{aligned} \min_{\boldsymbol{\omega} \succeq 0} \quad & \sum_{T \in \mathcal{T}(G)} \langle \boldsymbol{\omega}_T, \frac{\partial M}{\partial \boldsymbol{\omega}_T}(\boldsymbol{\omega}^k) \rangle \\ \textit{s.t.} \quad & \sum_{T \in \mathcal{T}(G)} \boldsymbol{\omega}_T = \boldsymbol{\theta} \end{aligned} \tag{14}$$

can be solved greedily by setting, $\forall c \in C(G), \forall x_c$, $\boldsymbol{\omega}_{T^*c}(\mathbf{x}_c) = \boldsymbol{\theta}_c(\mathbf{x}_c)$ and $\boldsymbol{\omega}_{Tc}(\mathbf{x}_c) = 0$ for any other CPSG T, where $T^* = \arg\min_{T \in \mathcal{T}(G)} \boldsymbol{\nu}^*_{Tc}(\mathbf{x}_c)$ and $\boldsymbol{\nu}^*_T = \arg\min_{\boldsymbol{\nu}_T \in \mathcal{L}_T} \langle \boldsymbol{\omega}^k_T, \boldsymbol{\nu}_T \rangle$.

Proof. Firstly, by Danskin's theorem, $\frac{\partial M}{\partial \boldsymbol{\omega}_T}(\boldsymbol{\omega}^k) = \frac{\partial S_T}{\partial \boldsymbol{\omega}_T}(\boldsymbol{\omega}^k_T) = \boldsymbol{\nu}^*_T$. The LMO can be rewritten as

$$\begin{aligned} & \sum_{c} \sum_{\mathbf{x}_c} \min_{\boldsymbol{\omega}_c(\mathbf{x}_c) \succeq 0} \sum_{T \in \mathcal{T}(G)} \boldsymbol{\omega}_{Tc}(\mathbf{x}_c) \boldsymbol{\nu}^*_{Tc}(\mathbf{x}_c) \\ & \text{s.t.} \sum_{T \in \mathcal{T}(G)} \boldsymbol{\omega}_{Tc}(\mathbf{x}_c) = \boldsymbol{\theta}_c(\mathbf{x}_c), \quad \forall c \in C(G), \forall \mathbf{x}_c \end{aligned} \tag{15}$$

Note that all inner minimization can be solved independently, since each of which involves only one constraint that contains $\boldsymbol{\omega}_c(\mathbf{x}_c)$. Let $\boldsymbol{\alpha}_{Tc}(\mathbf{x}_c) = \frac{\boldsymbol{\omega}_{Tc}(\mathbf{x}_c)}{\boldsymbol{\theta}_c(\mathbf{x}_c)}$. The inner minimization corresponding to a particular c and $\mathbf{x}_c$ is given by

$$\begin{aligned} \min_{\boldsymbol{\alpha}_c(\mathbf{x}_c) \succeq 0} \quad & \sum_{T \in \mathcal{T}(G)} \boldsymbol{\alpha}_{Tc}(\mathbf{x}_c) \boldsymbol{\nu}^*_{Tc}(\mathbf{x}_c) \\ \text{s.t.} \quad & \sum_{T \in \mathcal{T}(G)} \boldsymbol{\alpha}_{Tc}(\mathbf{x}_c) = 1 \end{aligned} \tag{16}$$

Thus, $\boldsymbol{\alpha}_c(\mathbf{x}_c)$ belong to the simplex $\Delta_{|\mathbf{x}_c|}$. Observing that

$$\begin{aligned} \sum_{T} \boldsymbol{\alpha}_{Tc}(\mathbf{x}_c) \boldsymbol{\nu}^*_{Tc}(\mathbf{x}_c) &\geq \sum_{T} \boldsymbol{\alpha}_{Tc}(\mathbf{x}_c) \min_{T} \boldsymbol{\nu}^*_{Tc}(\mathbf{x}_c) \\ &= \min_{T} \boldsymbol{\nu}^*_{Tc}(\mathbf{x}_c). \end{aligned} \tag{17}$$

The equality holds when assigning $\boldsymbol{\alpha}_{T^*c}(\mathbf{x}_c) = 1$, where $T^* = \arg\min_{T \in \mathcal{T}(G)} \boldsymbol{\nu}^*_{Tc}(\mathbf{x}_c)$. Thus, $\boldsymbol{\omega}_{T^*c}(\mathbf{x}_c) = \boldsymbol{\theta}_c(\mathbf{x}_c)$. □

Now, we can solve Eq. (6) using Frank-Wolfe algorithm. In our case, as shown in Eq. (16), the compact convex set $\mathcal{D} = \{\boldsymbol{\omega} \mid \sum_{T \in \mathcal{T}(G)} \boldsymbol{\omega}_T = \boldsymbol{\theta}\}$ can be represented as a product of scaled simplex. The detailed steps is given at Algorithm 1, which is referred to as Frank-Wolfe based dual decomposition (FWDD) for the rest of the paper.

Algorithm 1 Frank-Wolfe based Dual Decomposition

```
Require: $\mathcal{T}(G)$, $\boldsymbol{\theta}$ and $\boldsymbol{\omega}^0$ = initial potentials
  repeat
      Subproblems: for each $T$, solve $S_T(\boldsymbol{\omega}^k_T)$ to obtain $\boldsymbol{\nu}^*_T$
      LMO: $\mathbf{s} \leftarrow \arg\min_{\boldsymbol{\omega} \in \mathcal{D}} \sum_{T \in \mathcal{T}(G)} \langle \boldsymbol{\omega}_T, \boldsymbol{\nu}^*_T \rangle$
      Step size: $\gamma \leftarrow \frac{2}{k+2}$
      Update: $\boldsymbol{\omega}^{k+1} \leftarrow \boldsymbol{\omega}^k + \gamma(\mathbf{s} - \boldsymbol{\omega}^k)$
  until $|\text{gap}(\boldsymbol{\omega}^{k+1})| < \epsilon$
```

It is also possible to find the optimal step size by performing line search, i.e., $\gamma = \arg\min_{\gamma \in [0,1]} M(\boldsymbol{\omega}^k + \gamma(\mathbf{s} - \boldsymbol{\omega}^k))$. In practice, it is good enough to use empirical step size as $\frac{2}{k+2}$. Besides, it is worthy to mention that the bottleneck of Algorithm 1 is the step of solving subproblems. However, additional speed-up can be obtained by running subproblems in parallel. Thus, the proposed algorithm for dual decomposition can run in $O(\max_i |\mathcal{X}_i|^{\text{tw}(G)})$ time, where $\text{tw}(G)$ is the treewidth of G. Finally, to obtain the primal integer solution, we use the majority voting to decide the final label for each node based on the labeling solutions of subproblems (Komodakis, Paragios, and Tziritas 2011).

Convergence Analysis

TODO: FW for min-max problem: non-differentiable, convex. Lemma 3 is incorrect in Eq.(21).

In this section, we will analyze the primal and primal-dual convergence of the proposed FWDD algorithm. We first introduce the curvature measure C_f for a sum of max functions $f(\boldsymbol{\omega}, \boldsymbol{\nu}) = \sum_i \max_{\boldsymbol{\nu}_i} h_i(\boldsymbol{\omega}_i, \boldsymbol{\nu}_i)$, which is similar to the definition for a general function in (Jaggi 2013). The quantity C_f is a positive constant that measures the nonlinearity of f. The definition is given as the largest quantity so that for all $\boldsymbol{\omega}, \boldsymbol{\eta}, \mathrm{s} \in \mathrm{dom}(f)$, with $\boldsymbol{\eta}$ colinear with $\boldsymbol{\omega}$ and s, i.e., $\boldsymbol{\eta} = \boldsymbol{\omega} + \gamma(\mathrm{s} - \boldsymbol{\omega})$ for some $\gamma \in [0, 1]$. Formally,

$$C_f = \sup_{\substack{\boldsymbol{\omega}, \mathbf{s} \\ \gamma \in [0,1] \\ \boldsymbol{\eta} = \boldsymbol{\omega} + \gamma(\mathbf{s} - \boldsymbol{\omega})}} \frac{2}{\gamma^2} \Big(f(\boldsymbol{\eta}) - f(\boldsymbol{\omega}) - \langle \boldsymbol{\eta} - \boldsymbol{\omega}, \nabla f(\boldsymbol{\omega}) \rangle \Big), \tag{18}$$

where $\nabla f(\boldsymbol{\omega}) = (\frac{\partial h_i(\boldsymbol{\omega}_i, \boldsymbol{\nu}_i^*)}{\partial \boldsymbol{\omega}_i}(\boldsymbol{\omega}_i)^\top)_i^\top$, and $\boldsymbol{\nu}_i^* = \arg\max_{\boldsymbol{\nu}_i} h_i(\boldsymbol{\omega}_i, \boldsymbol{\nu}_i)$. A bounded C_f assures that the maximum deviation between f and an arbitary linear lower bound of f is bounded.

Lemma 3. *Let C_M be the curvature measure of $M(\boldsymbol{\omega})$. Then,*

$$C_M \le nL\|\boldsymbol{\theta}\|^2, \tag{19}$$

where $n = |\mathcal{T}(G)|$ is the number of CPSGs, and $L \le |C(G)|$ is the Lipschitz constant.

Proof. Note that $M(\boldsymbol{\omega})$ is a sum of piecewise linear function. Let $E_T(\boldsymbol{\omega}_T, \boldsymbol{\nu}_T) = \langle \boldsymbol{\omega}_T, \boldsymbol{\nu}_T \rangle$ denote the linear component. Then, $S_T(\boldsymbol{\omega}_T) = \max_{\boldsymbol{\nu}_T \in \mathcal{L}_G} E_T(\boldsymbol{\omega}_T, \boldsymbol{\nu}_T)$. E_T is a linear function with Lipschitz continuous gradient, thus we have

$$\begin{aligned} E_T(\boldsymbol{\eta}_T, \boldsymbol{\nu}_T) \le E_T(\boldsymbol{\omega}_T, \boldsymbol{\nu}_T) + \langle \frac{\partial E_T}{\partial \boldsymbol{\omega}_T}(\boldsymbol{\omega}_T), \boldsymbol{\eta}_T - \boldsymbol{\omega}_T \rangle \\ + \frac{L}{2}\|\boldsymbol{\eta}_T - \boldsymbol{\omega}_T\|^2. \end{aligned} \tag{20}$$

Taking sum and max over both sides, we get

$$\begin{aligned} M(\boldsymbol{\eta}) \le M(\boldsymbol{\omega}) + \langle \nabla M(\boldsymbol{\omega}), \boldsymbol{\eta} - \boldsymbol{\omega} \rangle \\ + \frac{L}{2} \sum_{T \in \mathcal{T}(G)} \|\boldsymbol{\eta}_T - \boldsymbol{\omega}_T\|^2. \end{aligned} \tag{21}$$

Now, substituting the upper bound of $M(\boldsymbol{\eta})$ to the definition of C_M in Eq. (18), we can upper bound the curvature measure as

$$\begin{aligned} C_M &\le \sup_{\boldsymbol{\omega}, \mathbf{s}, \gamma} \frac{L}{\gamma^2} \sum_{T \in \mathcal{T}(G)} \|\boldsymbol{\eta}_T - \boldsymbol{\omega}_T\|^2 \\ &= \sup_{\boldsymbol{\omega}, \mathbf{s}} L \sum_{T \in \mathcal{T}(G)} \|\mathrm{s}_T - \boldsymbol{\omega}_T\|^2 \\ &\le nL\|\boldsymbol{\theta}\|^2, \end{aligned} \tag{22}$$

where L can pick a value such that the Lipschitz condition holds for all E_T's, thus $L \le \max_{\boldsymbol{\nu}_T \in \mathcal{L}_G} \boldsymbol{\nu}_T \le |C(G)|$. □

Theorem 4. *Frank-Wolfe based dual decomposition (Algorithm 1) obtains an ϵ-approximation solution after at most $O(\frac{nL\|\boldsymbol{\theta}\|^2}{\epsilon})$ iterations, that is, $M(\boldsymbol{\omega}^k) - M(\boldsymbol{\omega}^*) \le \mathrm{gap}(\boldsymbol{\omega}^k) \le \epsilon$ for $k \ge \lceil \frac{nL\|\boldsymbol{\theta}\|^2}{\epsilon} \rceil$.*

Proof. The result follows directly from standard convergence analysis of Frank-Wolfe algorithm (Jaggi 2013) and Lemma 3. Specifically, for $1 \le k \le K$, and $K \ge 1$, the duality gap $\mathbb{E}[\mathrm{gap}(\boldsymbol{\omega}^k)] \le \frac{6C_f}{K+1} \le \frac{6nL\|\boldsymbol{\theta}\|^2}{K+1}$. □

Experiments

We evaluate our FWDD algorithm on two structured prediction tasks and compare with state-of-the-art MAP inference algorithms: tree-reweighted max-product message passing (TRW) (Kolmogorov 2006) and max-product linear programming (MPLP) (Globerson and Jaakkola 2007). The results show that our FWDD algorithm outperforms previous methods.

Synthetic Image Denoising

The first task is binary image denoising with synthetic images. Given a CRF model, the task of inference is to recover the original image from the corrupted noisy image. We assume that the CRF structure can be modeled as a regular 4-connected grid graph, where each node corresponds to a pixel.

To learn the model parameters, we generate four noisy images with 32×32 pixels, in which three of them are used as training images and one for the final evaluation. A noisy image is constructed by a binary pattern (i.e., the ground truth) and Gaussian noise. For a single pixel, the intensity is given by $x_i = y_i(1 - g_i) + (1 - y_i)g_i$, where y_i is the ground truth label and g_i is the Gaussian noise random variable (can also be scaled). We use $(x_i, 1)$ as the unary feature vector for node i and use $(|x_i - x_j|, 1)$ as the pairwise feature vector for edge (i, j). For a fair comparison between different inference algorithms, we build two logistic regression models for learning unary and pairwise parameters separately without calling any CRF inference. This learning scheme is justified by (Sutton and Mccallum 2005).

We use two strategies to generate CPSGs as graph structures for subproblems. For the first strategy, we use spanning trees as CPSGs, that is, no specific loops are preserved in subproblems. As a comparison, we use slightly more complex CPSGs for the second strategy. Given a graph, we first randomly pick 10 disjoint 1×1 squares, and then generate spanning forest for the rest of the graph. Note that we do not need to build a junction tree for the first strategy, since the max-product tree inference can be applied directly in this case. The comparison between these two strategies in terms of linearized duality gap is shown in Fig. 2(c). Although the gap decrease of CPSG case is slower than the spanning tree case, we can see that our FWDD algorithm is sublinear convergence in both cases.

The testing image and its predicted labels are shown in Fig. 2. The comparison with TRW and MPLP is shown in Fig. 2(d). It can be seen that our FWDD algorithm obtains comparable performance in terms of primal and dual objectives and Lagrangian duality gaps.

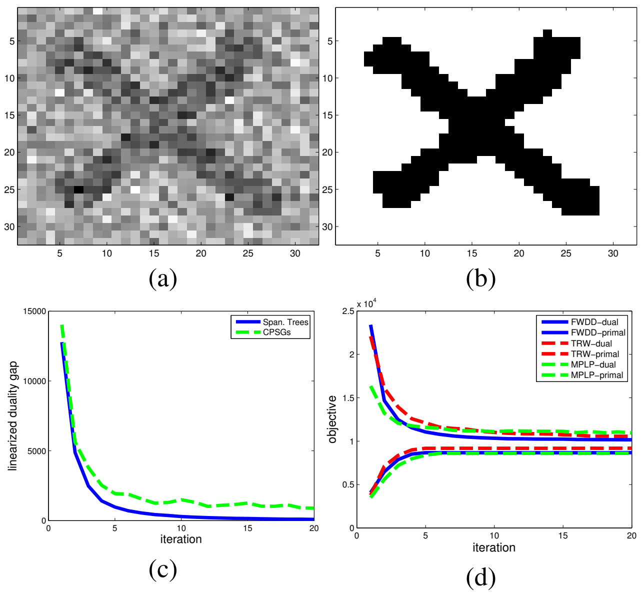

Figure 2: (a) The noisy testing image; (b) The predicted labels of (a); (c) The comparison of the linearized duality gap for our FWDD using spanning trees and CPSGs as subgraphs; (d) The comparison of integer primal objective and dual objective between TRW, MPLP and our FWDD.

Scene Labeling

We also evaluate our FWDD on real problems. One of the successful applications of structured prediction is the semantic scene labeling, which is a standard task in computer vision (Gould, Fulton, and Koller 2009; Socher et al. 2011). Given an image, the scene labeling task is to output the scene label of each pixel on the image. In this experiment, we use the Stanford Background Dataset (Gould, Fulton, and Koller 2009), which is a standard benchmark dataset consisting of 715 general images of natural scenes and 8 semantic classes.

Similar as (Gould, Fulton, and Koller 2009), we use pairwise Potts CRF model, where nodes are superpixels obtained by unsupervised spatial clustering, and edges are defined between neighboring superpixels. For the subproblems, we use CPSGs with triangle loops as subgraphs. Specifically, given a graph, we first search for loops with 3 nodes and randomly pick 10% such loops such that they are disjoint. Nodes and edges within these selected loops are treated as preserved clusters. Spanning forest is then generated for the rest of the graph. The final CPSG is constructed by combining the spanning forest and the preserved clusters. We keep generating CPSGs until the original graph is entirely covered.

For scene labeling, good unary and pairwise potentials depend heavily on superpixl and boundary features. We follow the feature extraction process recently evaluated as suitable for semantic segmentation in (Kae et al. 2013). Specifically, a superpixel is characterized by a feature vector consisting of the following properties: (1) texture: 64-dimensional histogram over textons; (2) color: 64-dimensional histogram of LAB colors generated by running K-means over pixels in the LAB color space; (3) layout: 12-dimensional histogram of pixel locations on a 3×4 grid. All these histograms are normalized separately, and normalized again after concatenating them together to form a 140-dimensional feature vector. The features for the pairwise potential are the difference of the superpixel features.

To decouple the learning of potential parameters with inference, piecewise training is performed for nodes and edges separately. This training scheme is quite reasonable and practical for Stanford background dataset. It has been shown that logistic regression and neural network model can achieve state-of-the-art performance with good mid-level features (Socher et al. 2011).

The standard evaluation metric of the scene labeling task is average pixel accuracy (i.e., the percentage of correctly labeled pixels). We define two CRF baselines using TRW and MPLP as MAP inference algorithm. Besides, we also compare with other state-of-the-art scene labeling methods in computer vision literature (Gould, Fulton, and Koller 2009; Munoz, Bagnell, and Hebert 2010; Socher et al. 2011). The comparison in terms of average pixel accuracy is shown in Table 1. We can see that FWDD outperforms other inference algorithms. For TRW and FWDD inference, apart from piecewise trained CRF model, we also train CRF model using structured SVM (Tsochantaridis et al. 2004). A general critisim of max-margin CRF model is that learning with approximate inference can result in bad model due to inexact gradient computation (Kulesza and Pereira 2007). It can be seen from Table 1 that CRF learning using FWDD inference gains small improvement comparing to that of TRW inference.

Table 1: The comparison of average pixel-wise accuracy on the Stanford background dataset.

Methods	Acc. %
Region Energy (Gould, Fulton, and Koller 2009)	76.4
Stack Hierarchy (Munoz, Bagnell, and Hebert 2010)	76.9
Recursive Neural Net (Socher et al. 2011)	78.1
CRF + TRW + Piecewise Training	76.33
CRF + MPLP + Piecewise Training	77.63
CRF + FWDD + Piecewise Training	**78.41**
CRF + TRW + Structured SVM	78.62
CRF + FWDD + Structured SVM	**79.66**

Conclusion

We presented a new Frank-Wolfe based algorithm using dual decomposition technique to solve linear programming relaxation of MAP inference. The equivalent dual problem is formulated as a constrained convex optimization, where the objective function becomes a sum of convex subproblems. We propose to use loopy spanning subgraph with bounded treewidth as graph structure for each subproblem, which can be solved efficiently by running max-product message passing on the junction tree. The master problem is solved by subgradient descent, where the descent direction is obtained by the linear minimization oracle. We also proved that the linear minimization oracle can be further decomposed and solved greedily. Our experiments on image denoising and large scale scene labeling show that FWDD performs on par with previous state-of-the-art MAP inference algorithms.

References

Domke, J. 2011. Dual decomposition for marginal inference. *AAAI*.

Franke, M., and Wolfe, P. 1956. An algorithm for quadratic programming. *Naval Res. Logis. Quart.*

Globerson, A., and Jaakkola, T. 2007. Fixing max-product: Convergent message passing algorithms for map lp-relaxations. In *NIPS*.

Gould, S.; Fulton, R.; and Koller, D. 2009. Decomposing a scene into geometric and semantically consistent regions. In *ICCV*.

Jaggi, M. 2013. Revisiting frank-wolfe: Projection-free sparse convex optimization. *ICML*.

Kae, A.; Sohn, K.; Lee, H.; and Learned-Miller, E. 2013. Augmenting CRFs with Boltzmann Machine Shape Priors for Image Labeling. *CVPR*.

Koller, D., and Friedman, N. 2009. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press.

Kolmogorov, V. 2006. Convergent tree-reweighted message passing for energy minimization. *PAMI*.

Komodakis, N.; Paragios, N.; and Tziritas, G. 2011. Mrf energy minimization and beyond via dual decomposition. *PAMI*.

Kulesza, A., and Pereira, F. 2007. Structured learning with approximate inference. In *NIPS*.

Martins, A. F. T.; Figueiredo, M. A. T.; Aguiar, P. M. Q.; Smith, N. A.; and Xing, E. P. 2014. Ad3: Alternating directions dual decomposition for map inference in graphical models. *JMLR*.

Meshi, O.; Sontag, D.; Jaakkola, T.; and Globerson, A. 2010. Learning efficiently with approximate inference via dual losses. *ICML*.

Meshi, O.; Jaakkola, T.; and Globerson, A. 2012. Convergence rate analysis of map coordinate minimization algorithms. *NIPS*.

Munoz, D.; Bagnell, J. A.; and Hebert, M. 2010. Stacked Hierarchical Labeling. In *ECCV*.

Rabiner, L. R. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proc. IEEE*.

Socher, R.; Lin, C. C.; Ng, A. Y.; and Manning, C. D. 2011. Parsing Natural Scenes and Natural Language with Recursive Neural Networks. In *ICML*.

Sontag, D.; Meltzer, T.; Globerson, A.; Weiss, Y.; and Jaakkola, T. 2008. Tightening LP relaxations for MAP using message-passing. In *UAI*.

Sontag, D.; Globerson, A.; and Jaakkola, T. 2010. Introduction to dual decomposition for inference. *Tech. Report*.

Sutton, C., and Mccallum, A. 2005. Piecewise training of undirected models. In *UAI*.

Tsochantaridis, I.; Hofmann, T.; Joachims, T.; and Altun, Y. 2004. Support vector learning for interdependent and structured output spaces. In *ICML*.

Wainwright, M., and Jordan, M. I. 2008. *Graphical Models, Exponential Families, and Variational Inference*. Found. Trends Mach. Learn.

Wainwright, M. J.; Jaakkola, T. S.; and Willsky, A. S. 2005. Map estimation via agreement on trees: Message-passing and linear programming. *PAMI*.

Werner, T. 2007. A linear programming approach to max-sum, a review. *PAMI*.

Yanover, C.; Meltzer, T.; and Weiss, Y. 2006. Linear programming relaxations and belief propagation – an empirical study. *JMLR*.

Yedidia, J. S.; Freeman, W. T.; and Weiss, Y. 2005. Constructing free energy approximations and generalized belief propagation algorithms. *IEEE Transactions on Information Theory, vol. 51, July 2005, pp. 2282-2313.*