Greedy Direction Method of Multiplier for MAP Inference of Large Output Domain

Xiangru Huang † **Ian E.H. Yen** ‡ **Ruohan Zhang** †
Qixing Huang † **Pradeep Ravikumar** ‡ **Inderjit S. Dhillon** †
† University of Texas at Austin ‡ Carnegie Mellon University

Abstract

Maximum-a-Posteriori (MAP) inference lies at the heart of Graphical Models and Structured Prediction. Despite the intractability of exact MAP inference, approximate methods based on LP relaxations have exhibited superior performance across a wide range of applications. Yet for problems involving large output domains (i.e., the state space for each variable is large), standard LP relaxations can easily give rise to a large number of variables and constraints which are beyond the limit of existing optimization algorithms. In this paper, we introduce an effective MAP inference method for problems with large output domains. The method builds upon alternating minimization of an Augmented Lagrangian that exploits the sparsity of messages through greedy optimization techniques. A key feature of our greedy approach is to introduce variables in an on-demand manner with a pre-built data structure over local factors. This results in a single-loop algorithm of sublinear cost per iteration and $O(\log(1/\epsilon))$-type iteration complexity to achieve ϵ sub-optimality. In addition, we introduce a variant of GDMM for binary MAP inference problems with a large number of factors. Empirically, the proposed algorithms demonstrate orders of magnitude speedup over state-of-the-art MAP inference techniques on MAP inference problems including Segmentation, Protein Folding, Graph Matching, and Multilabel prediction with pairwise interaction.

1 Introduction

Graphical Models and Structured Prediction have become prevalent in a wide range of applications including Computer Vision, Natural Language Processing, Social Networks, and Computational Biology. A common theme in these problems is that the elements in output space are highly dependent. Such dependencies are parametrized by *factors* in the language of graphical models, and the prediction of structured outputs can be formulated as a problem of *Maximum-a-Posteriori (MAP) inference*, which finds the *mode* of the output distribution — that is, an output configuration maximizing the summation of factor values.

Although MAP inference problems are NP-hard in general, approximation methods based on various convex relaxations have proven to be effective, e.g., Linear Programming (LP) [1], Quadratic Programming (QP) [2] and Semidefinite Programming (SDP) [3]. Among them the LP-based relaxation exhibits the best tradeoff between accuracy and efficiency (particularly on large-scale problems) and is arguably the most investigated approach in recent years [1, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14].

In the context of linear programming relaxations for MAP inference, efficient algorithms usually take advantage of the underlying graph structures. However, the number of variables in an LP relaxation still grows linearly with both the number of factors and the size of factor domains, where the latter can be quadratic or even cubic in the number of states of each random variable. Therefore, for problems with large output domains, most existing linear programming techniques become intractably slow. This paper addresses the following question: can we iterate over only a small portion of potential labels for each random variable and still ensure a desired convergence rate to an optimal solution?

The contribution of this research is two-folded. First, we show that a greedy version of the Augmented La-

Proceedings of the 20^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2017, Fort Lauderdale, Florida, USA. JMLR: W&CP volume 54. Copyright 2017 by the author(s).

grangian Method which only passes a sparse set of active messages can possess a $O(\log(1/\epsilon))$-type iteration complexity. In particular, we propose two variants of the greedy schemes—Frank-Wolfe (FW) and Block-Greedy Coordinate Descent (BGCD)— for problems with a large state space and problems with a large number of factors, respectively. Second, we show by maintaining a sparse set of messages, the generation of active states or active factors can be performed in time sublinear to the number of variables utilizing a pre-built data structure. Therefore the proposed algorithm is orders-of-magnitude faster than state-of-the-art MAP inference algorithms on several problems with large output domains, including Protein Folding, Graph Matching, and Pairwise-Interacted Multilabel Prediction.

2 Related Work

The optimization procedures of the existing LP relaxations for MAP inference share a similar underlying mechanism — alternating between local optimizations within each block of factors and message passing across different blocks. A fundamental tradeoff is between the cost of solving the local optimizations and the induced global convergence rate. A representative work with simple local optimizations is the *Projected Subgradient Dual Decomposition (PSDD)* algorithm [5]. Each of its local optimization depends on a single factor and optimizes a linear objective under a simplex constraint, hence admits a close-form solution via a simple enumeration. Yet the number of iterations for PSDD to find an ϵ-precise solution is $O(1/\epsilon^2)$. A representative work with faster convergence is the *Alternating Direction Dual Decomposition (AD3)* method [14] which has $O(1/\epsilon)$ outer iteration complexity. However, its local optimizations are based on *quadratic programmings*, which usually do not admit closed-form solutions and are solved via costly optimization sub-procedures. The AD3 algorithm exploits an active-set method to reduce the cost spent on the inner iterations, but the cost for variable selection is linear to the factor domain size. In contrast, our active-set method performs only a closed-form update w.r.t. the active set, and utilizes data structures to achieve a sublinear cost of variable selection.

The barrier to fast convergence of the *Dual Decomposition* techniques is their inherent non-smoothness of dual objectives due to LP relaxations. This non-smoothness also hinders global convergence for algorithms that replace the Subgradient Descent with the Block Coordinate Descent, such as *Tree-Reweighted Belief Propagation (TRWBP) [15]*, *Tree-Reweighted max-product message passing with Sequential update(TRW-S)* [7], and *Max-Product Linear Programming (MPLP)* [12]. In consequence, even though the Block-Coordinate Descent (TRW-S, MPLP) methods exhibit faster convergence rate, they often suffer from the issue of suboptimal fixed-points.

To achieve the fast convergence rate, a common strategy is to add strongly convex terms to the primal and/or dual objectives [9, 11]. However, such a methodology incurs another layer of relaxation due to these additional terms. Although these approaches still achieve $O(1/\epsilon)$ iteration complexity (e.g., when the Nersterov's acceleration technique is used [16]), their performance heavily relies on adjusting the weights of these additional terms carefully. Moreover, these approaches typically use full-gradient updates which obstruct exploiting the sparsity in the messages (dual variables). One exception is the work of [11] which utilizes the sparse optimizations on the unsmoothed objective function. However, this method only achieves an $O(1/\epsilon^2)$ iteration complexity.

Note that the per-iteration cost of all existing LP relaxation methods grows linearly with the size of an output domain. Apparently, this becomes intractable when the size of state space increases. In this work, we show how to exploit the sparsity of the messages in large-output-domain problems to achieve a sublinear cost per iteration and guarantee a fast convergence rate. Note a similar strategy could be also exploited at the training phase for some specific loss function such as structured support vector machine [17, 18].

3 Algorithm

In this section, we describe the proposed GDMM algorithm for MAP inference. We begin by describing a variant of the augmented Lagrangian to a linear programming relaxation for the MAP inference problem (Sec. 3.1). We then show how to optimize it efficiently using a greedy direction method of multiplier (Sec. 3.2). We also propose another variant to efficiently solve the problems with binary variables and a large number of factors (Sec. 3.3). For simplicity, we will focus our description on pairwise-interacted factors, but the technique can be easily generalized to higher order factors.

3.1 Augmented Lagrangian for MAP Inference

We study the standard MAP inference problem described by a factor graph $\mathcal{G} = \{\mathcal{V}, \mathcal{F}, \mathcal{E}\}$, where $\mathcal{V}, \mathcal{F}, \mathcal{E}$ are the sets of random variables, factors, and connecting edges, respectively. We denote the neighboring variables for each factor $f \in \mathcal{F}$ as $\partial(f) = \{i | (f, i) \in \mathcal{E}\}$ and similarly the neighboring factors for each variable

$i \in \mathcal{V}$ as $\partial(i) = \{f|(f,i) \in \mathcal{E}\}$. Each variable $i \in \mathcal{V}$ takes a value x_i in a finite label set, i.e. $x_i \in \mathcal{X}_i$. Variables and factors are associated with local score functions $\theta_i, \forall i \in \mathcal{V}$ and $\theta_f, f \in \mathcal{F}$, respectively.

The goal of the Maximum-a-Posteriori (MAP) problem is to find an assignment $\{x_i \in \mathcal{X}_i | i \in \mathcal{V}\}$ (i.e. each variable takes one value) that maximizes the sum of local score functions:

$$\max_{x} \quad \sum_{i \in \mathcal{V}} \theta_i(x_i) + \sum_{f \in \mathcal{F}} \theta_f(y_f) \tag{1}$$

Here $y_f \in \mathcal{Y}_f := \prod_{i \in \partial(f)} \mathcal{X}_i$ collects assignments from variables in $\partial(f)$. Note that the factors associated with only one variable can be absorbed by its neighboring variable, so we assume all factors are of order at least two.

We consider a standard LP relaxation for (1). Specifically, we replace x_i and y_f with marginal vectors $\boldsymbol{x}_i \in \Delta_i = \Delta^{|\mathcal{X}_i|}$ and $\boldsymbol{y}_f \in \Delta_f = \Delta^{|\mathcal{Y}_f|}$, where $\Delta^M = \{a \in \mathbb{R}^M | a \succeq \mathbf{0}, \sum_{i=1}^M a_i = 1\}$ is a simplex of dimension M. The marginal vectors are highly constrained with each other. The LP relaxations typically relax these constraints as linear constraints. In this paper, we consider standard linear constraints given by $\forall (f,i) \in \mathcal{E}, M_{if}\boldsymbol{y}_f = \boldsymbol{x}_i$, where

$$M_{if}(x_i, y_f) = \begin{cases} 1 & \text{if } y_f \sim x_i \\ 0 & \text{otherwise} \end{cases}.$$

and $y_f \sim x_i$ means that x_i is consistent with y_f's configuration.

Let $\boldsymbol{\theta}_i$ and $\boldsymbol{\theta}_f$ be vector representations of the local score functions, we derive the LP relaxation in this paper:

$$\begin{aligned} \max_{\substack{\boldsymbol{x}_i \in \Delta_i \\ \boldsymbol{y}_f \in \Delta_f}} \quad & \sum_{i \in \mathcal{V}} \boldsymbol{\theta}_i^T \boldsymbol{x}_i + \sum_{f \in \mathcal{F}} \boldsymbol{\theta}_f^T \boldsymbol{y}_f \\ s.t. \quad & \forall (f,i) \in \mathcal{E}, M_{if}\boldsymbol{y}_f = \boldsymbol{x}_i \end{aligned} \tag{2}$$

Applying the Augmented Lagrangian method, the objective function becomes (constraints remain the same):

$$\begin{aligned} \mathcal{L}(\{\boldsymbol{x}_i\}, \{\boldsymbol{y}_f\}) = & \sum_{i \in \mathcal{V}} (-\boldsymbol{\theta}_i)^T \boldsymbol{x}_i + \sum_{f \in \mathcal{F}} (-\boldsymbol{\theta}_f)^T \boldsymbol{y}_f \\ & + \sum_{(f,i) \in \mathcal{E}} \Big(\frac{\rho}{2} \| M_{if}\boldsymbol{y}_f - \boldsymbol{x}_i + \frac{1}{\rho}\boldsymbol{\mu}_{if}^t \|_2^2 \Big). \end{aligned} \tag{3}$$

For optimization, we alternate between optimizing the primal variables $\{\boldsymbol{x}_i, \boldsymbol{y}_f\}$ and the dual variables $\boldsymbol{\mu}_{if}$. Note that the primal constraints $\boldsymbol{x}_i \in \Delta_i$ and $\boldsymbol{y}_f \in \Delta_f$ are strictly enforced during optimization.

Algorithm 1 GDMM for Large Factor Domain

```
Initialization: $\forall i \in \mathcal{V}, \boldsymbol{x}_i \leftarrow \mathbf{0}, \mathcal{A}_i \leftarrow \emptyset$; $\forall f \in \mathcal{F}, \boldsymbol{y}_f \leftarrow \mathbf{0}, \mathcal{A}_f \leftarrow \emptyset$; $\forall (f,i) \in \mathcal{E}, \boldsymbol{\mu}_{if}^0 \leftarrow \mathbf{0}$.
repeat
  for $s = 1, 2, \cdots, S$ do
    for $\gamma = 1, 2, 4, 8, \cdots, \gamma_{\max}$ do
      1. $\forall f \in \mathcal{F}$, update $\mathcal{A}_f$, $\boldsymbol{y}_f(\mathcal{A}_f)$ according to (8) and (6) with $Q = \gamma Q_{max}$.
      2. $\forall i \in \mathcal{V}$, update $\mathcal{A}_i$, $\boldsymbol{x}_i(\mathcal{A}_i)$ according to (7) and (4)
      3. Break if $\mathcal{L}(\boldsymbol{x}', \boldsymbol{y}') - \mathcal{L}(\boldsymbol{x}, \boldsymbol{y}) \leq \sigma \langle \nabla_{\boldsymbol{y}} \mathcal{L}, \Delta \boldsymbol{y} \rangle$
    end for
  end for
  for $\forall (f,i) \in \mathcal{E}$ do
    3. Compute $\boldsymbol{\mu}_{if}^{t+1}$ based on (9)
  end for
  4. $t \leftarrow t + 1$
until primal/dual infeasibility $< \epsilon$
```

3.2 Greedy Direction Method of Multiplier for Factor of Large Domain

We proceed to describe the proposed greedy direction method of multiplier (or GDMM) for optimizing (3) (Please refer to Algorithm 1 for details). GDMM alternates between updating the dual variables and sequentially optimizing the individual primal variables $\boldsymbol{x}_i$ and $\boldsymbol{y}_f$ w.r.t. a judiciously selected active set of states within the factor. However, the cost of such optimization is still expensive when the label space is large. This leads to the key component of this paper, i.e., by maintaining active sets of coordinates for each variable, one can pass only *active messages* to the nearby factors. Once the messages become sparse, the selection of active variables can be implemented efficiently using pre-built data structures. These active sets are initialized as empty sets and are gradually augmented in an on-demand manner. Our approach is motivated by the observations that the non-zero entries in $\boldsymbol{x}_i$ and $\boldsymbol{y}_f$ are relatively sparse and the size of these active sets are small throughout the optimization. These observations could result in significant speedup of the algorithm. Next, we present each step of the GDMM method in detail: optimizing the primal variables with respect to the active sets, updating the active sets, and updating the dual variables.

Primal variable optimization. The primal variables are optimized by the Fully-corrective Frank-Wolfe (FCFW) with approximate correction [19] which alternates between optimizing $\{\boldsymbol{x}_i\}$ and $\{\boldsymbol{y}_f\}$ until a certain precision is achieved. For each $i \in V$, we use $\mathcal{A}_i$ and $\bar{\mathcal{A}}_i$ to denote its active set and the complementary set, respectively. The subproblem of (3) w.r.t. $\mathcal{A}_i$

is given by

$$\min_{\substack{\boldsymbol{x}_i\in\Delta_i\\ \boldsymbol{x}_i(\bar{\mathcal{A}}_i)=\mathbf{0}}} -\boldsymbol{\theta}_i^T\boldsymbol{x}_i + \frac{\rho}{2}\sum_{f\in\partial(i)}\|M_{if}\boldsymbol{y}_f - \boldsymbol{x}_i + \frac{1}{\rho}\boldsymbol{\mu}_{if}^t\|_2^2$$

which is equivalent to the simplex projection problem

$$\min_{\substack{\boldsymbol{x}_i\in\Delta_i\\ \boldsymbol{x}_i(\bar{\mathcal{A}}_i)=\mathbf{0}}} \left\|\boldsymbol{x}_i - \left(\sum_{f\in\partial(i)}(\frac{M_{if}\boldsymbol{y}_f}{|\partial(i)|}+\frac{1}{\rho}\boldsymbol{\mu}_{if}^t)+\frac{\boldsymbol{\theta}_i}{\rho|\partial(i)|}\right)\right\|_2^2 \tag{4}$$

As in [20], (4) can be solved via a simplex projection operation with time complexity $\mathrm{O}(|\mathcal{A}_i|\log(|\mathcal{A}_i|)$. This also implies that, when fixing $\boldsymbol{\mu}$, $\boldsymbol{x}$ can be written as a simple function $\boldsymbol{x}(\boldsymbol{y})$.

Now we consider optimizing the variables associated with each factor $f\in\mathcal{F}$. Let $\mathcal{A}_f$ and $\bar{\mathcal{A}}_f$ be its active set and the complementary set. The subproblem w.r.t. $\mathcal{A}_f$ is

$$\begin{aligned}&\min_{\substack{\boldsymbol{y}_f\in\Delta_f\\ \boldsymbol{y}_f(\bar{\mathcal{A}}_f)=\mathbf{0}}} \mathcal{L}_f(\boldsymbol{x},\boldsymbol{y};\boldsymbol{\mu})\\ &:= -\boldsymbol{\theta}_f^T\boldsymbol{y}_f + \frac{\rho}{2}\sum_{i\in\partial(f)}\|M_{if}\boldsymbol{y}_f - \boldsymbol{x}_i + \frac{1}{\rho}\boldsymbol{\mu}_{if}^t\|_2^2\end{aligned} \tag{5}$$

Instead of solving (5) exactly, we propose to minimize a quadratic upper bound of the function (5). Let $Q=\rho\|M\|^2$ where $\|.\|$ is the spectral norm. (5) becomes:

$$\min_{\substack{\boldsymbol{y}_f^+\in\Delta_f\\ \boldsymbol{y}_f^+(\bar{\mathcal{A}}_f)=\mathbf{0}}} \frac{Q}{2}\|\boldsymbol{y}_f^+ - \boldsymbol{y}_f\|_2^2 + \nabla_{\boldsymbol{y}_f}\mathcal{L}^T(\boldsymbol{y}_f^+ - \boldsymbol{y}_f) \tag{6}$$

where $\nabla_{\boldsymbol{y}_f}\mathcal{L} = \sum_{i\in\partial(f)} M_{if}^T(\boldsymbol{\mu}_{if}^t - \rho\boldsymbol{x}_i) - \boldsymbol{\theta}_f$. Note the quadratic upper bound (6) is tighter than that used in the Proximal Gradient Descent scheme described in [21] by a factor of $\frac{|\mathcal{Y}_f|}{|\mathcal{A}_f|}$. It can be solved by a simplex projection

$$\min_{\substack{\boldsymbol{y}_f^+\in\Delta_f\\ \boldsymbol{y}_f^+(\bar{\mathcal{A}}_f)=\mathbf{0}}} \|\boldsymbol{y}_f^+ - (\boldsymbol{y}_f - \frac{1}{Q}(\sum_{i\in\partial(f)} M_{if}^T(\boldsymbol{\mu}_{if}^t - \rho\boldsymbol{x}_i) - \boldsymbol{\theta}_f))\|_2^2$$

in time $\mathrm{O}(|\mathcal{A}_f|\log(|\mathcal{A}_f|))$. In practice, computing Q could be expensive, so we introduce a line-search procedure over Q, starting from a lower bound Q_{max} where $Q_{max} := \max_{f\in\mathcal{F}}\ Q_{\mathcal{A}_f}$ and $Q_{\mathcal{A}_f} = \rho\|[M_{if}]_{\mathcal{A}_f}\|^2$. In practice, we found the descent amount often passes the line-search condition without backtracking. Although we introduce a fully-corrective loop ($s=1...S$) in Algorithm 1 for ease of our analysis, we found setting $S=1$ suffices for fast convergence in our experiments. Note by maintaining a small set of non-zero variables, we limit the size of messages (non-zero dual variables), yielding an efficient scheme for finding new active variables as introduced next.

Updating the active sets. The active sets are updated by finding a currently non-active coordinate with largest gradient magnitude. For each $i\in\mathcal{V}$, we update the corresponding active set as

$$\mathcal{A}_i \leftarrow \mathcal{A}_i \cup \arg\max_{x_i\notin\mathcal{A}_i} \nabla_{x_i}\mathcal{L}_i(\boldsymbol{x},\boldsymbol{y};\boldsymbol{\mu}) \tag{7}$$

where $\nabla\mathcal{L}_{x_i}(\boldsymbol{x},\boldsymbol{y};\boldsymbol{\mu}) = -\sum_{f\in\partial(i)}[\rho(M_{if}\boldsymbol{y}_f - \boldsymbol{x}_i) + \boldsymbol{\mu}_{if}^t]_{x_i} - \boldsymbol{\theta}_i(x_i)$. Similarly, for each $f\in\mathcal{F}$, we update the corresponding active set as

$$\mathcal{A}_f \leftarrow \mathcal{A}_f \cup \arg\max_{y_f\notin\mathcal{A}_f} \nabla_{y_f}\mathcal{L}_f(\boldsymbol{x},\boldsymbol{y};\boldsymbol{\mu}) \tag{8}$$

where $\nabla_{y_f}\mathcal{L}_f(\boldsymbol{x},\boldsymbol{y};\boldsymbol{\mu}) = \sum_{i\in\partial(f)}[M_{if}^T(\rho(M_{if}\boldsymbol{y}_f - \boldsymbol{x}_i) + \boldsymbol{\mu}_{if}^t)]_{y_f} - \boldsymbol{\theta}_f(y_f)$. Note that the new active coordinates can be searched efficiently as long as the messages passed from nearby factors $(M_{if}\boldsymbol{y}_f - \boldsymbol{x}_i)$ are sparse. Specifically, as $\boldsymbol{\theta}_i$ and $\boldsymbol{\theta}_f$ are constant, we can build *sorted lists* on $\boldsymbol{\theta}_i$, $\boldsymbol{\theta}_f$ in the preprocessing phase. When executing our algorithm, the selection (7) can be done in time $O(|\mathcal{A}_i| + \sum_{f\in\partial(i)}|\mathcal{A}_f|)$ since the size of non-zero messages is bounded by $\sum_{f\in\partial(i)}|\mathcal{A}_f|$.

The time complexity of the coordinate selection of factor based on (8) is $\mathrm{O}(|\mathcal{A}_i||\mathcal{A}_{i'}| + |\mathcal{A}_f|)$, where the messages are denoted by $\boldsymbol{\delta}_{if} = \rho(M_{if}\boldsymbol{y}_f - \boldsymbol{x}_i) + \boldsymbol{\mu}_{if}^t$. For a pairwise-interaction factor $f=(i,i')\in\mathcal{F}$ with $y_f = (x_i, x_{i'})$ and $i,i'\in\mathcal{V}$, we can find a coordinate of largest gradient magnitude (8) by searching in four divisions of the coordinates: (i) $\{(x_i,x_{i'})|, \boldsymbol{\delta}_{if}(x_i) = \boldsymbol{\delta}_{i'f}(x_{i'}) = 0\}$, (ii) $\{(x_i,x_{i'})|\boldsymbol{\delta}_{if}(x_i) = 0\}$, (iii) $\{(x_i,x_{i'})|\boldsymbol{\delta}_{i'f}(x_{i'}) = 0\}$ and (iv) $\{(x_i,x_{i'})|\boldsymbol{\delta}_{if}(x_i) \neq 0, \boldsymbol{\delta}_{i'f}(x_{i'}) \neq 0\}$. In particular, (i) can be done in $\mathrm{O}(1)$ via a sorted list on $\boldsymbol{\theta}_f$. (ii), (iii) can be done in time $\mathrm{O}(|\mathcal{A}_i||\mathcal{A}_{i'}| + |\mathcal{A}_f|)$ via sorted lists built on the sets $\{\theta_f(x_i,x_{i'})|x_i = k\}$, $\{\theta_f(x_i,x_{i'})|x_{i'} = k\}$ respectively. The complexity of (iv) is $O(|\mathcal{A}_i||\mathcal{A}_{i'}|)$.

To maintain a compact active set, after solving (4) and (6), we remove all coordinates x_i, y_f with $\boldsymbol{x}_i(x_i) = 0$, $\boldsymbol{y}_f(y_f) = 0$ from $\mathcal{A}_i$, $\mathcal{A}_f$, respectively.

Updating the dual After optimizing the primal variables, the dual variables are updated as

$$\boldsymbol{\mu}_{if}^{t+1} = \boldsymbol{\mu}_{if}^t + \eta(M_{if}\boldsymbol{y}_f - \boldsymbol{x}_i), \qquad \forall i\in\mathcal{V},\ f\in\mathcal{F}, \tag{9}$$

where η is the dual step size. In the analysis section we show for a sufficiently small η, Algorithm 1 globally converges to the optimum.

Algorithm 2 GDMM for Large Number of Factors

```
Initialization: $\forall i \in \mathcal{V}, \boldsymbol{x}_i \leftarrow \mathbf{0}$; $\forall f \in \mathcal{F}, \boldsymbol{y}_f \leftarrow \mathbf{0}$; $\forall (f,i) \in \mathcal{E}, \boldsymbol{\mu}_{if}^0 \leftarrow \mathbf{0}; t \leftarrow 0$. $\mathcal{A}_\mathcal{V} \leftarrow \emptyset$; $\mathcal{A}_\mathcal{F} \leftarrow \emptyset$.
repeat
  1. Find $i^*$ satisfying (10), $\mathcal{A}_\mathcal{V} \leftarrow \mathcal{A}_\mathcal{V} \cup \{i^*\}$.
  2. $\mathcal{A}_\mathcal{F} \leftarrow \mathcal{A}_\mathcal{F} \cup \{f \in \mathcal{F} | \partial(f) \subseteq \mathcal{A}_\mathcal{V}\}$
  3. $\forall i \in \mathcal{A}_\mathcal{V}$, update $\boldsymbol{x}_i(\mathcal{X}_i)$ according to (4).
  4. Find $f^*$ satisfying (11), $\mathcal{A}_\mathcal{F} \leftarrow \mathcal{A}_\mathcal{F} \cup \{f^*\}$.
  5. $\forall f \in \mathcal{A}_\mathcal{F}$, update $\boldsymbol{y}_f(\mathcal{Y}_f)$ according to (6).
  for $\forall f \in \mathcal{A}_\mathcal{F}$, $i \in \partial(f)$ do
    6. Update $\boldsymbol{\mu}_{if}^{t+1}$ base on (9)
  end for
  7. $t \leftarrow t+1$
until primal/dual infeasibility $< \epsilon$
```

3.3 GDMM for a Large Number of Factors

For applications such as alignment, segmentation and multilabel prediction, the factor graphs comprise a large number of factors with binary random variables. In this section, we introduce a variant of the GDMM specifically for such problems. The procedure is sketched in Algorithm 2.

The idea behind Algorithm 2 is to maintain two active sets for variables and factors, denoted as $\mathcal{A}_\mathcal{V} \subseteq \mathcal{V}$ and $\mathcal{A}_\mathcal{F} \subseteq \mathcal{F}$, respectively. Then we ensure that at each iteration a variable i^* satisfying

$$i^* = \underset{i}{argmin} \min_{\boldsymbol{x}_i + \boldsymbol{d} \in \Delta_i} \langle \nabla_{\boldsymbol{x}_i} \mathcal{L}_i(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\mu}), \boldsymbol{d} \rangle + \frac{1}{2} \|\boldsymbol{d}\|^2 \tag{10}$$

and a factor f^* satisfying

$$f^* = \underset{f}{argmin} \min_{\boldsymbol{y}_f + \boldsymbol{d} \in \Delta_f} \langle \nabla_{\boldsymbol{y}_f} \mathcal{L}_f(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\mu}), \boldsymbol{d} \rangle + \frac{Q_f}{2} \|\boldsymbol{d}\|^2. \tag{11}$$

are in the active sets (i.e. $i^* \in \mathcal{A}_\mathcal{V}$ and $f^* \in \mathcal{A}_\mathcal{F}$).

Note that all variables are binary and all factors have only 4 states, i.e. $\mathcal{Y}_f = \{(0,0), (0,1), (1,0), (1,1)\}$. Therefore one can always write $x_i(0) = 1 - x_i(1)$ and write $y_f(0,0), y_f(0,1), y_f(1,0)$ as a function of $y_f(1,1), x_j(1)$ and $x_i(1)$ by enforcing consistency constraints $y_f(1,0) + y_f(1,1) = x_i(1)$ and $y_f(0,1) + y_f(1,1) = x_j(1)$ for a factor connecting to two variables i and j. Then one can verify the remaining constraints between factors and variables are:

$$\begin{aligned} x_i(1) - y_f(1,1) = y_f(1,0) \geq 0 \\ x_j(1) - y_f(1,1) = y_f(0,1) \geq 0 \\ 1 - x_j(1) - x_i(1) + y_f(1,1) = y_f(0,0) \geq 0 \end{aligned} \tag{12}$$

One can encode (12) as the new set of constraints $M_{if} \boldsymbol{y}_f - \boldsymbol{x}_i = 0$ by introducing slack variables for the inequalities. Under this scheme, we show an efficient method to include all variables and factors that satisfy (10) and (11) in the active sets.

We achieve this by first pushing any inactive f into $\mathcal{A}_\mathcal{F}$ if its neighboring variables i and j are both active. Note this accounts for only a small fraction of the factors if $|\mathcal{A}_\mathcal{V}| \ll |\mathcal{V}|$, and thus can be computed efficiently. Then we consider any inactive factor f with either i or j inactive. For this type of factor, we have (12) satisfied and $\boldsymbol{\mu}_{if} = \boldsymbol{\mu}_{jf} = \mathbf{0}$, and the gradient of each factor $y_f := y_f(1,1)$ is simply

$$\nabla_{y_f} \mathcal{L}_f(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\mu}) = -\theta_f(1,1).$$

Therefore, the search of active factor (11) reduces to finding the minimal $\theta_f(1,1)$ among $f \notin \mathcal{A}_\mathcal{F}$, which can be easily done via a sorted list of $-\boldsymbol{\theta}_f(1,1)$. For each iteration, we add the first inactive f from the head until we have $-\boldsymbol{\theta}_f(1,1) > 0$ which can be done with $O(1)$ amortized cost. In our experiments, only a small fraction of $\mathcal{F}$ will be added to $\mathcal{A}_\mathcal{F}$. One can refer to Appendix 9 for the statistics of active size $|\mathcal{A}_\mathcal{F}|$ in our experiments.

4 Convergence Analysis

In this section, we show the GDMM algorithms for the large factor domains (Algorithm 1) and a large number of factors (Algorithm 2) admit linear convergence. To state the main result formally, we introduce the following notations. We use $\boldsymbol{z}^t = (\boldsymbol{x}^t, \boldsymbol{y}^t)$ to encapsulate the primal variables, and denote the feasible space as $\mathcal{M} := \{(\boldsymbol{x}, \boldsymbol{y}) \in \ | \ \boldsymbol{x}_i \in \Delta_i, \forall i \in \mathcal{V}; \boldsymbol{y}_f \in \Delta_f, \forall f \in \mathcal{F}\}$. Then the Augmented Lagrangian is expressed as

$$\mathcal{L}(\boldsymbol{z}, \boldsymbol{\mu}) = \langle -\boldsymbol{\theta} + M^T \boldsymbol{\mu}, \boldsymbol{z} \rangle + \frac{\rho}{2} \|M\boldsymbol{z}\|^2,$$

where $M\boldsymbol{z} = \mathbf{0}$ collects all the constraints of the form

$$M_{if} \boldsymbol{y}_f - \boldsymbol{x}_i = \mathbf{0}, \quad \forall (i,f) \in \mathcal{E}.$$

To assess the convergence rate, denote a primal minimizer respect to a dual iterate $\boldsymbol{\mu}^t$ at iteration t as

$$\bar{\boldsymbol{z}}^t := arg \min_{\boldsymbol{z} \in \mathcal{M}} \mathcal{L}(\boldsymbol{z}, \boldsymbol{\mu}^t). \tag{13}$$

The convergence rate is measured by the primal gap Δ_p^t and the dual gap Δ_d^t:

$$\begin{aligned} \Delta_p^t &:= \mathcal{L}(\boldsymbol{z}^{t+1}, \boldsymbol{\mu}^t) - \mathcal{L}(\bar{\boldsymbol{z}}^t, \boldsymbol{\mu}^t) \\ \Delta_d^t &:= d^* - \mathcal{L}(\bar{\boldsymbol{z}}^t, \boldsymbol{\mu}^t) \end{aligned}$$

where $d^* := \max_{\boldsymbol{\mu}} \min_{\boldsymbol{z} \in \mathcal{M}}; \mathcal{L}(\boldsymbol{z}, \boldsymbol{\mu})$ is the optimal dual objective value. The following part states two major results for the convergence of the GDMM in the form of both Algorithm 1 and Algorithm 2.

Theorem 1 (Convergence of Algorithm 1). *Let $Q = \rho \|M\|^2$. For any constant dual step size η satisfying*

$$0 < \eta \leq \frac{\rho}{4(1 + |\mathcal{F}|Q/m_{\mathcal{M}})},$$

the iterates given by Algorithm 1 obey

$$\Delta_p^t + \Delta_d^t \leq \epsilon, \ \forall t \geq \max\left\{2(1 + \frac{|\mathcal{F}|Q}{m_{\mathcal{M}}}), \frac{\tau}{\eta}\right\} \log(\frac{1}{\epsilon}).$$

Here $m_{\mathcal{M}}$ is the generalized geometric strong convexity constant of function $\mathcal{L}(., \boldsymbol{\mu})$ on the domain $\mathcal{M}$ (defined in Lemma 5), and τ is a constant characterized in Lemma 4.

Theorem 2 (Convergence of Algorithm 2). *Let $Q_{\max} := \max_{f \in \mathcal{F}} Q_f$. For any constant dual step size η satisfying*

$$0 < \eta \leq \frac{\rho}{4(1 + Q_{max}/m_1)},$$

the iterates given by Algorithm 2 have

$$\Delta_p^t + \Delta_d^t \leq \epsilon, \ \forall t \geq \max\left\{2(1 + \frac{Q_{max}}{m_1}), \frac{\tau}{\eta}\right\} \log(\frac{1}{\epsilon})$$

where m_1 is a generalized strong convexity constant of function $\mathcal{L}(., \boldsymbol{\mu})$ measured in ℓ_1-norm (defined in Lemma 19).

Theorem 1 and 2 are based on the analysis framework of [22] for the Augmented Lagrangian Method, as well as the recent results on the linear convergence of Frank-Wolfe variants [23] and Greedy Coordinate Descent for non-strongly convex functions of the form (13) [24, 25]. A proof outline can be find in Appendix A followed by a detailed proof in Appendix B.

5 Experimental Evaluation

Here we evaluate the GDMM algorithm experimentally. Section 5.1 discusses the experimental setup. Section 5.2 analyzes the performance of the GDMM and compare it with state-of-the-art MAP inference techniques.

5.1 Experimental Setup

Benchmark datasets. We select five datasets that involve large output domains. These datasets are selected from two benchmarks: OPENGM[2] and *The Probabilistic Inference Challenge 2011* (PASCAL)[3]. As shown in Table 1, they cover a diverse set of examples with varying scales, network connectivities, and output domain sizes. To help understand the performance of the algorithms, we briefly describe each dataset:

Dataset	$\lvert\mathcal{X}_i\rvert$	$\lvert\mathcal{Y}_f\rvert$	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{F}\rvert$
Segmentation	21	441	226	842
ImageAlignment	83	6889	400	3334
MultiLabel	2	4	3884	7544670
Protein	404	163216	37	703
GraphMatching	1034	1069156	188	1864

Table 1: Data Statistics [1]. $|\mathcal{X}_i|$ denotes the maximum domain size of an output variable, $|\mathcal{Y}_f|$ is the maximum domain size of a factor, and $|\mathcal{V}|$, $|\mathcal{F}|$ denote the number of nodes, factors respectively.

- **Protein.** [26, 27] A Protein Folding dataset that is included in both PASCAL and OpenGM. Its factor graph has bigram factors between every pair of variables.
- **Graph Matching.** A real world social network matching dataset generated using Facebook network data from SNAP[4]. We attempt to match subgraphs to the original graph. In particular, we construct the factor graph as follows: 1. one variable for each node in a subgraph, with vertex set of the original graph as its domain. We use inner product of the features provided from SNAP to generate the unigram factors. 2. For each induced edge on the subgraph, we introduce a bigram factor for the two variables and use inverse of the shortest distance on the graph to generate the bigram factor.
- **Segmentation.** A dataset from PASCAL which has a grid-4 neighborhood structure.
- **ImageAlignment.** A dataset from PASCAL.
- **Multilabel.** A multilabel data set from Mulan [5]. The MAP inference problem is generated from a trained model and an instance from the testset.

Compared Algorithms. The compared algorithms include several top performing algorithms as well as widely used MAP inference algorithms.

- TRWS [7]: a variant of the *Tree-Reweighted max-product message passing* (TRW) [1] algorithm that a) decomposes the graph into *monotonic chains* instead of trees, and b) updates messages

[1] The specific instances we used for each dataset are: 16_16_s.21.uai (Segmentation), fileforGal_400markers.uai (ImageAlignment), EurLex (Multilabel), and 2BBN.uai (Protein)

[2] `http://hciweb2.iwr.uni-heidelberg.de/opengm/`

[3] `http://www.cs.huji.ac.il/project/PASCAL/showNet.php`

[4] `http://snap.stanford.edu`

[5] `http://mulan.sourceforge.net/datasets-mlc.html`

Dataset	Segmentation		ImageAlignment		Multilabel		Protein		Graph Matching	
Algorithm	Time	Primal	Time	Primal	Time	Primal	Time	Primal	Time	Primal
LBP	0.406s	**-252.39**	4.136s	-6907	5012.6s	**899.06**	86407s	11904	17696s	**3733**
TRWBP	1.6336s	**-252.39**	17.2s	-6907	25474s	**899.06**	75205s	11888	1598s	**3733**
TreeEP	11.514s	**-252.39**	2666s	-6916.95	2863.2s	**899.06**	2010.4s	11974	TLE	TLE
HAK	6.75s	**-252.39**	274.5s	-6908	TLE	TLE	10013s	10740	87886s	3717
GBP	0.22s	**-252.39**	6.07s	-6908	TLE	TLE	NE	NE	3432s	3721
SoftBCFW	20.89s	-356.57	5977s	-19572	54.22s	889.32	1838s	9250	MLE	MLE
LPsparse	0.57s	**-252.39**	370s	-6913.1	N/A	N/A	32000s	12179	MLE	MLE
SmoothMSD	11.19s	**-252.39**	6462s	**-6907.6**	53462s	893.66	13648s	11670	15312s	**3733**
MPLP	0.12s	**-252.39**	**9.85s**	**-6907.6**	45514s	846.64	124.3s	11771	212.5s	3730
PSDD	2.596s	-319.38	531.4s	**-6907.6**	1503s	**899.06**	20865s	10601.6	1192.9s	**3733**
AD^3	0.15s	**-252.39**	60.2s	-6917.09	243.03s	**899.06**	48137s	12000.8	1659s	**3733**
TRW-S	**0.0046s**	**-252.39**	0.65s	-6907	21.89s	**899.06**	39.67s	12067	47.3s	3732
GDMM	0.26s	**-252.39**	18.09s	**-6907.6**	**16.02s**	**899.06**	**661s**	**12263**	**25.9s**	**3733**

Table 2: Primal : best decoded (integer solution) primal objective; Time: time taken to reach the decoded primal objective. The shortest time taken to reach the best MAP objective among all methods are marked with boldface. For all experiments we set memory limit = 100G and time limit = 2 day. Experiments violating these limits are marked MLE: Memory Limit Exceeded, or TLE: Time Limit Exceeded; NE means the solver throws error message "quantity not normalizable".

according to a global order on each chain. We used implementation from the OPENGM Library [28, 29], which wraps the original TRWS code.

- AD^3: The *Alternating Directions Dual Decomposition* algorithm with public implementation [6] provided by the authors of [14].
- PSDD: The *Projected Subgradient Dual Decomposition* algorithm by [5]. Its implementation is contained in the AD^3 code.
- MPLP: The *Max-Product Linear Programming* algorithm [8, 12, 13] with public implementation [7]. For a fair comparison, we disabled its tightening process to make sure that GDMM and MPLP optimize the same objective function.
- According to the comparison figures in [11], we add two leading methods in terms of performance.
  - SmoothMSD: as a variant of the coordinate minimization method on smoothed dual (the *CD soft* in [11]), we implemented the smooth Max-Sum Diffusion (MSD) algorithm in [30].
  - SoftBCFW: the *Block-Coordinate Frank-Wolfe* for soft-constrained primal, described as Algorithm 1 by [11].

For each experiment, we choose $\gamma \in \{10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}\}$ for SmoothMSD and $\lambda \in \{1, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}\}$ for SoftBCFW. Among parameters that give the best decoded primal objective, we select the fastest.

- LPsparse: A recently proposed general-purpose LP solver for problems with sparse structures [31].
- Algorithm implemented in libDAI [32], including the *Loopy Belief Propagation* (LBP) [33], the *Tree-Reweighted Belief Propagation* (TRWBP) [15], the *Tree Expectation Propagation* (TreeEP)[34], the *Double-loop GBP* (HAK) [35], and *Generalized Belief Propagation* (GBP) [36].

In all experiments, we pick $\rho \in \{1, 0.1, 0.01, 0.001\}$ and $\eta \in \{0.1, 1\}$ for GDMM. Generally speaking, the performance of MAP inference algorithms is largely determined by the structure and domain size of the factor graph. Among these five datasets, Segmentation and Image Alignment represent factor graphs with small domain sizes and a small number of factors; Multilabel represents graphs with a large number of factors; Protein and Graph Matching represent graphs with a large number of factors and large domain sizes.

5.2 Benchmark Evaluation

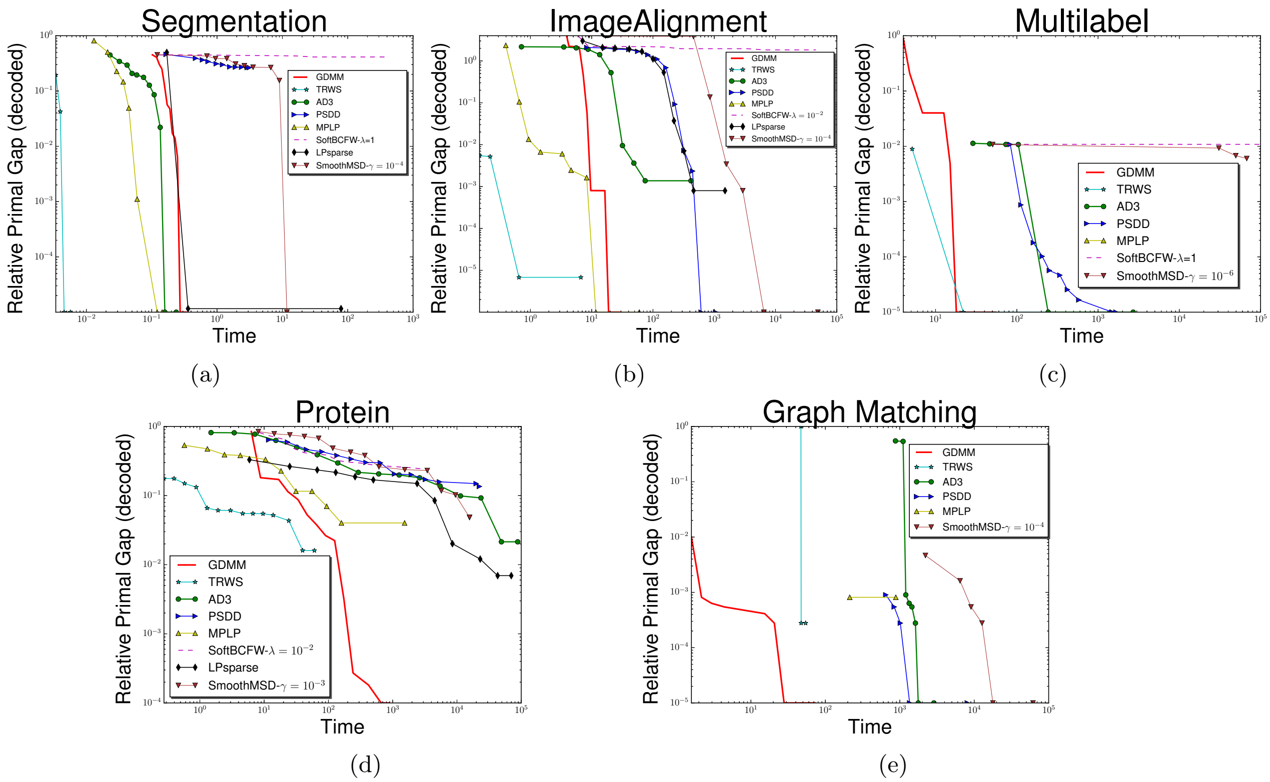
In this section, we present our benchmark evaluation results. All participating methods are compared in terms of running time and decoded (integer solution) primal objective. Table 2 presents the statistics, and Figure 1 illustrates the convergence behavior.

Overall, GDMM is the top-performing algorithm. It returns the best solutions among all five datasets. In terms of running time, it is also competent with the top algorithms on each dataset.

On the grid-4 structured factor graphs such as Segmentation, TRWS and MPLP are generally fast and accurate. But they are also less accurate on several exam-

[6] http://www.cs.cmu.edu/~ark/AD3/

[7] http://cs.nyu.edu/~dsontag/code/README_v2.html



(a)

(b)

(c)

(d)

(e)

Figure 1: Convergence results for data sets with (a)(b): small #state and small #factor; (c): large #factor; (d),(e): large #states. In this paper we only consider decoded (integer solution) primal objective. Relative Primal Gap (decoded) is defined as $\frac{P^*-P}{|P^*|}$ where P is the current primal objective and P^* is the best primal objective found among all algorithms.

ples. AD^3 could provide more accurate solutions, but its active-set method introduces a variable selection procedure that has cost linear to the factor domain size, which results in the slow convergence on Protein and Graph Matching. In contrast, GDMM takes advantage of the active sets of size sublinear to the factor domains (see Appendix C for details), which leads to orders of magnitude speedup. Algorithms such as SoftBCFW and SmoothMSD minimize over a smoothed objective function. However SoftBCFW can not explicityly enforce consistency constraints and hence often stuck at solutions with poor quality. Meanwhile SmoothMSD can provide solutions with good quality at the cost of slow convergence.

Another key feature of the GDMM is its small memory overhead. Most algorithms require memory that is linear in the sum of factor domain sizes, due to the dense messages and primal variables. For example, for the Graph Matching dataset, the memory consumptions for the compared algorithms are AD^3 / PSDD: 69G, MPLP: 45G, TRWS:13G, LPsparse / SoftBCFW:>100G. GDMM addresses this issue by maintaining an active set and sparse messages, results in a memory footprint of only 260M.

6 Conclusions and Future Work

In this paper, we introduce GDMM, an efficient algorithm for MAP inference of large output domain. GDMM combines the power of greedy method (FW, GCD) with ALM to achieve a cost sublinear to the domain size. We show that despite the usage of active sets, the algorithm still admits a linear-type convergence rate, and experimental results on benchmark datasets reveal superiority of the algorithm on large-scale problems. As a future work, we plan to develop algorithm that integrates our approach with the fast convergence of TRWS.

Acknowledgement Q.H. acknowledges the support of NSF via DMS-1700234. I.D. acknowledges the support of NSF via CCF-1320746, IIS-1546452, and CCF-1564000. P.R. acknowledges the support of ARO via W911NF-12-1-0390 and NSF via IIS-1149803, IIS-1320894, IIS-1447574, DMS-1264033, and NIH via R01 GM117594-01 as part of the Joint DMS/NIGMS Initiative to Support Research at the Interface of the Biological and Mathematical Sciences.

References

[1] Martin J Wainwright, Tommi S Jaakkola, and Alan S Willsky. Map estimation via agreement on trees: message-passing and linear programming. *IEEE transactions on information theory*, 51(11):3697–3717, 2005.

[2] Pradeep Ravikumar and John Lafferty. Quadratic programming relaxations for metric labeling and markov random field map estimation. In *Proceedings of the 23rd international conference on Machine learning*, pages 737–744. ACM, 2006.

[3] Roy Frostig, Sida Wang, Percy S Liang, and Christopher D Manning. Simple map inference via low-rank relaxations. In *Advances in Neural Information Processing Systems*, pages 3077–3085, 2014.

[4] Pradeep Ravikumar, Alekh Agarwal, and Martin J Wainwright. Message-passing for graph-structured linear programs: Proximal methods and rounding schemes. *Journal of Machine Learning Research*, 11(Mar):1043–1080, 2010.

[5] Nikos Komodakis, Nikos Paragios, and Georgios Tziritas. Mrf optimization via dual decomposition: Message-passing revisited. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8. IEEE, 2007.

[6] Sebastian Nowozin, Peter V Gehler, Jeremy Jancsary, and Christoph H Lampert. *Advanced Structured Prediction*. MIT Press, 2014.

[7] Vladimir Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *IEEE transactions on pattern analysis and machine intelligence*, 28(10):1568–1583, 2006.

[8] David Sontag, Do Kook Choe, and Yitao Li. Efficiently searching for frustrated cycles in map inference. In *28th Conference on Uncertainty in Artificial Intelligence, UAI 2012*, 2012.

[9] Vladimir Jojic, Stephen Gould, and Daphne Koller. Accelerated dual decomposition for map inference. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 503–510, 2010.

[10] Ofer Meshi, Mehrdad Mahdavi, Adrian Weller, and David Sontag. Train and test tightness of lp relaxations in structured prediction. 2016.

[11] Ofer Meshi, Mehrdad Mahdavi, and Alex Schwing. Smooth and strong: Map inference with linear convergence. In *Advances in Neural Information Processing Systems*, pages 298–306, 2015.

[12] 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, 2008.

[13] David Sontag, Talya Meltzer, Amir Globerson, Tommi S Jaakkola, and Yair Weiss. Tightening lp relaxations for map using message passing. *arXiv preprint arXiv:1206.3288*, 2012.

[14] André FT Martins, Mário AT Figueiredo, Pedro MQ Aguiar, Noah A Smith, and Eric P Xing. Ad3: Alternating directions dual decomposition for map inference in graphical models. *Journal of Machine Learning Research*, 16:495–545, 2015.

[15] Martin J Wainwright, Tommi S Jaakkola, and Alan S Willsky. Tree-reweighted belief propagation algorithms and approximate ml estimation by pseudo-moment matching. In *AISTATS*, 2003.

[16] Yu Nesterov. Smooth minimization of non-smooth functions. *Mathematical programming*, 103(1):127–152, 2005.

[17] Ian En-Hsu Yen, Xiangru Huang, Kai Zhong, Ruohan Zhang, Pradeep K Ravikumar, and Inderjit S Dhillon. Dual decomposed learning with factorwise oracle for structural svm of large output domain. In *Advances in Neural Information Processing Systems*, pages 5024–5032, 2016.

[18] Ian En-Hsu Yen, Xiangru Huang, Kai Zhong, Pradeep Ravikumar, and Inderjit S Dhillon. Pd-sparse: A primal and dual sparse approach to extreme multiclass and multilabel classification. In *Proceedings of the 33nd International Conference on Machine Learning*, 2016.

[19] Simon Lacoste-Julien and Martin Jaggi. On the global linear convergence of frank-wolfe optimization variants. In *Advances in Neural Information Processing Systems*, pages 496–504, 2015.

[20] John Duchi, Shai Shalev-Shwartz, Yoram Singer, and Tushar Chandra. Efficient projections onto the l 1-ball for learning in high dimensions. In *Proceedings of the 25th international conference on Machine learning*, pages 272–279. ACM, 2008.

[21] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

[22] M. Hong and Z. Luo. On linear convergence of alternating direction method of multipliers. *arXiv*, 2012.

[23] Simon Lacoste-Julien and Martin Jaggi. On the global linear convergence of frank-wolfe optimization variants. In *Advances in Neural Information Processing Systems*, pages 496–504, 2015.

[24] Julie Nutini, Mark Schmidt, Issam H Laradji, Michael Friedlander, and Hoyt Koepke. Coordinate descent converges faster with the gauss-southwell rule than random selection. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 1632–1641, 2015.

[25] Ian En-Hsu Yen, Cho-Jui Hsieh, Pradeep K Ravikumar, and Inderjit S Dhillon. Constant nullspace strong convexity and fast convergence of proximal methods under high-dimensional settings. In *Advances in Neural Information Processing Systems*, pages 1008–1016, 2014.

[26] Gal Elidan, Amir Globerson, and Uri Heinemann. Pascal 2011 probabilistic inference challenge, 2012.

[27] Chen Yanover, Ora Schueler-Furman, and Yair Weiss. Minimizing and learning energy functions for side-chain prediction. *Journal of Computational Biology*, 15(7):899–911, 2008.

[28] Bjoern Andres, Thorsten Beier, and Jörg H Kappes. Opengm: A c++ library for discrete graphical models. *arXiv preprint arXiv:1206.0111*, 2012.

[29] Björn Andres, Jörg H Kappes, Ullrich Köthe, Christoph Schnörr, and Fred A Hamprecht. An empirical comparison of inference algorithms for graphical models with higher order factors using opengm. In *Joint Pattern Recognition Symposium*, pages 353–362. Springer, 2010.

[30] Tomáš Werner. Revisiting the decomposition approach to inference in exponential families and graphical models. *Center for Machine Perception, Czech Technical University Prague, Research Report, CTU-CMP-2009-06, ftp://cmp. felk. cvut. cz/pub/cmp/articles/werner/Werner-TR-2009-06. pdf*, 2009.

[31] Ian En-Hsu Yen, Kai Zhong, Cho-Jui Hsieh, Pradeep K Ravikumar, and Inderjit S Dhillon. Sparse linear programming via primal and dual augmented coordinate descent. In *Advances in Neural Information Processing Systems*, pages 2368–2376, 2015.

[32] Joris M. Mooij. libDAI: A free and open source C++ library for discrete approximate inference in graphical models. *Journal of Machine Learning Research*, 11:2169–2173, August 2010.

[33] Frank R Kschischang, Brendan J Frey, and H-A Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on information theory*, 47(2):498–519, 2001.

[34] Yuan Qi and TP Minka. Tree-structured approximations by expectation propagation. *Advances in Neural Information Processing Systems (NIPS)*, 16:193, 2004.

[35] Tom Heskes, Kees Albers, and Bert Kappen. Approximate inference and constrained optimization. In *Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence*, pages 313–320. Morgan Kaufmann Publishers Inc., 2002.

[36] Jonathan S Yedidia, William T Freeman, and Yair Weiss. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Transactions on Information Theory*, 51(7):2282–2312, 2005.

[37] A.J. Hoffman. On approximate solutions of systems of linear inequalities. *Journal of Research of the National Bureau of Standards*, 1952.

7 Proof Roadmap

The key in proving Theorem 1 and 2 is to establish bounds on the primal-dual progress $\Delta_p^t + \Delta_d^t - \Delta_p^{t-1} - \Delta_d^{t-1}$. As intermediate steps, the two lemmas below bound the dual-progress $\Delta_d^t - \Delta_d^{t-1}$ and the primal-progress $\Delta_p^t - \Delta_p^{t-1}$ with respect to the primal variables $\{\boldsymbol{z}^t\}$ and the optimal primal variables $\{\bar{\boldsymbol{z}}^t\}$ at each iteration.

Lemma 1 (Dual Progress)**.** *The dual progress is upper bounded as*

$$\Delta_d^t - \Delta_d^{t-1} \le -\eta(M\boldsymbol{z}^t)^T(M\bar{\boldsymbol{z}}^t). \tag{14}$$

Lemma 2 (Primal Progress)**.** *The primal progress is upper bounded as*

$$\begin{aligned}\Delta_p^t - \Delta_p^{t-1} \le \mathcal{L}(\boldsymbol{z}^{t+1}, \boldsymbol{\mu}^t) - \mathcal{L}(\boldsymbol{z}^t, \boldsymbol{\mu}^t)\\ + \eta\|M\boldsymbol{z}^t\|^2 - \eta\langle M\boldsymbol{z}^t, M\bar{\boldsymbol{z}}^t\rangle\end{aligned}$$

By combining results of Lemma 1 and 2, we obtain an intermediate upper bound on the primal-dual progress:

$$\begin{aligned}&\Delta_d^t - \Delta_d^{t-1} + \Delta_p^t - \Delta_p^{t-1}\\ &\le \eta\|M\boldsymbol{z}^t - M\bar{\boldsymbol{z}}^t\|^2 - \eta\|M\bar{\boldsymbol{z}}^t\|^2\\ &+ \mathcal{L}(\boldsymbol{z}^{t+1}, \boldsymbol{\mu}^t) - \mathcal{L}(\boldsymbol{z}^t, \boldsymbol{\mu}^t)\end{aligned} \tag{15}$$

The following four lemmas provide upper bounds on the three sub-terms in (15), i.e., $\|M\boldsymbol{z}^t - M\bar{\boldsymbol{z}}^t\|^2$, $-\eta\|M\bar{\boldsymbol{z}}^t\|^2$, and $\mathcal{L}(\boldsymbol{z}^{t+1}, \boldsymbol{\mu}^t) - \mathcal{L}(\boldsymbol{z}^t, \boldsymbol{\mu}^t)$, where the bounds on the last term are algorithm-dependent and therefore are tackled by Lemma 5 and Lemma 19 for Algorithm 1 and Algorithm 2 respectively.

Lemma 3.

$$\|M\boldsymbol{z}^t - M\bar{\boldsymbol{z}}^t\|^2 \le \frac{2}{\rho}(\mathcal{L}(\boldsymbol{z}^t, \boldsymbol{\mu}^t) - \mathcal{L}(\bar{\boldsymbol{z}}^t, \boldsymbol{\mu}^t)). \tag{16}$$

Lemma 4 (*Hong and Luo 2012*)**.** *There is a constant $\tau > 0$ such that*

$$\Delta_d(\boldsymbol{\mu}) \le \tau\|M\bar{\boldsymbol{z}}(\boldsymbol{\mu})\|^2. \tag{17}$$

for any $\boldsymbol{\mu}$ in the dual domain and any primal minimizer $\bar{\boldsymbol{z}}(\boldsymbol{\mu})$ satisfying (13).

Lemma 5. *The descent amount of Augmented Lagrangian function produced by one pass of FCFW (in Algorithm 1) has*

$$\begin{aligned}&\mathcal{L}(\boldsymbol{z}^{t+1}, \boldsymbol{\mu}^t) - \mathcal{L}(\boldsymbol{z}^t, \boldsymbol{\mu}^t)\\ &\le -\frac{m_{\mathcal{M}}}{2|\mathcal{F}|Q}(\mathcal{L}(\boldsymbol{z}^t, \boldsymbol{\mu}^t) - \mathcal{L}(\bar{\boldsymbol{z}}^t, \boldsymbol{\mu}^t))\end{aligned} \tag{18}$$

where $Q = \rho\|M\|^2$.

Lemma 6. *The descent amount of Augmented Lagrangian function produced by iterations of Algorithm 2 has*

$$\begin{aligned}&\mathcal{L}(\boldsymbol{z}^{t+1}, \boldsymbol{\mu}^t) - \mathcal{L}(\boldsymbol{z}^t, \boldsymbol{\mu}^t)\\ &\le \frac{-m_1}{Q_{max}}(\mathcal{L}(\boldsymbol{z}^t, \boldsymbol{\mu}^t) - \mathcal{L}(\bar{\boldsymbol{z}}^t, \boldsymbol{\mu}^t))\end{aligned} \tag{19}$$

where $Q_{max} = \max_{f\in\mathcal{F}} Q_f$ and

$$m_1 := \frac{1}{\max\{16\theta_1\Delta\mathcal{L}^0, 2\theta_1(1+4L_g^2)/\rho, 6\}} \tag{20}$$

is the generalized strong convexity constant *for function $\mathcal{L}(., \boldsymbol{\mu})$. Here $\Delta\mathcal{L}^0$ is a bound on $\mathcal{L}(\boldsymbol{z}^0, \boldsymbol{\mu}^t) - \mathcal{L}(\bar{\boldsymbol{z}}^0, \boldsymbol{\mu}^t)$, L_g is local Lipschitz-continuous constant of the function $g(\boldsymbol{x}) := \|\boldsymbol{x}\|^2$, and θ_1 is the Hoffman constant depending on the geometry of optimal solution set.*

Now we are ready to prove Theorem 1 and 2.

Proof of Theorem 1. Let $\kappa = m_{\mathcal{M}}/(|\mathcal{F}|Q)$. By lemma 5 and (15), we have

$$\begin{aligned}&\Delta_d^t - \Delta_d^{t-1} + \Delta_p^t - \Delta_p^{t-1}\\ \le&\frac{-\kappa}{1+\kappa}\left(\mathcal{L}(\boldsymbol{z}^t, \boldsymbol{\mu}^t) - \mathcal{L}(\bar{\boldsymbol{z}}^t, \boldsymbol{\mu}^t)\right)\\ &+ \frac{2\eta}{\rho}(\mathcal{L}(\boldsymbol{z}^t, \boldsymbol{\mu}^t) - \mathcal{L}(\bar{\boldsymbol{z}}^t, \boldsymbol{\mu}^t)) - \eta\|M\bar{\boldsymbol{z}}^t\|^2.\end{aligned} \tag{21}$$

Then by choosing $\eta < \frac{\kappa\rho}{2(1+\kappa)}$, we have guaranteed descent on $\Delta_p + \Delta_d$ for each GDMM iteration. By choosing $\eta \le \frac{\kappa\rho}{4(1+\kappa)}$, we have

$$\begin{aligned}&(\Delta_d^t + \Delta_p^t) - (\Delta_d^{t-1} + \Delta_p^{t-1})\\ \le&\frac{-\kappa}{2(1+\kappa)}\left(\mathcal{L}(\boldsymbol{z}^t, \boldsymbol{\mu}^t) - \mathcal{L}(\bar{\boldsymbol{z}}^t, \boldsymbol{\mu}^t)\right) - \eta\|M\bar{\boldsymbol{z}}^t\|^2\\ \le&\frac{-\kappa}{2(1+\kappa)}\Delta_d^t - \frac{\eta}{\tau}\Delta_d^t\\ \le&-\min\left(\frac{\kappa}{2(1+\kappa)}, \frac{\eta}{\tau}\right)(\Delta_p^t + \Delta_d^t)\end{aligned}$$

where the second inequality is from Lemma 4. We thus obtain a recursion of the form

$$\Delta_d^t + \Delta_p^t \le \frac{1}{1 + \min(\frac{\kappa}{2(1+\kappa)}, \frac{\eta}{\tau})}\left(\Delta_d^{t-1} + \Delta_p^{t-1}\right),$$

which then leads to the conclusion. □

The proof of Theorem 2 is the same as above except that the definition of κ is changed to m_1/Q_{max} and Lemma 5 is replaced by Lemma 19.

8 Proof of Lemmas

Proof of Lemma 1.

$$\begin{aligned}\Delta_d^t - \Delta_d^{t-1} &= \mathcal{L}(\bar{\boldsymbol{z}}^{t-1}, \boldsymbol{\mu}^{t-1}) - \mathcal{L}(\bar{\boldsymbol{z}}^t, \boldsymbol{\mu}^t)\\ &\leq \mathcal{L}(\bar{\boldsymbol{z}}^t, \boldsymbol{\mu}^{t-1}) - \mathcal{L}(\bar{\boldsymbol{z}}^t, \boldsymbol{\mu}^t)\\ &= \langle \boldsymbol{\mu}^{t-1} - \boldsymbol{\mu}^t, M\bar{\boldsymbol{z}}^t\rangle\\ &= -\eta\langle M\boldsymbol{z}^t, M\bar{\boldsymbol{z}}^t\rangle\end{aligned}$$

where the first inequality follows from the optimality of $\bar{\boldsymbol{z}}^{t-1}$ for the function $\mathcal{L}(\boldsymbol{z}, \boldsymbol{\mu}^{t-1})$ defined by $\boldsymbol{\mu}^{t-1}$, and the last equality follows from the dual update (9). □

Proof of Lemma 2.

$$\begin{aligned}&\Delta_p^t - \Delta_p^{t-1}\\ =&\mathcal{L}(\boldsymbol{z}^{t+1}, \boldsymbol{\mu}^t) - \mathcal{L}(\boldsymbol{z}^t, \boldsymbol{\mu}^{t-1}) - (d(\boldsymbol{\mu}^t) - d(\boldsymbol{\mu}^{t-1}))\\ \leq&\mathcal{L}(\boldsymbol{z}^{t+1}, \boldsymbol{\mu}^t) - \mathcal{L}(\boldsymbol{z}^t, \boldsymbol{\mu}^t) + \mathcal{L}(\boldsymbol{z}^t, \boldsymbol{\mu}^t) - \mathcal{L}(\boldsymbol{z}^t, \boldsymbol{\mu}^{t-1})\\ &+ (d(\boldsymbol{\mu}^{t-1}) - d(\boldsymbol{\mu}^t))\\ \leq&\mathcal{L}(\boldsymbol{z}^{t+1}, \boldsymbol{\mu}^t) - \mathcal{L}(\boldsymbol{z}^t, \boldsymbol{\mu}^t) + \eta\|M\boldsymbol{z}^t\|^2 - \eta\langle M\boldsymbol{z}^t, M\bar{\boldsymbol{z}}^t\rangle\end{aligned}$$

where the last inequality uses Lemma 1 on $d(\boldsymbol{\mu}^{t-1}) - d(\boldsymbol{\mu}^t) = \Delta_d^t - \Delta_d^{t-1}$. □

Proof of Lemma 3. Introduce

$$\tilde{\mathcal{L}}(\boldsymbol{z}, \boldsymbol{\mu}) = h(\boldsymbol{z}) + G(M\boldsymbol{z}),$$

where

$$G(M\boldsymbol{z}) = \frac{\rho}{2}\|M\boldsymbol{z}\|^2,$$

and

$$h(\boldsymbol{z}) = \langle -\boldsymbol{\theta}, \boldsymbol{z}\rangle + \langle \boldsymbol{\mu}, M\boldsymbol{z}\rangle + \boldsymbol{I}_{\boldsymbol{z}\in\mathcal{M}}.$$

Here

$$\boldsymbol{I}_{\boldsymbol{z}\in\mathcal{M}} = \begin{cases} 0 & \boldsymbol{z} \in \mathcal{M},\\ \infty & \text{otherwise.}\end{cases}$$

As feasibility is strictly enforced during primal updates, we have

$$\tilde{\mathcal{L}}(\bar{\boldsymbol{z}}^t, \boldsymbol{\mu}^t) = \mathcal{L}(\bar{\boldsymbol{z}}^t, \boldsymbol{\mu}^t), \quad \tilde{\mathcal{L}}(\boldsymbol{z}^t, \boldsymbol{\mu}^t) = \mathcal{L}(\boldsymbol{z}^t, \boldsymbol{\mu}^t). \tag{22}$$

As $\bar{\boldsymbol{z}}^t$ is a critical point of $\mathcal{L}(\boldsymbol{z}, \boldsymbol{\mu}^t)$, and by definition, $\mathcal{L}(\boldsymbol{z}, \boldsymbol{\mu}^t) \leq \tilde{\mathcal{L}}(\boldsymbol{z}, \boldsymbol{\mu}^t)$, we obtain,

$$0 \in \partial_{\boldsymbol{z}}\tilde{\mathcal{L}}(\bar{\boldsymbol{z}}^t, \boldsymbol{\mu}^t) = \partial h(\bar{\boldsymbol{z}}^t) + M^T\nabla G(M\bar{\boldsymbol{z}}^t).$$

Note that $h(\cdot)$ is convex, it follows that

$$h(\boldsymbol{z}^t) - h(\bar{\boldsymbol{z}}^t) \geq \langle \boldsymbol{v}, \boldsymbol{z}^t - \bar{\boldsymbol{z}}^t\rangle, \qquad \forall \boldsymbol{v} \in \partial h(\bar{\boldsymbol{z}}^t). \tag{23}$$

Moreover,

$$\begin{aligned}&G(M(\boldsymbol{z}^t)) - G(M(\bar{\boldsymbol{z}}^t)) \qquad (24)\\ &= \frac{\rho}{2}(\|M\boldsymbol{z}^t\|^2 - \|M\bar{\boldsymbol{z}}^t\|^2)\\ &= \frac{\rho}{2}(\boldsymbol{z}^t - \bar{\boldsymbol{z}}^t)^T M^T M(\boldsymbol{z}^t + \bar{\boldsymbol{z}}^t)\\ &= \rho(\boldsymbol{z}^t - \bar{\boldsymbol{z}}^t)^T M^T M\bar{\boldsymbol{z}}^t + \frac{\rho}{2}(\boldsymbol{z}^t - \bar{\boldsymbol{z}}^t)^T M^T M(\boldsymbol{z}^t - \bar{\boldsymbol{z}}^t)\\ &= \langle M^T\nabla G(M\bar{\boldsymbol{z}}^t), \boldsymbol{z}^t - \bar{\boldsymbol{z}}^t\rangle + \frac{\rho}{2}\|M\boldsymbol{z}^t - M\bar{\boldsymbol{z}}^t\|^2. \qquad (25)\end{aligned}$$

Combing (22), (23), and (25), we arrive at

$$\mathcal{L}(\boldsymbol{z}^t, \boldsymbol{\mu}^t) - \mathcal{L}(\bar{\boldsymbol{z}}^t, \boldsymbol{\mu}^t) \geq \frac{\rho}{2}\|M(\boldsymbol{z}^t) - M(\bar{\boldsymbol{z}}^t)\|^2.$$

□

Proof of Lemma 4. This is a lemma adapted from [22]. Since our primal objective (2) is a linear function with each block of primal variables $\boldsymbol{x}_i$ (or $\boldsymbol{y}_f$) constrained in a simplex domain, it satisfies the *assumptions A(a)—A(e)* and *A(g)* in [22]. Then Lemma 3.1 of [22] guarantees that, as long as $\|\nabla d(\boldsymbol{\mu})\|$ is always bounded, there is a constant $\tau > 0$ s.t.

$$\Delta_d(\boldsymbol{\mu}) \leq \tau\|\nabla d(\boldsymbol{\mu})\|^2 = \|M\bar{\boldsymbol{z}}(\boldsymbol{\mu})\|^2$$

for all $\boldsymbol{\mu}$ in the dual domain. Note our problem satisfies the condition of bounded gradient magnitude since

$$\begin{aligned}\|\nabla d(\boldsymbol{\mu})\| &= \|M\bar{\boldsymbol{z}}(\boldsymbol{\mu})\| \leq \|M\bar{\boldsymbol{z}}(\boldsymbol{\mu})\|_1\\ &\leq \|M\|_1\|\bar{\boldsymbol{z}}(\boldsymbol{\mu})\|_1 \leq (\max_f |\mathcal{Y}_f|)(|\mathcal{F}| + |\mathcal{V}|)\end{aligned}$$

where the last inequality is because each block of variables in $\bar{\boldsymbol{z}}(\boldsymbol{\mu})$ lie in a simplex domain. □

Proof of Lemma 5. Recall that the Augmented Lagrangian $\mathcal{L}(\boldsymbol{z}, \boldsymbol{\mu})$ is of the form

$$\mathcal{L}(\boldsymbol{z}, \boldsymbol{\mu}) = \langle -\boldsymbol{\theta} + M^T\boldsymbol{\mu}, \boldsymbol{z}\rangle + G(M\boldsymbol{z}) \;, \forall i \in \mathcal{V} \tag{26}$$

where M is the matrix that encodes all constraints of the form

$$M_{if}\boldsymbol{z}_f - \boldsymbol{z}_i = \begin{bmatrix} M_{if} & -I_i \end{bmatrix}\begin{bmatrix} \boldsymbol{z}_f \\ \boldsymbol{z}_i \end{bmatrix} = \boldsymbol{0}.$$

and function $G(\boldsymbol{w}) = \frac{\rho}{2}\|\boldsymbol{w}\|^2$ is strongly convex with parameter ρ. Let

$$H(\boldsymbol{z}) := \mathcal{L}(\boldsymbol{z}, \boldsymbol{\mu}). \tag{27}$$

Since we are minimizing the function subject to a convex, polyhedral domain $\mathcal{M}$, by Theorem 10 of [23], we have the *generalized geometrical strong convexity* constant $m_{\mathcal{M}}$ of the form

$$m_{\mathcal{M}} := m(PWidth(\mathcal{M}))^2 \tag{28}$$

where $PWidth(\mathcal{M}) > 0$ is the pyramidal width of the simplex domain $\mathcal{M}$ and m is the *generalized strong convexity* constant of function (26) (defined by Lemma 9 of [23]). By definition of the geometric strong convexity constant, we have

$$H(\boldsymbol{z}) - H^* \leq \frac{g_{FW}^2}{2m_{\mathcal{M}}} \tag{29}$$

from (23) in [23], where $g_{FW} := \langle \nabla H(\boldsymbol{z}), \boldsymbol{v}_{FW} - \boldsymbol{v}_A\rangle$. $\boldsymbol{v}_{FW}$ is the greedy Frank-Wolfe (FW) direction

$$\boldsymbol{v}_{FW} := arg \min_{\boldsymbol{v}\in\mathcal{M}} \langle \nabla H(\boldsymbol{z}), \boldsymbol{v}\rangle \tag{30}$$

and $\boldsymbol{v}_A$ is the away direction

$$\boldsymbol{v}_A := arg \max_{\boldsymbol{v}\in\mathcal{M}} \langle \nabla \tilde{H}(\boldsymbol{z}), \boldsymbol{v}\rangle \tag{31}$$

where

$$\nabla_k \tilde{H}(\boldsymbol{z}) = \left\{ \begin{array}{ll} \nabla_k H(\boldsymbol{z}), & z_k \neq 0 \\ -\infty, & o.w. \end{array} \right.$$

Then let $m = |\mathcal{F}|$ be the number of factors. For each inner iteration s of the Fully-Corrective FW, by minimizing subproblem (5) w.r.t. an active set that contains the FW direction and also the away direction (by the definition (31)), we have, for any $\forall \gamma \in [0, 1]$,

$$H(\boldsymbol{z}^{t+1}) - H(\boldsymbol{z}^t) \leq \gamma g^t_{FW} + mQ\gamma^2. \tag{32}$$

Suppose the minimizer of (32) $\gamma^* = -\frac{g^t_{FW}}{2mQ}$ has $\gamma^* < 1$, we have

$$H(\boldsymbol{z}^{t+1}) - H(\boldsymbol{z}^t) \leq -\frac{g^{t2}_{FW}}{4mQ} \tag{33}$$

Otherwise, let $\gamma^* = 1$, we have

$$\begin{aligned} & H(\boldsymbol{z}^{t+1}) - H(\boldsymbol{z}^t) \\ & \leq g^t_{FW} + mQ \leq \frac{g^t_{FW}}{2} < -\frac{g^{t2}_{FW}}{2mQ} \leq -\frac{g^{t2}_{FW}}{4mQ}, \end{aligned}$$

where the second inequality holds since $-\frac{g^t_{FW}}{2Qm} \geq 1$.

Combining with the error bound (29), we have

$$H(\boldsymbol{z}^{t+1}) - H(\boldsymbol{z}^t) \leq -\frac{m_{\mathcal{M}}(H(\boldsymbol{z}^t) - H^*)}{2mQ}. \tag{34}$$

□

Proof of Lemma 19.

For problem of the form (13), the optimal solution is profiled by the polyhedral set $\mathcal{S} := \{\boldsymbol{z} \mid M\boldsymbol{z} = \boldsymbol{t}^*,\ \boldsymbol{\Delta}^T\boldsymbol{z} = s^*,\ \boldsymbol{z} \in \mathcal{M}\}$ for some $\boldsymbol{t}^*$, s^*. Denoting $\bar{\boldsymbol{z}} := \Pi_{\mathcal{S}}(\boldsymbol{z})$, we can bound the distance of any feasible point $\boldsymbol{z}$ to its projection $\Pi_{\mathcal{S}}(\boldsymbol{z})$ to set $\mathcal{S}$ by

$$\begin{aligned} \|\bar{\boldsymbol{z}} - \boldsymbol{z}\|^2_{2,1} &= (\sum_{f\in\mathcal{F}} \|\bar{\boldsymbol{z}}_f - \boldsymbol{z}_f\|_2)^2 \\ &\leq \theta_1 \left(\|M\boldsymbol{z} - \boldsymbol{t}^*\|^2 + \|\boldsymbol{\Delta}^T\boldsymbol{z} - s^*\|^2 \right) \end{aligned} \tag{35}$$

where θ_1 is a constant depending on the set $\mathcal{S}$, using the Hoffman's inequality [37].

Then for each iteration t of the Algorithm 2, consider the descent amount produced by the update w.r.t. the selected factor satisfying (11). We have

$$\begin{aligned} & H(\boldsymbol{z}^{t+1}) - H(\boldsymbol{z}^t) \\ & \leq \min_{\boldsymbol{z}^t_{f^*} + \boldsymbol{d}_{f^*} \in \Delta_{f^*}} \langle \nabla_{\boldsymbol{z}_{f^*}} H, \boldsymbol{d}_{f^*}\rangle + \frac{Q_{\max}}{2}\|\boldsymbol{d}_{f^*}\|^2 \\ & = \min_{\boldsymbol{z}^t + \boldsymbol{d} \in \mathcal{M}} \sum_{f\in\mathcal{F}} \langle \nabla_{\boldsymbol{z}_f} H, \boldsymbol{d}_f\rangle + \frac{Q_{\max}}{2} \left(\sum_{f\in\mathcal{F}} \|\boldsymbol{d}_f\| \right)^2 \end{aligned} \tag{36}$$

where the second equality is from the definition (11) of f^*.

Then we have

$$\begin{aligned} & H(\boldsymbol{z}^{t+1})] - H(\boldsymbol{z}^t) \\ & \leq \min_{\boldsymbol{z}^t + \boldsymbol{d} \in \mathcal{M}} \left(\sum_{f\in\mathcal{F}} \langle \nabla_{\boldsymbol{z}_f} H, \boldsymbol{d}_f\rangle + \frac{Q_{\max}}{2} \Big(\sum_{f\in\mathcal{F}} \|\boldsymbol{d}_f\| \Big)^2 \right) \\ & \leq \min_{\boldsymbol{z}^t + \boldsymbol{d} \in \mathcal{M}} H(\boldsymbol{z}^t + \boldsymbol{d}) - H(\boldsymbol{z}^t) + \frac{Q_{\max}}{2} \Big(\sum_{f\in\mathcal{F}} \|\boldsymbol{d}_f\| \Big)^2 \\ & \leq \min_{\beta\in[0,1]} H(\boldsymbol{z}^t + \beta(\bar{\boldsymbol{z}}^t - \boldsymbol{z}^t)) - H(\boldsymbol{z}^t) \\ & \quad + \frac{Q_{\max}\beta^2}{2} \Big(\sum_{f\in\mathcal{F}} \|\bar{\boldsymbol{z}}^t_f - \boldsymbol{z}^t_f\| \Big)^2 \\ & \leq \min_{\beta\in[0,1]} \beta(H(\bar{\boldsymbol{z}}^t) - H(\boldsymbol{z}^t)) + \frac{Q_{\max}\beta^2}{2}\|\bar{\boldsymbol{z}}^t - \boldsymbol{z}^t\|^2_{2,1} \end{aligned} \tag{37}$$

where $\bar{\boldsymbol{z}}^t = \Pi_{\mathcal{S}}(\boldsymbol{z}^t)$ is the projection of $\boldsymbol{z}^t$ to the optimal solution set $\mathcal{S}$. The second and last inequality is due to convexity, and the third inequality is due to a confinement of optimization domain. Then let L_g be the local Lipschitz-continuous constant of function $G(M\boldsymbol{z}) = \frac{\rho}{2}\|M\boldsymbol{z}\|^2$ in the bounded domain of $M\boldsymbol{z}$. We discuss two cases in the following.

Case 1: $4L_g^2\|M\boldsymbol{z}^t - \boldsymbol{t}^*\|^2 < (\boldsymbol{\Delta}^T\boldsymbol{z}^t - s^*)^2$.

In this case, we have

$$\begin{aligned} \|\boldsymbol{z}^t - \bar{\boldsymbol{z}}^t\|^2_{2,1} &\leq \theta_1(\|M\boldsymbol{z}^t - \boldsymbol{t}^*\|^2 + (\boldsymbol{\Delta}^T\boldsymbol{z}^s - s^*)^2) \\ &\leq \theta_1(\frac{1}{L_g^2} + 1)(\boldsymbol{\Delta}^T\boldsymbol{z}^t - s^*)^2 \\ &\leq 2\theta_1(\boldsymbol{\Delta}^T\boldsymbol{z}^t - s^*)^2, \end{aligned} \tag{38}$$

and

$$|\boldsymbol{\Delta}^T\boldsymbol{z}^t - s^*| \geq 2L_g\|M\boldsymbol{z}^t - \boldsymbol{t}^*\| \geq 2|G(M\boldsymbol{z}^t) - G(\boldsymbol{t}^*)|$$

by the definition of Lipschitz constant L_g. Note $\boldsymbol{\Delta}^T\boldsymbol{z}^t - s^*$ is non-negative since otherwise, $H(\boldsymbol{z}^t) - H^* = G(M\boldsymbol{z}^t) - G(\boldsymbol{t}^*) + (\boldsymbol{\Delta}^T\boldsymbol{z}^t - s^*) \leq |G(M\boldsymbol{z}^t) - G(\boldsymbol{t}^*)| - |\boldsymbol{\Delta}^T\boldsymbol{z}^t - s^*| \leq -\frac{1}{2}|\boldsymbol{\Delta}^T\boldsymbol{z}^t - s^*| < 0$, which leads to contradiction. Therefore, we have

$$\begin{aligned} & H(\boldsymbol{z}^t) - H^* \\ & = G(M\boldsymbol{z}^t) - G(\boldsymbol{t}^*) + (\boldsymbol{\Delta}^T\boldsymbol{z}^t - s^*) \\ & \geq -|G(M\boldsymbol{z}^t) - G(\boldsymbol{t}^*)| + (\boldsymbol{\Delta}^T\boldsymbol{z}^t - s^*) \\ & \geq \frac{1}{2}(\boldsymbol{\Delta}^T\boldsymbol{z}^t - s^*). \end{aligned} \tag{39}$$

Combining (37), (38) and (39), we have

$$\begin{aligned} & H(\boldsymbol{z}^{t+1}) - H(\boldsymbol{z}^t) \\ & \leq \min_{\beta \in [0,1]} -\frac{\beta}{2}(\boldsymbol{\Delta}^T \boldsymbol{z}^t - s^*) + \frac{2Q_{max}\theta_1\beta^2}{2}(\boldsymbol{\Delta}^T \boldsymbol{z}^t - s^*)^2 \\ & = \begin{cases} -1/(16Q_{\max}\theta_1) & , 1/(4\rho\theta_1(\boldsymbol{\Delta}^T \boldsymbol{z}^t - s^*)) \leq 1 \\ -\frac{1}{4}(\boldsymbol{\Delta}^T \boldsymbol{\alpha}^s - s^*) & , o.w. \end{cases} \end{aligned}$$

Furthermore, we have

$$-\frac{1}{16Q_{max}\theta_1} \leq -\frac{1}{16Q_{max}\theta_1(H^0 - H^*)}\left(H(\boldsymbol{z}^t) - H^*\right)$$

where $H^0 = H(\boldsymbol{z}^0)$, and

$$-\frac{1}{4}(\boldsymbol{\Delta}^T \boldsymbol{z}^t - s^*) \leq -\frac{1}{6}(H(\boldsymbol{z}^t) - H^*)$$

since $H(\boldsymbol{z}^t) - H^* \leq |G(M\boldsymbol{z}^t) - G(\boldsymbol{t}^*)| + \boldsymbol{\Delta}^T \boldsymbol{z}^t - s^* \leq \frac{3}{2}(\boldsymbol{\Delta}^T \boldsymbol{z}^t - s^*)$. In summary, for Case 1 we obtain

$$H(\boldsymbol{z}^{t+1})] - H^* \leq (1 - \frac{m_0}{Q_{max}})\left(H(\boldsymbol{z}^t) - H^*\right) \tag{40}$$

where

$$m_0 = \frac{1}{\max\{16\theta_1(H^0 - H^*)\ ,\ 6\}}. \tag{41}$$

Case 2: $4L_g^2\|M\boldsymbol{z}^t - \boldsymbol{t}^*\|^2 \geq (\boldsymbol{\Delta}^T \boldsymbol{z}^t - s^*)^2$.

In this case, we have

$$\|\bar{\boldsymbol{z}}^t - \boldsymbol{z}^t\|^2 \leq \theta_1\left(1 + 4L_g^2\right)\|M\boldsymbol{z}^t - \boldsymbol{t}^*\|^2, \tag{42}$$

and by strong convexity of $G(.)$,

$$\begin{aligned} & H(\boldsymbol{z}^t) - H^* \geq \\ & \boldsymbol{\Delta}^T(\boldsymbol{z}^t - \boldsymbol{z}^*) + \nabla G(t^*)^T M(\bar{\boldsymbol{z}}^t - \boldsymbol{z}^t) + \frac{\rho}{2}\|M\boldsymbol{z}^t - \boldsymbol{t}^*\|^2. \end{aligned}$$

Now let $h(\boldsymbol{\alpha})$ be a function that takes value 0 when $\boldsymbol{z}$ is feasible and takes value ∞ otherwise. Adding inequality $0 = h(\boldsymbol{z}^t) - h(\bar{\boldsymbol{z}}^t) \geq \langle\boldsymbol{\sigma}^*, \boldsymbol{z}^t - \bar{\boldsymbol{z}}^t\rangle$ for some $\boldsymbol{\sigma}^* \in \partial h(\bar{\boldsymbol{z}}^t)$ to the above gives

$$H(\boldsymbol{z}^t) - H^* \geq \frac{\rho}{2}\|M\boldsymbol{z}^t - \boldsymbol{t}^*\|^2 \tag{43}$$

since $\boldsymbol{\sigma}^* + \boldsymbol{\Delta} + \nabla G(\boldsymbol{t}^*)^T M = \boldsymbol{\sigma}^* + \nabla H(\boldsymbol{z}^t) = 0$. Combining (37), (42), and (43), we obtain

$$\begin{aligned} & H(\boldsymbol{z}^{t+1}) - H(\boldsymbol{z}^t) \\ & \leq \min_{\beta \in [0,1]} -\beta(H(\boldsymbol{z}^t) - H^*) + \frac{\theta_1(1 + 4L_g^2)Q_{max}\beta^2}{2\rho}\left(H(\boldsymbol{z}^t) - H^*\right) \\ & = -\frac{\rho}{2\theta_1(1 + 4L_g^2)Q_{max}}\left(H(\boldsymbol{z}^t) - H^*\right) \end{aligned} \tag{44}$$

Combining results of Case 1 (40) and Case 2 (44), we have

$$H(\boldsymbol{z}^{t+1}) - H(\boldsymbol{z}^t) \leq -\frac{m_1}{Q_{\max}}(H(\boldsymbol{z}^t) - H^*), \tag{45}$$

where

$$m_1 = \frac{1}{\max\{16\theta_1\Delta\mathcal{L}^0, 2\theta_1(1 + 4L_g^2)/\rho, 6\}}$$

This leads to the conclusion.

□

9 Active set size statistics for all experiments

Dataset	$\|\mathcal{F}\|$	$\mathbb{E}_t\|\mathcal{A}_{\mathcal{F}}^t\|$
MultiLabel	7544670	6128.2
Dataset	$\|\mathcal{Y}_f\|$	$\mathbb{E}_{t,f}\|\mathcal{A}_f^t\|$
Segmentation	441	4.9
ImageAlignment	6889	2.4
Protein	163216	12.7
GraphMatching	1069156	1.66

Table 3: Run time statistics for GDMM active set. For multilabel dataset, we use Algorithm 2, thus $|\mathcal{F}|$ and $\mathbb{E}_t|\mathcal{A}_{\mathcal{F}}^t|$ are compared, where $\mathbb{E}_t|\mathcal{A}_{\mathcal{F}}^t|$ is the expected size of $\mathcal{A}_{\mathcal{F}}$ over all iterations. For other datasets, we use Algorithm 1, thus $|\mathcal{Y}_f|$ and $\mathbb{E}_{t,f}|\mathcal{A}_f^t|$ are compared, the latter is the expected size of $\mathcal{A}_f$ over all iterations and bigram factors.