An Alternating Direction Method for Dual MAP LP Relaxation

Ofer Meshi and Amir Globerson

The School of Computer Science and Engineering,
The Hebrew University of Jerusalem, Jerusalem, Israel
{meshi,gamir}@cs.huji.ac.il

Abstract. Maximum a-posteriori (MAP) estimation is an important task in many applications of probabilistic graphical models. Although finding an exact solution is generally intractable, approximations based on linear programming (LP) relaxation often provide good approximate solutions. In this paper we present an algorithm for solving the LP relaxation optimization problem. In order to overcome the lack of strict convexity, we apply an augmented Lagrangian method to the dual LP. The algorithm, based on the alternating direction method of multipliers (ADMM), is guaranteed to converge to the global optimum of the LP relaxation objective. Our experimental results show that this algorithm is competitive with other state-of-the-art algorithms for approximate MAP estimation.

Keywords: Graphical Models, Maximum a-posteriori, Approximate Inference, LP Relaxation, Augmented Lagrangian Methods.

1   Introduction

Graphical models are widely used to describe multivariate statistics for discrete variables, and have found widespread applications in numerous domains. One of the basic inference tasks in such models is to find the *maximum a-posteriori* (MAP) assignment. Unfortunately, this is typically a hard computational problem which cannot be solved exactly for many problems of interest. It has turned out that *linear programming* (LP) relaxations provide effective approximations to the MAP problem in many cases (e.g., see [15, 21, 24]).

Despite the theoretical computational tractability of MAP-LP relaxations, solving them in practice is a challenge for real world problems. Using off-the-shelf LP solvers is typically inadequate for large models since the resulting LPs have too many constraints and variables [29]. This has led researchers to seek optimization algorithms that are tailored to the specific structure of the MAP-LP [7, 13, 14, 16, 20, 28]. The advantage of such methods is that they work with very simple local updates and are therefore easy to implement in the large scale setting.

The suggested algorithms fall into several classes, depending on their approach to the problem. The TRW-S [14], MSD [28] and MPLP [7] algorithms employ

coordinate descent in the dual of the LP. While these methods typically show good empirical behavior, they are not guaranteed to reach the global optimum of the LP relaxation. This is a result of non strict-convexity of the dual LP and the fact that block coordinate descent might get stuck in suboptimal points under these conditions. One way to avoid this problem is to use a *soft-max* function which is smooth and strictly convex, hence this results in globally convergent algorithms [6, 10, 12]. Another class of algorithms [13, 16] uses the same dual objective, but employs variants of subgradient descent to it. While these methods are guaranteed to converge globally, they are typically slower in practice than the coordinate descent ones (e.g., see [13] for a comparison). Finally, there are also algorithms that optimize the primal LP directly. One example is the proximal point method of Ravikumar et al. [20]. While also globally convergent, it has the disadvantage of using a double loop scheme where every update involves an iterative algorithm for projecting onto the local polytope.

More recently, Martins et al. [17] proposed a globally convergent algorithm for MAP-LP based on the *alternating direction method of multipliers* (ADMM) [8, 5, 4, 2]. This method proceeds by iteratively updating primal and dual variables in order to find a saddle point of an *augmented Lagrangian* for the problem. They suggest to use an augmented Lagrangian of the *primal* MAP-LP problem. However, their formulation is restricted to binary pairwise factors and several specific global factors. In this work, we propose an algorithm that is based on the same key idea of ADMM, however it stems from augmenting the Lagrangian of the *dual* MAP-LP problem instead. An important advantage of our approach is that the resulting algorithm can be applied to models with *general local factors* (non-pairwise, non-binary). We also show that in practice our algorithm converges much faster than the primal ADMM algorithm and that it compares favorably with other state-of-the-art methods for MAP-LP optimization.

2   MAP and LP Relaxation

Markov Random Fields (MRFs) are probabilistic graphical models that encode the joint distribution of a set of discrete random variables $\mathcal{X} = \{X_1, ..., X_n\}$. The joint probability is defined by combining a set C of local functions $\theta_c(x_c)$, termed *factors*. The factors depend only on (small) subsets of the variables ($X_c \subseteq \mathcal{X}$) and model the direct interactions between them (to simplify notation we drop the variable name in $X_c = x_c$; see [27]). The joint distribution is then given by: $P(x) \propto \exp\left(\sum_i \theta_i(x_i) + \sum_{c \in C} \theta_c(x_c)\right)$, where we have included also singleton factors over individual variables [27]. In many applications of MRFs we are interested in finding the maximum probability assignment (MAP assignment). This yields the optimization problem:

$$\arg\max_x \sum_i \theta_i(x_i) + \sum_{c \in C} \theta_c(x_c)$$

Due to its combinatorial nature, this problem is NP-hard for general graphical models, and tractable only in isolated cases such as tree structured graphs. This has motivated research on approximation algorithms.

One of the most successful approximation schemes has been to use LP relaxations of the MAP problem. In this approach the original combinatorial problem is posed as a LP and then some of the constraints are relaxed to obtain a tractable LP problem that approximates the original one. In our case, the resulting MAP-LP relaxation problem is:

$$\max_{\mu \in L(G)} \sum_i \sum_{x_i} \mu_i(x_i)\theta_i(x_i) + \sum_c \sum_{x_c} \mu_c(x_c)\theta_c(x_c) \tag{1}$$

where μ are auxiliary variables that correspond to (pseudo) marginal distributions, and $L(G)$ is the reduced set of constraints called the *local polytope* [27], defined by:

$$L(G) = \left\{ \mu \geq 0 \left| \begin{array}{ll} \sum_{x_{c\backslash i}} \mu_c(x_{c\backslash i}, x_i) = \mu_i(x_i) \ \forall c, i : i \in c, x_i \\ \sum_{x_i} \mu_i(x_i) = 1 & \forall i \end{array} \right. \right\}$$

In this paper we use the dual problem of Eq. (1), which takes the form:

$$\min_\delta \sum_i \max_{x_i} \left(\theta_i(x_i) + \sum_{c:i\in c} \delta_{ci}(x_i) \right) + \sum_c \max_{x_c} \left(\theta_c(x_c) - \sum_{i:i\in c} \delta_{ci}(x_i) \right) \tag{2}$$

where δ are dual variables corresponding to the marginalization constraints in $L(G)$ (see [22, 28, 23]).[1] This formulation offers several advantages. First, it minimizes an upper bound on the true MAP value. Second, it provides an optimality certificate through the duality gap w.r.t. a decoded primal solution [23]. Third, the resulting problem is unconstrained, which facilitates its optimization. Indeed, several algorithms have been proposed for optimizing this dual problem. The two main approaches are block coordinate descent [14, 28, 7] and subgradient descent [16], each with its advantages and disadvantages. In particular, coordinate descent algorithms are typically much faster at minimizing the dual, while the subgradient method is guaranteed to converge to the global optimum (see [23] for in-depth discussion).

Recently, Jojic et al. [13] presented an accelerated dual decomposition algorithm which stems from adding strongly convex smoothing terms to the subproblems in the dual function Eq. (2). Their method achieves a better convergence rate over the standard subgradient method ($O\left(\frac{1}{\epsilon}\right)$ vs. $O\left(\frac{1}{\epsilon^2}\right)$). An alternative approach, that is also globally convergent, has been recently suggested by Martins et al. [17]. Their approach is based on an augmented Lagrangian method, which we next discuss.

3   The Alternating Direction Method of Multipliers

We now briefly review ADMM for convex optimization [8, 5, 4, 2].

---

[1] An equivalent optimization problem can be derived via a dual decomposition approach [23].

Consider the following optimization problem:

$$\text{minimize } f(x) + g(z) \quad \text{s.t. } Ax = z \tag{3}$$

where f and g are convex functions. The ADMM approach begins by adding the function $\frac{\rho}{2} \|Ax - z\|^2$ to the above objective, where $\rho > 0$ is a penalty parameter. This results in the optimization problem:

$$\text{minimize } f(x) + g(z) + \frac{\rho}{2} \|Ax - z\|^2 \quad \text{s.t. } Ax = z \tag{4}$$

Clearly the above has the same optimum as Eq. (3) since when the constraints $Ax = z$ are satisfied, the added quadratic term equals zero. The Lagrangian of the augmented problem Eq. (4) is given by:

$$\mathcal{L}_\rho(x, z, \nu) = f(x) + g(z) + \nu^\top(Ax - z) + \frac{\rho}{2} \|Ax - z\|^2 \tag{5}$$

where ν is a vector of Lagrange multipliers. The solution to the problem of Eq. (4) is given by $\max_\nu \min_{x,z} \mathcal{L}_\rho(x, z, \nu)$. The ADMM method provides an elegant algorithm for finding this saddle point. The idea is to combine subgradient descent over ν with coordinate descent over the x and z variables. The method applies the following iterations:

$$x^{t+1} = \arg\min_x \mathcal{L}_\rho(x, z^t, \nu^t)$$
$$z^{t+1} = \arg\min_z \mathcal{L}_\rho(x^{t+1}, z, \nu^t)$$
$$\nu^{t+1} = \nu^t + \rho\left(Ax^{t+1} - z^{t+1}\right) \tag{6}$$

The algorithm consists of primal and dual updates, where the primal update is executed sequentially, minimizing first over x and then over z. This split retains the decomposition of the objective that has been lost due to the addition of the quadratic term.

The algorithm is run either until the number of iterations exceeds a predefined limit, or until some termination criterion is met. A commonly used such stopping criterion is: $\|Ax - z\|^2 \leq \epsilon$ and $\|z^{t+1} - z^t\|^2 \leq \epsilon$. These two conditions can serve to bound the suboptimality of the solution.

The ADMM algorithm is guaranteed to converge to the global optimum of Eq. (3) under rather mild conditions [2]. However, in terms of convergence rate, the worst case complexity of ADMM is $O(\frac{1}{\epsilon^2})$. Despite this potential caveat, ADMM has been shown to work well in practice (e.g., [1, 26]). Recently, accelerated variants on the basic alternating direction method have been proposed [9]. These faster algorithms are based on linearization and come with improved convergence rate of $O(\frac{1}{\epsilon})$, achieving the theoretical lower bound for first-order methods [19]. In this paper we focus on the basic ADMM formulation and leave derivation of accelerated variants to future work.

4   The Augmented Dual LP Algorithm

In this section we derive our algorithm by applying ADMM to the dual MAP-LP problem of Eq. (2). The challenge is to design the constraints in a way that facilitates efficient closed-form solutions for all updates.

To this end, we duplicate the dual variables δ and denote the second copy by $\bar{\delta}$. We then introduce additional variables λ_c corresponding to the summation of δ's pertaining to factor c. These agreement constraints are enforced through $\bar{\delta}$, and thus we have a constraint $\delta_{ci}(x_i) = \bar{\delta}_{ci}(x_i)$ for all $c, i : i \in c, x_i$, and $\lambda_c(x_c) = \sum_{i:i\in c} \bar{\delta}_{ci}(x_i)$ for all c, x_c.

Following the ADMM framework, we add quadratic terms and obtain the augmented Lagrangian for the dual MAP-LP problem of Eq. (2):

$$\mathcal{L}_\rho(\delta, \lambda, \bar{\delta}, \gamma, \mu) =$$

$$\sum_i \max_{x_i} \left(\theta_i(x_i) + \sum_{c:i\in c} \delta_{ci}(x_i) \right) + \sum_c \max_{x_c} \left(\theta_c(x_c) - \lambda_c(x_c) \right)$$

$$+ \sum_c \sum_{i:i\in c} \sum_{x_i} \gamma_{ci}(x_i) \left(\delta_{ci}(x_i) - \bar{\delta}_{ci}(x_i) \right) + \frac{\rho}{2} \sum_c \sum_{i:i\in c} \sum_{x_i} \left(\delta_{ci}(x_i) - \bar{\delta}_{ci}(x_i) \right)^2$$

$$+ \sum_c \sum_{x_c} \mu_c(x_c) \left(\lambda_c(x_c) - \sum_{i:i\in c} \bar{\delta}_{ci}(x_i) \right) + \frac{\rho}{2} \sum_c \sum_{x_c} \left(\lambda_c(x_c) - \sum_{i:i\in c} \bar{\delta}_{ci}(x_i) \right)^2$$

To see the relation of this formulation to Eq. (5), notice that (δ, λ) subsume the role of x, $\bar{\delta}$ subsumes the role of z (with $g(z) = 0$), and the multipliers (γ, μ) correspond to ν.

The updates of our algorithm, which stem from Eq. (6), are summarized in Alg. 1 (a detailed derivation appears in Appendix A). In Alg. 1 we define $N(i) = \{c : i \in c\}$, and the subroutine $w = \mathrm{TRIM}(v, d)$ that serves to clip the values in the vector v at some threshold t (i.e., $w_i = \min\{v_i, t\}$) such that the sum of removed parts equals $d > 0$ (i.e., $\sum_i v_i - w_i = d$). This can be carried out efficiently in linear time (in expectation) by partitioning [3].

Notice that all updates can be computed efficiently so the cost of each iteration is similar to that of message passing algorithms like MPLP [7] or MSD [28], and to that of dual decomposition [13, 16]. Furthermore, significant speedup is attained by caching some results for future iterations. In particular, the threshold in the TRIM subroutine (the new maximum) can serve as a good initial guess at the next iteration, especially at later iterations where the change in variable values is quite small. Finally, many of the updates can be executed in parallel. In particular, the δ update can be carried out simultaneously for all variables i, and likewise all factors c can be updated simultaneously in the λ and $\bar{\delta}$ updates. In addition, δ and λ can be optimized independently, since they appear in different parts of the objective. This may result in considerable reduction in runtime when executed on parallel architecture.[2]

---

[2] In our experiments we used sequential updates.

Algorithm 1. The Augmented Dual LP Algorithm (ADLP)

---

for $t = 1$ to T **do**

    Update δ: for all $i = 1, ..., n$

    Set $\bar{\theta}_i = \theta_i + \sum_{c:i \in c}(\bar{\delta}_{ci} - \frac{1}{\rho}\gamma_{ci})$

    $\bar{\theta}'_i = \text{TRIM}(\bar{\theta}_i, \frac{|N(i)|}{\rho})$

    $q = (\bar{\theta}_i - \bar{\theta}'_i)/|N(i)|$

    Update $\delta_{ci} = \bar{\delta}_{ci} - \frac{1}{\rho}\gamma_{ci} - q \quad \forall c : i \in c$

    Update λ: for all $c \in C$

    Set $\bar{\theta}_c = \theta_c - \sum_{i:i \in c}\bar{\delta}_{ci} + \frac{1}{\rho}\mu_c$

    $\bar{\theta}'_c = \text{TRIM}(\bar{\theta}_c, \frac{1}{\rho})$

    Update $\lambda_c = \theta_c - \bar{\theta}'_c$

    Update $\bar{\delta}$: for all $c \in C, i : i \in c, x_i$

    Set $v_{ci}(x_i) = \delta_{ci}(x_i) + \frac{1}{\rho}\gamma_{ci}(x_i) + \sum_{x_{c \setminus i}} \lambda_c(x_{c \setminus i}, x_i) + \frac{1}{\rho}\sum_{x_{c \setminus i}} \mu_c(x_{c \setminus i}, x_i)$

    $\bar{v}_c = \frac{1}{1 + \sum_{k:k \in c}|X_{c \setminus k}|}\sum_{k:k \in c}|X_{c \setminus k}|\sum_{x_k} v_{ck}(x_k)$

    Update $\bar{\delta}_{ci}(x_i) = \frac{1}{1 + |X_{c \setminus i}|}\left[v_{ci}(x_i) - \sum_{j:j \in c, j \neq i}|X_{c \setminus \{i,j\}}|\left(\sum_{x_j} v_{cj}(x_j) - \bar{v}_c\right)\right]$

    Update the multipliers:

    $\gamma_{ci}(x_i) \leftarrow \gamma_{ci}(x_i) + \rho\left(\delta_{ci}(x_i) - \bar{\delta}_{ci}(x_i)\right) \qquad$ for all $c \in C, i : i \in c, x_i$

    $\mu_c(x_c) \leftarrow \mu_c(x_c) + \rho\left(\lambda_c(x_c) - \sum_{i:i \in c}\bar{\delta}_{ci}(x_i)\right)$ for all $c \in C, x_c$

end for

---

5 Experimental Results

To evaluate our augmented dual LP (ADLP) algorithm (Alg. 1) we compare it to two other algorithms for finding an approximate MAP solution. The first is MPLP of Globerson and Jaakkola [7], which minimizes the dual LP of Eq. (2) via block coordinate descent steps (cast as message passing). The second is the accelerated dual decomposition (ADD) algorithm of Jojic et al. [13].[3] We conduct experiments on protein design problems from the dataset of Yanover et al. [29]. In these problems we are given a 3D structure and the goal is to find a sequence of amino-acids that is the most stable for that structure. The problems are modeled by singleton and pairwise factors and can be posed as finding a MAP assignment for the given model. This is a demanding setting in which each problem may have hundreds of variables with 100 possible states on average [29, 24].

    Figure 1 shows two typical examples of protein design problems. It plots the objective of Eq. (2) (computed using δ variables only) as a function of the execution time for all algorithms. First, in Figure 1 (left) we observe that the co-ordinate descent algorithm (MPLP) converges faster than the other algorithms,

---

[3] For both algorithms we used the same `C++` implementation used by Jojic et al. [13], available at `http://ai.stanford.edu/~sgould/svl`. Our own algorithm was implemented as an extension of their package.
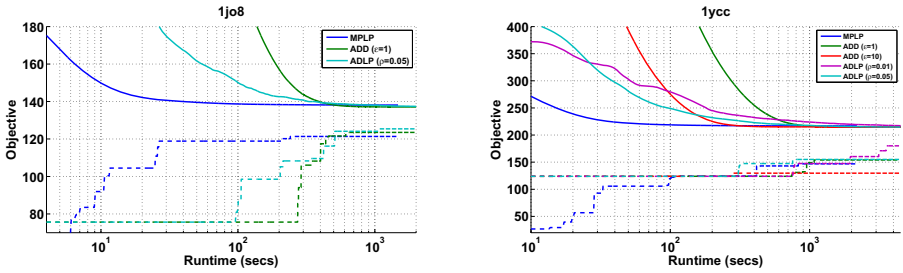
Fig. 1. Comparison of three algorithms for approximate MAP estimation: our augmented dual LP algorithm (ADLP), accelerated dual decomposition algorithm (ADD) by Jojic et al. [13], and the dual coordinate descent MPLP algorithm [7]. The figure shows two examples of protein design problems, for each the dual objective of Eq. (2) is plotted as a function of execution time. Dashed lines denote the value of the best decoded primal solution.

however it tends to stop prematurely and yield suboptimal solutions. In contrast, ADD and ADLP take longer to converge but achieve the globally optimal solution to the approximate objective. Second, it can be seen that the convergence times of ADD and ADLP are very close, with a slight advantage to ADD. The dashed lines in Figure 1 show the value of the decoded primal solution (assignment) [23]. We see that there is generally a correlation between the quality of the dual objective and the decoded primal solution, namely the decoded primal solution improves as the dual solution approaches optimality. Nevertheless, we note that there is no dominant algorithm in terms of decoding (here we show examples where our decoding is superior). In many cases MPLP yields better decoded solutions despite being suboptimal in terms of the dual objective (not shown; this is also noted in [13]).

We also conduct experiments to study the effect of the penalty parameter ρ. Our algorithm is guaranteed to globally converge for all $\rho > 0$, but its choice affects the actual rate of convergence. In Figure 1 (right) we compare two values of the penalty parameter $\rho = 0.01$ and $\rho = 0.05$. It shows that setting $\rho = 0.01$ results in somewhat slower convergence to the optimum, however in this case the final primal solution (dashed line) is better than that of the other algorithms. In practice, in order to choose an appropriate ρ, one can run a few iterations of ADLP with several values and see which one achieves the best objective [17]. We mention in passing that ADD employs an accuracy parameter ϵ which determines the desired suboptimality of the final solution [13]. Setting ϵ to a large value results in faster convergence to a lower accuracy solution. On the one hand, this trade-off can be viewed as a merit of ADD, which allows to obtain coarser approximations at reduced cost. On the other hand, an advantage of our method is that the choice of penalty ρ affects only the rate of convergence and does not impose additional reduction in solution accuracy over that of the LP relaxation. In Figure 1 (left) we use $\epsilon = 1$, as in Jojic et al., while in Figure 1 (right) we compare two values $\epsilon = 1$ and $\epsilon = 10$ to demonstrate the effect of this accuracy parameter.
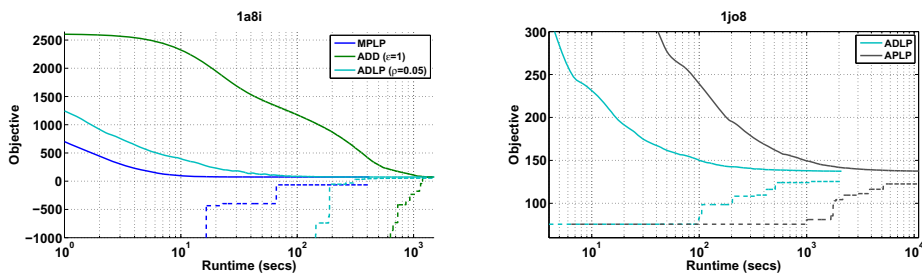
Fig. 2. (Left) Comparison for a side chain prediction problem similar to Figure 1 (left). (Right) Comparison of our augmented dual LP algorithm (ADLP) and a generalized variant (APLP) of the ADMM algorithm by Martins et al. [17] on a protein design problem. The dual objective of Eq. (2) is plotted as a function of execution time. Dashed lines denote the value of the best decoded primal solution.

We next compare performance of the algorithms on a side chain prediction problem [29]. This problem is the inverse of the protein design problem, and involves finding the 3D configuration of rotamers given the backbone structure of a protein. Figure 2 (left) shows a comparison of MPLP, ADD and ADLP on one of the largest proteins in the dataset (812 variables with 12 states on average). As in the protein design problems, MPLP converges fast to a suboptimal solution. We observe that here ADLP converges somewhat faster than ADD, possibly because the smaller state space results in faster ADLP updates.

As noted earlier, Martins et al. [17] recently presented an approach that applies ADMM to the primal LP (i.e., Eq. (1)). Although their method is limited to binary pairwise factors (and several global factors), it can be modified to handle non-binary higher-order factors, as the derivation in Appendix B shows. We denote this variant by APLP. As in ADLP, in the APLP algorithm all updates are computed analytically and executed efficiently. Figure 2 (right) shows a comparison of ADLP and APLP on a protein design problem. It illustrates that ADLP converges significantly faster than APLP (similar results, not shown here, are obtained for the other proteins).

6   Discussion

Approximate MAP inference methods based on LP relaxation have drawn much attention lately due to their practical success and attractive properties. In this paper we presented a novel globally convergent algorithm for approximate MAP estimation via LP relaxation. Our algorithm is based on the augmented Lagrangian method for convex optimization, which overcomes the lack of strict convexity by adding a quadratic term to smooth the objective. Importantly, our algorithm proceeds by applying simple to implement closed-form updates, and it is highly scalable and parallelizable. We have shown empirically that our algorithm compares favorably with other state-of-the-art algorithms for approximate MAP estimation in terms of accuracy and convergence time.

Several existing globally convergent algorithms for MAP-LP relaxation rely on adding local entropy terms in order to smooth the objective [6, 10, 12, 13]. Those methods must specify a temperature control parameter which affects the quality of the solution. Specifically, solving the optimization subproblems at high temperature reduces solution accuracy, while solving them at low temperature might raise numerical issues. In contrast, our algorithm is quite insensitive to the choice of such control parameters. In fact, the penalty parameter ρ affects the rate of convergence but not the accuracy or numerical stability of the algorithm. Moreover, despite lack of fast convergence rate guarantees, in practice the algorithm has similar or better convergence times compared to other globally convergent methods in various settings. Note that [17] also show an advantage of their primal based ADMM method over several baselines.

Several improvements over our basic algorithm can be considered. One such improvement is to use smart initialization of the variables. For example, since MPLP achieves larger decrease in objective at early iterations, it is possible to run it for a limited number of steps and then take the resulting variables δ for the initialization of ADLP. Notice, however, that for this scheme to work well, the Lagrange multipliers γ and μ should be also initialized accordingly. Another potential improvement is to use an adaptive penalty parameter ρ_t (e.g., [11]). This may improve convergence in practice, as well as reduce sensitivity to the initial choice of ρ. On the downside, the theoretical convergence guarantees of ADMM no longer hold in this case. Martins et al. [17] show that the ADMM framework is also suitable for handling certain types of global factors, which include a large number of variables in their scope (e.g., XOR factor). Using an appropriate formulation, it is possible to incorporate such factors in our dual LP framework as well.[4] Finally, it is likely that our method can be further improved by using recently introduced accelerated variants of ADMM [9]. Since these variants achieve asymptotically better convergence rate, the application of such methods to MAP-LP similar to the one we presented here will likely result in faster algorithms for approximate MAP estimation.

In this paper, we assumed that the model parameters were given. However, in many cases one wishes to learn these from data, for example by minimizing a prediction loss (e.g., hinge loss [25]). We have recently shown how to incorporate dual relaxation algorithms into such learning problems [18]. It will be interesting to apply our ADMM approach in this setting to yield an efficient learning algorithm for structured prediction problems.

---

[4] The auxiliary variables λ_c are not used in this case.

A   Derivation of Augmented Dual LP Algorithm

In this section we derive the ADMM updates for the augmented Lagrangian of
the dual MAP-LP which we restate here for convenience:

$$\mathcal{L}_\rho(\delta, \lambda, \bar{\delta}, \gamma, \mu) =$$

$$\sum_i \max_{x_i} \left(\theta_i(x_i) + \sum_{c:i\in c} \delta_{ci}(x_i) \right) + \sum_c \max_{x_c} \left(\theta_c(x_c) - \lambda_c(x_c) \right)$$

$$+ \sum_c \sum_{i:i\in c} \sum_{x_i} \gamma_{ci}(x_i) \left(\delta_{ci}(x_i) - \bar{\delta}_{ci}(x_i) \right) + \frac{\rho}{2} \sum_c \sum_{i:i\in c} \sum_{x_i} \left(\delta_{ci}(x_i) - \bar{\delta}_{ci}(x_i) \right)^2$$

$$+ \sum_c \sum_{x_c} \mu_c(x_c) \left(\lambda_c(x_c) - \sum_{i:i\in c} \bar{\delta}_{ci}(x_i) \right) + \frac{\rho}{2} \sum_c \sum_{x_c} \left(\lambda_c(x_c) - \sum_{i:i\in c} \bar{\delta}_{ci}(x_i) \right)^2$$

Updates

- **The δ update:**
  For each variable $i = 1, ..., n$ consider a block δ_i which consists of δ_{ci} for all
  $c : i \in c$. For this block we need to minimize the following function:

$$\max_{x_i} \left(\theta_i(x_i) + \sum_{c:i\in c} \delta_{ci}(x_i) \right) + \sum_{c:i\in c} \sum_{x_i} \gamma_{ci}(x_i)\delta_{ci}(x_i) + \frac{\rho}{2} \sum_{c:i\in c} \sum_{x_i} \left(\delta_{ci}(x_i) - \bar{\delta}_{ci}(x_i) \right)^2$$

  Equivalently, this can be written more compactly in vector notation as:

$$\min_{\delta_i} \frac{1}{2} \|\delta_i\|^2 - (\bar{\delta}_i - \frac{1}{\rho}\gamma_i)^\top \delta_i + \frac{1}{\rho} \max_{x_i} (\theta_i(x_i) + \sum_{c:i\in c} \delta_{ci}(x_i))$$

  where $\bar{\delta}_i$ and γ_i are defined analogous to δ_i. The closed-form solution to
  this QP is given by the update in Alg. 1. It is obtained by inspecting KKT
  conditions and exploiting the structure of the summation inside the max (for
  a similar derivation see [3]).
- **The λ update:**
  For each factor $c \in C$ we seek to minimize the function:

$$\max_{x_c} (\theta_c(x_c) - \lambda_c(x_c)) + \sum_{x_c} \mu_c(x_c)\lambda_c(x_c) + \frac{\rho}{2} \sum_{x_c} \left(\lambda_c(x_c) - \sum_{i:i\in c} \bar{\delta}_{ci}(x_i) \right)^2$$

  In equivalent vector notation we have the problem:

$$\min_{\lambda_c} \frac{1}{2} \|\lambda_c\|^2 - \left(\sum_{i:i\in c} \bar{\delta}_{ci} - \frac{1}{\rho}\mu_c \right)^\top \lambda_c + \frac{1}{\rho} \max_{x_c}(\theta_c(x_c) - \lambda_c(x_c))$$

  This QP is very similar to that of the δ update and can be solved using the
  same technique. The resulting closed-form update is given in Alg. 1.

– **The $\bar{\delta}$ update:**

For each $c \in C$ we consider a block which consists of $\bar{\delta}_{ci}$ for all $i : i \in c$. We seek a minimizer of the function:

$$-\sum_{i:i\in c}\sum_{x_i}\gamma_{ci}(x_i)\bar{\delta}_{ci}(x_i) + \frac{\rho}{2}\sum_{i:i\in c}\sum_{x_i}\left(\delta_{ci}(x_i) - \bar{\delta}_{ci}(x_i)\right)^2$$

$$-\sum_{x_c}\mu_c(x_c)\sum_{i:i\in c}\bar{\delta}_{ci}(x_i) + \frac{\rho}{2}\sum_{x_c}\left(\lambda_c(x_c) - \sum_{i:i\in c}\bar{\delta}_{ci}(x_i)\right)^2$$

Taking partial derivative w.r.t. $\bar{\delta}_{ci}(x_i)$ and setting to 0 yields:

$$\bar{\delta}_{ci}(x_i) = \frac{1}{1 + |X_{c\setminus i}|}\left(v_{ci}(x_i) - \sum_{j:j\in c, j\neq i}|X_{c\setminus\{i,j\}}|\sum_{x_j}\bar{\delta}_{cj}(x_j)\right)$$

where: $v_{ci}(x_i) = \delta_{ci}(x_i) + \frac{1}{\rho}\gamma_{ci}(x_i) + \sum_{x_{c\setminus i}}\lambda_c(x_{c\setminus i}, x_i) + \frac{1}{\rho}\sum_{x_{c\setminus i}}\mu_c(x_{c\setminus i}, x_i)$. Summing this over x_i and $i : i \in c$ and plugging back in, we get the update in Alg. 1.

– Finally, the multipliers update is straightforward.

B   Derivation of Augmented Primal LP Algorithm

We next derive the algorithm for optimizing Eq. (1) with general local factors.

Consider the following formulation which is equivalent to the primal MAP-LP problem of Eq. (1). Define:

$$f_i(\mu_i) = \begin{cases} \sum_{x_i}\mu_i(x_i)\theta_i(x_i) & \mu_i(x_i) \geq 0 \text{ and } \sum_{x_i}\mu_i(x_i) = 1 \\ -\infty & \text{otherwise} \end{cases}$$

$$f_c(\mu_c) = \begin{cases} \sum_{x_c}\mu_c(x_c)\theta_c(x_c) & \mu_c(x_c) \geq 0 \text{ and } \sum_{x_c}\mu_c(x_c) = 1 \\ -\infty & \text{otherwise} \end{cases}$$

f accounts for the non-negativity and normalization constraints in $L(G)$. We add the marginalization constraints via copies of μ_c for each $i \in c$, denoted by $\bar{\mu}_{ci}$. Thus we get the augmented Lagrangian:

$$\mathcal{L}_\rho(\mu, \bar{\mu}, \delta, \beta) =$$

$$\sum_i f_i(\mu_i) + \sum_c f_c(\mu_c)$$

$$-\sum_c\sum_{i:i\in c}\sum_{x_i}\delta_{ci}(x_i)\left(\bar{\mu}_{ci}(x_i) - \mu_i(x_i)\right) - \frac{\rho}{2}\sum_c\sum_{i:i\in c}\sum_{x_i}\left(\bar{\mu}_{ci}(x_i) - \mu_i(x_i)\right)^2$$

$$-\sum_c\sum_{i:i\in c}\sum_{x_c}\beta_{ci}(x_c)\left(\bar{\mu}_{ci}(x_c) - \mu_c(x_c)\right) - \frac{\rho}{2}\sum_c\sum_{i:i\in c}\sum_{x_c}\left(\bar{\mu}_{ci}(x_c) - \mu_c(x_c)\right)^2$$

where $\bar{\mu}_{ci}(x_i) = \sum_{x_{c\backslash i}} \bar{\mu}_{ci}(x_{c\backslash i}, x_i)$.

To draw the connection with Eq. (5), in this formulation μ subsumes the role of x, $\bar{\mu}$ subsumes the role of z (with $g(z) = 0$), and the multipliers (δ, β) correspond to ν. We next show the updates which result from applying Eq. (6) to this formulation.

– **Update μ_i** for all $i = 1, ..., n$:

$$\mu_i \leftarrow \arg\max_{\mu_i \in \Delta_i} \mu_i^\top \left(\theta_i + \sum_{c:i\in c} (\delta_{ci} + \rho M_i \bar{\mu}_{ci}) \right) - \frac{1}{2}\mu_i^\top (\rho |N(i)| I) \mu_i$$

where $M_i \bar{\mu}_{ci} = \sum_{x_{c\backslash i}} \bar{\mu}_{ci}(x_{c\backslash i}, \cdot)$.

We have to maximize this QP under simplex constraints on μ_i. Notice that the objective matrix is diagonal, so this can be solved in closed form by shifting the target vector and then truncating at 0 such that the sum of positive elements equals 1 (see [3]). The solution can be computed in linear time (in expectation) by partitioning [3].

– **Update μ_c** for all $c \in C$:

$$\mu_c \leftarrow \arg\max_{\mu_c \in \Delta_c} \mu_c^\top \left(\theta_c + \sum_{i:i\in c} (\beta_{ci} + \rho \bar{\mu}_{ci}) \right) - \frac{1}{2}\mu_c^\top (\rho |N(c)| I) \mu_c$$

where $N(c) = \{i : i \in c\}$.

Again we have a projection onto the simplex with diagonal objective matrix, which can be done efficiently.

– **Update $\bar{\mu}_{ci}$** for all $c \in C, i : i \in c$:

$$\bar{\mu}_{ci} \leftarrow \arg\max_{\bar{\mu}_{ci}} \bar{\mu}_{ci}^\top \left(M_i^\top (\rho \mu_i - \delta_{ci}) - \beta_{ci} + \rho \mu_c \right) - \frac{\rho}{2} \bar{\mu}_{ci}^\top \left(M_i^\top M_i + I \right) \bar{\mu}_{ci}$$

Here we have an unconstrained QP, so the solution is obtained by $H^{-1}v$. Further notice that the inverse H^{-1} can be computed in closed form. To see how, $M_i^\top M_i$ is a block-diagonal matrix with blocks of ones with size $|X_i|$. Therefore, $H = \rho \left(M_i^\top M_i + I \right)$ is also block-diagonal. It follows that the inverse H^{-1} is a block-diagonal matrix where each block is the inverse of the corresponding block in H. Finally, it is easy to verify that the inverse of a block $\rho \left(1_{|X_i|} + I_{|X_i|} \right)$ is given by $\frac{1}{\rho} \left(I_{|X_i|} - \frac{1}{|X_i|+1} 1_{|X_i|} \right)$.

– **Update the multipliers**:

$$\delta_{ci}(x_i) \leftarrow \delta_{ci}(x_i) + \rho \left(\bar{\mu}_{ci}(x_i) - \mu_i(x_i) \right) \qquad \text{for all } c \in C, i : i \in c, x_i$$
$$\beta_{ci}(x_c) \leftarrow \beta_{ci}(x_c) + \rho \left(\bar{\mu}_{ci}(x_c) - \mu_c(x_c) \right) \qquad \text{for all } c \in C, i : i \in c, x_c$$

References

[1] Afonso, M., Bioucas-Dias, J., Figueiredo, M.: Fast image recovery using variable splitting and constrained optimization. IEEE Transactions on Image Processing 19(9), 2345–2356 (2010)

[2] Bertsekas, D.P., Tsitsiklis, J.N.: Parallel and distributed computation: numerical methods. Prentice-Hall, Inc., Upper Saddle River (2003)

[3] Duchi, J., Shalev-Shwartz, S., Singer, Y., Chandra, T.: Efficient projections onto the l1-ball for learning in high dimensions. In: Proceedings of the 25th International Conference on Machine learning, pp. 272–279 (2008)

[4] Eckstein, J., Bertsekas, D.P.: On the douglas-rachford splitting method and the proximal point algorithm for maximal monotone operators. Mathematical Programming 55, 293–318 (1992)

[5] Gabay, D., Mercier, B.: A dual algorithm for the solution of nonlinear variational problems via finite-element approximations. Computers and Mathematics with Applications 2, 17–40 (1976)

[6] Gimpel, K., Smith, N.A.: Softmax-margin crfs: training log-linear models with cost functions. In: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pp. 733–736 (2010)

[7] Globerson, A., Jaakkola, T.: Fixing max-product: Convergent message passing algorithms for MAP LP-relaxations. In: Advances in Neural Information Processing Systems, pp. 553–560 (2008)

[8] Glowinski, R., Marrocco, A.: Sur lapproximation, par elements finis dordre un, et la resolution, par penalisation-dualité, dune classe de problems de dirichlet non lineares. Revue Française d'Automatique, Informatique, et Recherche Opérationelle 9, 41–76 (1975)

[9] Goldfarb, D., Ma, S., Scheinberg, K.: Fast alternating linearization methods for minimizing the sum of two convex functions. Technical report, UCLA CAM (2010)

[10] Hazan, T., Shashua, A.: Norm-product belief propagation: Primal-dual message-passing for approximate inference. IEEE Transactions on Information Theory 56(12), 6294–6316 (2010)

[11] He, B.S., Yang, H., Wang, S.L.: Alternating direction method with self-adaptive penalty parameters for monotone variational inequalities. Journal of Optimization Theory and Applications 106, 337–356 (2000)

[12] Johnson, J.: Convex Relaxation Methods for Graphical Models: Lagrangian and Maximum Entropy Approaches. PhD thesis, EECS, MIT (2008)

[13] Jojic, V., Gould, S., Koller, D.: Fast and smooth: Accelerated dual decomposition for MAP inference. In: Proceedings of International Conference on Machine Learning (2010)

[14] Kolmogorov, V.: Convergent tree-reweighted message passing for energy minimization. IEEE Transactions on Pattern Analysis and Machine Intelligence 28(10), 1568–1583 (2006)

[15] Komodakis, N., Paragios, N.: Beyond loose LP-relaxations: Optimizing mRFs by repairing cycles. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part III. LNCS, vol. 5304, pp. 806–820. Springer, Heidelberg (2008)

[16] Komodakis, N., Paragios, N., Tziritas, G.: Mrf energy minimization and beyond via dual decomposition. IEEE Transactions on Pattern Analysis and Machine Intelligence 33, 531–552 (2011)

[17] Martins, A.F.T., Figueiredo, M.A.T., Aguiar, P.M.Q., Smith, N.A., Xing, E.P.: An augmented lagrangian approach to constrained map inference. In: International Conference on Machine Learning (June 2011)

[18] Meshi, O., Sontag, D., Jaakkola, T., Globerson, A.: Learning efficiently with approximate inference via dual losses. In: Proceedings of the 27th International Conference on Machine Learning, pp. 783–790 (2010)

[19] Nesterov, Y.: Smooth minimization of non-smooth functions. Mathematical Programming 103, 127–152 (2005)

[20] Ravikumar, P., Agarwal, A., Wainwright, M.: Message-passing for graph-structured linear programs: proximal projections, convergence and rounding schemes. In: Proc. of the 25th International Conference on Machine Learning, pp. 800–807 (2008)

[21] Rush, A.M., Sontag, D., Collins, M., Jaakkola, T.: On dual decomposition and linear programming relaxations for natural language processing. In: Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, EMNLP (2010)

[22] Schlesinger, M.I.: Syntactic analysis of two-dimensional visual signals in noisy conditions. Kibernetika 4, 113–130 (1976)

[23] Sontag, D., Globerson, A., Jaakkola, T.: Introduction to dual decomposition for inference. In: Sra, S., Nowozin, S., Wright, S.J. (eds.) Optimization for Machine Learning. MIT Press, Cambridge (2011)

[24] Sontag, D., Meltzer, T., Globerson, A., Jaakkola, T., Weiss, Y.: Tightening LP relaxations for MAP using message passing. In: Proc. of the 24th Annual Conference on Uncertainty in Artificial Intelligence, pp. 503–510 (2008)

[25] Taskar, B., Guestrin, C., Koller, D.: Max margin Markov networks. In: Thrun, S., Saul, L., Schölkopf, B. (eds.) Advances in Neural Information Processing Systems, vol. 16, pp. 25–32. MIT Press, Cambridge (2004)

[26] Tosserams, S., Etman, L., Papalambros, P., Rooda, J.: An augmented lagrangian relaxation for analytical target cascading using the alternating direction method of multipliers. Structural and Multidisciplinary Optimization 31, 176–189 (2006)

[27] Wainwright, M.J., Jordan, M.: Graphical models, exponential families, and variational inference. Foundations and Trends in Machine Learning 1(1-2), 1–305 (2008)

[28] Werner, T.: A linear programming approach to max-sum problem: A review. IEEE Transactions on Pattern Analysis and Machine Intelligence 29, 1165–1179 (2007)

[29] Yanover, C., Meltzer, T., Weiss, Y.: Linear programming relaxations and belief propagation – an empirical study. Journal of Machine Learning Research 7, 1887–1907 (2006)