HAL open science

Newton-type Methods for Inference in Higher-Order Markov Random Fields

Hariprasad Kannan, Nikos Komodakis, Nikos Paragios

To cite this version:

Hariprasad Kannan, Nikos Komodakis, Nikos Paragios. Newton-type Methods for Inference in Higher-Order Markov Random Fields. IEEE International Conference on Computer Vision and Pattern Recognition, Jul 2017, Honolulu, United States. pp.7224 - 7233. hal-01580862

HAL Id: hal-01580862

https://hal.science/hal-01580862v1

Submitted on 3 Sep 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Newton-type Methods for Inference in Higher-Order Markov Random Fields

Hariprasad Kannan
CentraleSupélec-INRIA Saclay
Université Paris-Saclay
hkannan@gmail.com

Nikos Komodakis
Ecole des Ponts ParisTech
Université Paris Est
nikos.komodakis@enpc.fr

Nikos Paragios
CentraleSupélec-INRIA Saclay
Université Paris-Saclay
nikos.paragios@ecp.fr

Abstract

Linear programming relaxations are central to MAP *inference in discrete Markov Random Fields. The ability to properly solve the Lagrangian dual is a critical component of such methods. In this paper, we study the benefit of using Newton-type methods to solve the Lagrangian dual of a smooth version of the problem. We investigate their ability to achieve superior convergence behavior and to better handle the ill-conditioned nature of the formulation, as compared to first order methods. We show that it is indeed possible to efficiently apply a trust region Newton method for a broad range of* MAP *inference problems. In this paper we propose a provably convergent and efficient framework that includes (i) excellent compromise between computational complexity and precision concerning the Hessian matrix construction, (ii) a damping strategy that aids efficient optimization, (iii) a truncation strategy coupled with a generic pre-conditioner for Conjugate Gradients, (iv) efficient sum-product computation for sparse clique potentials. Results for higher-order Markov Random Fields demonstrate the potential of this approach.*

1. Introduction

Many computer vision problems can be modelled using Markov Random Fields (MRFs). Maximum a posteriori (MAP) estimation in an MRF assigns labels to the nodes, in order to maximize their joint probability distribution. However, MAP inference is NP-hard for a general graph and several approaches exist to achieve approximate solutions - *graph cuts*, *belief propagation* and *LP relaxation* based methods [17]. In recent years, higher-order MRFs have achieved excellent results in various applications, since they model far-reaching interactions between the nodes. While the topic of inference in pairwise MRFs is well studied, development of scalable and efficient techniques for higher order models is evolving [15], [21], [18], [10], [20], [40].

The LP relaxation approach has led to state-of-the-art algorithms and also, provides a theoretical foundation to the topic of MAP inference. An attractive property of this approach is that it readily lends itself to inference in higher-order MRFs [21], [39]. Solving the LP relaxation in the primal is not scalable and the better approach is to solve the dual by exploiting the graph structure of the problem [37], [41]. The various approaches to solve the dual can be categorized into coordinate optimization and gradient based methods. Block coordinate methods converge fast but can get stuck in sub-optimal corners when optimizing the non-smooth dual [39], [20]. Since, we can recover better integer primal labels when closer to the optimum of the non-smooth dual, this can lead to poor solutions. On the other hand, supergradient methods can theoretically reach the global optimum [22]. However, they have an $O(\frac{1}{\epsilon^2})$ rate of convergence towards an ϵ-accurate solution.

These drawbacks can be addressed by approximating the dual with a smooth version. Smoothing the dual and applying accelerated gradient techniques was introduced by [16] and was further studied by [33], [34]. These algorithms reach an ϵ-accurate solution in $O(\frac{1}{\epsilon})$ iterations. Block coordinate approaches can also work with a smooth objective [26], [14], [28], which allows these algorithms to avoid sub-optimal corners. Some of these methods [14] can still get stuck, when the smoothing is reduced as we go closer to the optimum, in order to get more accurate results. Also, the scope for parallelization is more limited in block coordinate optimization methods compared to gradient based methods.

Alternating Direction Method of Multipliers (ADMM) inspired methods [25], [27] work with an augmented Lagrangian, which is also a smooth objective. However, convergence rate analysis for many of these methods has not been addressed and it is observed in practice, too. For example, AD3 [25] works well for many medium sized problems but can fail to converge in some instances. Also, scaling these methods to large vision problems with higher order cliques, has not been successful so far.

In recent years, Newton-type methods have led to state-of-the-art results in various Machine Learning problems [36], [24], [23]. These methods are able to take a better direction by considering curvature information and have

quadratic convergence rate when sufficiently close to the optimum. One of the challenges while solving any optimization problem, is the conditioning of the objective. Intuitively, a problem is ill-conditioned if the change in objective value, due to a perturbation in variable value, varies radically depending on the direction of the perturbation. People have found that first order methods are faster when the condition number is small or moderate but second-order Newton-type methods perform much better for ill-conditioned problems [11]. In MAP inference, the smoothing has to be reduced as one gets closer to the opimum, leading to a considerably ill-conditioned dual objective.

The main disadvantage of Newton-type methods is the need to compute the Hessian, which can be very costly. However, it is worth investigating whether this is indeed the case or not for the problem in hand. Moreover, quasi-Newton methods which work with a Hessian approximation, can also lead to state-of-the-art results [36], [5].

Our contributions are summarized as follows: (i) We show that for the smoothing based approach, it is possible to efficiently compute the Hessian and Hessian-vector product for a broad class of problems. (ii) We present a study of how to adapt Newton-type methods for MAP inference in higher-order MRFs. (iii) We demonstrate an efficient way to perform sum-product inference in chains of sparse pattern-based higher order cliques. This greatly enables the applicability of smoothing based approach for inference in higher order MRFs. (iv) We showcase how Newton-type methods can beat first order methods for higher-order datasets. The remainder of this paper is organized as follows: section 2 outlines the concept of MAP inference and the smoothing based approach. The main contributions of the paper are presented in sections 3 and 4, while experimental validation is in section 5, with a concluding discussion.

2. MAP inference by optimizing a smooth dual

Consider a graph $\mathcal{G}$, where V is the set of nodes, representing the random variables and C is the set of *cliques*, which enforce a certain relationship (e.g., smoothness) among the nodes they contain. Each node takes a label from a discrete set l. For example, object detection can be formulated using an MRF, where each node corresponds to a part of the object, cliques represent geometric constraints between the parts and the labels are locations in the image. MAP inference is the task of finding the labeling that maximizes the joint probability distribution of these random variables. It can be formulated as an equivalent energy minimization problem, as follows,

$$\text{minimize} \sum_{c \in C} \theta_c(\boldsymbol{x_c}) + \sum_{i=1}^{n} \theta_i(x_i) \tag{1}$$

where $\theta_i(x_i)$ is the potential of node i for label x_i and $\theta_c(\boldsymbol{x_c})$ is the potential of clique c for label $\boldsymbol{x_c}$. Cliques having more than two nodes become higher-order cliques.

Further, this energy minimization problem can be represented as an Integer Linear Program (ILP) as follows,

$$\begin{aligned} \text{minimize} \sum_{c} \sum_{\boldsymbol{x_c}} \theta_c(\boldsymbol{x_c})\phi_c(\boldsymbol{x_c}) + \sum_{i} \sum_{x_i} \theta_i(x_i)\phi_i(x_i) \\ \text{subject to} \sum_{x_{c \setminus i}} \phi_c(\boldsymbol{x_c}) = \phi_i(x_i), \quad \forall c, i \in c, x_i \\ \sum_{x_i} \phi_i(x_i) = 1, \quad \forall i; \quad \sum_{\boldsymbol{x_c}} \phi_c(\boldsymbol{x_c}) = 1, \quad \forall c \\ \phi_i(x_i) \in \{0,1\}, \forall i, x_i; \quad \phi_c(\boldsymbol{x_c}) \in \{0,1\} \forall c, \boldsymbol{x_c}. \end{aligned} \tag{2}$$

Here, $\phi_i(x_i)$ and $\phi_c(\boldsymbol{x_c})$ are indicator vectors for a given node i (resp., clique c) for a label x_i (resp., $\boldsymbol{x_c}$) and they are the discrete optimization variables. The constraints represent a polytope, called the *Marginal* polytope. Relaxing the integrality constraint (last line), results in the LP relaxation, which is defined over the *Local* polytope.

The Lagrangian dual of this LP relaxation is obtained through dual decomposition [22], [38]. The underlying idea is to decompose the graph into tractable sub-graphs, in order to derive the Lagrangian. This results in a concave, unconstrained and non-smooth optimization problem. Especially, if one treats each clique as the tractable sub-graph, then the approach given in [38], leads to the following dual,

$$\begin{aligned} \max_{\delta}. \sum_{c} \min_{\boldsymbol{x_c}}. \big(\theta_c(\boldsymbol{x_c}) - \sum_{i:i \in c} \delta_{ci}(x_i)\big) \\ + \sum_{i} \min_{x_i}. \big(\theta_i(x_i) \; + \sum_{c:i \in c} \delta_{ci}(x_i)\big) \end{aligned} \tag{3}$$

Here the dual variables are $\delta_{ci}(x_i)$, for each label x_i of each node i within each clique c. If all the cliques have k nodes each, with the nodes taking l labels, then the total number of dual variables is $|C|.k.l$. We will denote this number as $\mathcal{N}$ and $\boldsymbol{\delta} \in \mathbb{R}^{\mathcal{N}}$, is the vector of dual variables. We note that the size of the dual is much lesser than the primal (2), hence, there has been considerable research effort towards solving the MAP problem through the dual.

In MAP inference, there is an interplay between the sizes of the graph and the sub-graphs within dual decomposition [21] [42]. For medium sized graphs, decomposing into individual cliques, leads to excellent practical convergence. However, for large graphs, only bigger sub-graphs like chains of cliques lead to practical convergence [21], [29]. We would like to emphasize that the formulation shown in (3) decomposes the graph according to the cliques but in general, the sub-graphs can be bigger regions.

2.1. Smooth dual

The optimization of the non-smooth dual (3) has slow convergence and can stall at a point away from the opti-

mum. In order to reach closer to the optimum of the non-smooth dual, optimization over smoother versions are carried out. The smooth dual can be obtained by adding to the primal objective (2), entropies corresponding to all the nodes and cliques. The dual of this modified problem can be derived based on duality theory (refer [28], [16] for further details). The smooth dual looks very similar to the non-smooth one, where each minimum operation in (3) is replaced by the negative soft-max function. The smooth optimization problem takes the following form,

$$\begin{aligned} \max_{\delta}. \sum_{c} \operatorname*{smin}_{\boldsymbol{x_c}} \big(\theta_c(\boldsymbol{x_c}) - \sum_{i:i\in c} \delta_{ci}(x_i); \tau\big) \\ + \sum_{i} \operatorname*{smin}_{x_i} \big(\theta_i(x_i) \; + \sum_{c:i\in c} \delta_{ci}(x_i); \tau\big) \end{aligned} \tag{4}$$

We will denote the smooth dual function as $g(\boldsymbol{\delta})$. The negative soft-max function is defined as $\operatorname*{smin}_{x}(f(x);\tau) = \frac{-1}{\tau}\log\sum_x \exp(-\tau f(x))$. As τ increases, (4) is a closer approximation of (3) and we get closer to the optimum of the non-smooth dual. However, a smooth approximation becomes increasingly ill-conditioned as it approaches the shape of the non-smooth problem (§7, [31]). Hence, it is costlier to optimize for larger values of τ. It is better to start with a very smooth version and switch to less smoothness, with warm start from the previous smoother version [34].

3. Trust-Region Newton for MAP Inference

The central concept in Newton-type methods is the use of curvature information to compute the search direction. In each iteration, a local quadratic approximation is constructed and a step towards the minimum of this quadratic is taken. For the smooth dual $g(\boldsymbol{\delta})$, the equations to obtain the Newton direction are shown below. In these equations, $\boldsymbol{B}(\boldsymbol{\delta}) \in \mathbb{R}^{\mathcal{N}X\mathcal{N}}$ carries the curvature information and can be either the exact Hessian ($\nabla^2 g(\boldsymbol{\delta})$) or a modified Hessian or a Hessian approximation (e.g., quasi-Newton).

$$g(\boldsymbol{\delta}+\boldsymbol{p}) \approx g(\boldsymbol{\delta}) + \nabla g(\boldsymbol{\delta})^T\boldsymbol{p} + \boldsymbol{p}^T\boldsymbol{B}(\boldsymbol{\delta})\boldsymbol{p} \tag{5}$$

$$q(\boldsymbol{p}) = \nabla g(\boldsymbol{\delta})^T\boldsymbol{p} + \boldsymbol{p}^T\boldsymbol{B}(\boldsymbol{\delta})\boldsymbol{p} \tag{6}$$

$$\boldsymbol{p}^* = \operatorname*{argmin}_{\boldsymbol{p}} \nabla g(\boldsymbol{\delta})^T\boldsymbol{p} + \boldsymbol{p}^T\boldsymbol{B}(\boldsymbol{\delta})\boldsymbol{p} \tag{7}$$

$$\therefore \boldsymbol{B}(\boldsymbol{\delta})\boldsymbol{p}^* = -\nabla g(\boldsymbol{\delta}) \tag{8}$$

Here, the minimum $\boldsymbol{p}^*$ of the quadratic approximation $q(\boldsymbol{p})$ will be a descent direction for positive definite $\boldsymbol{B}(\boldsymbol{\delta})$ and a line search along this direction determines the step size. For unconstrained problems, the Newton direction is found by solving the linear system (8). Since, $\mathcal{N}$ is large for computer vision problems, we use Conjugate Gradients (CG) to obtain the Newton direction in our work.

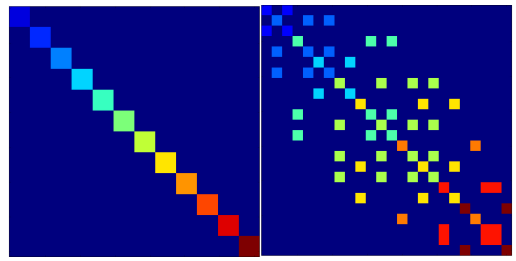

Figure 1. The two components of the Hessian. Within each component, blocks of the same color have the same values. In component one, there are as many unique blocks as cliques. In component two, each block-row/column has the same blocks.

For a Newton-type method, if $\boldsymbol{B}(\boldsymbol{\delta})$ is the exact Hessian ($\nabla^2 g(\boldsymbol{\delta})$) and if the backtracking line search tests full step length first, quadratic convergence is achieved, when sufficiently close to the optimum [4]. This is desirable, when compared to first order methods. As mentioned in section 2.1, smoothing is reduced as the algorithm proceeds and this results in ill-conditioning. Newton-type methods have some robustness against ill-conditioning due their affine invariance, i.e., for some functions $f(\boldsymbol{x})$ and $\bar{f}(\bar{\boldsymbol{x}}) = f(\boldsymbol{A}\boldsymbol{x})$, where $\boldsymbol{A}$ is an invertible square matrix, the iterates of Newton-type methods will be related as $\bar{\boldsymbol{p}}_k = \boldsymbol{A}\boldsymbol{p_k}$. Hence, in theory, these methods are not affected by ill-conditioning and in finite precision, the range of condition numbers in which Newton-type methods exibit robust behavior is more than that of first order methods. It is worth investigating how Newton-type methods handle MAP inference.

3.1. Hessian related computations

Even though Newton-type methods have advantages, it may be expensive to populate and solve the linear system (8). The computationally heavy steps for the Conjugate Gradient (CG) algorithm are: computing Hessian-vector products and regularly constructing and applying a preconditioning matrix. Generally, if the Hessian is difficult to populate, Complex Step Differentiation (CSD) [1] can give very accurate Hessian-vector products, with each product costing roughly one gradient computation, which can be costly to be used within CG. So, we investigate the possibility of efficiently populating the Hessian and obtaining fast Hessian-vector products.

As mentioned in section 2, for medium sized problems it is sufficient to decompose according to cliques and achieve practical convergence. We will now show that for decompositions according to small sub-graphs like individual cliques, the Hessian can be computed very efficiently. In fact, it only takes time of about twice the gradient computation time to compute both gradient and Hessian together.

If we take a closer look at the Hessian, we observe that it can be written as the sum of two components, as shown in

figure 1. Both components have a block structure. Consider the following quantities,

$$\mu_{ci}(x_i) = \frac{\sum\limits_{\boldsymbol{x_c}:\boldsymbol{x_c}(i)=x_i} \exp\big[\tau.(\theta_c(\boldsymbol{x_c}) - \sum\limits_{n:n\in c} \delta_{cn}(x_i))\big]}{\sum\limits_{\boldsymbol{x_c}} \exp\big[\tau.(\theta_c(\boldsymbol{x_c}) - \sum\limits_{n:n\in c} \delta_{cn}(x_n))\big]} \tag{9}$$

$$\mu_i(x_i) = \frac{\exp\big[\tau.(\theta_i(x_i) + \sum\limits_{k:i\in k} \delta_{ki}(x_i))\big]}{\sum\limits_{x_l} \exp\big[\tau.(\theta_i(x_l) + \sum\limits_{k:i\in k} \delta_{ki}(x_l))\big]} \tag{10}$$

The elements of component one can be written as $H_{c,ij}(x_i, x_j) = \tau\big(\mu_{cij}(x_i, x_j) - \mu_{ci}(x_i)\mu_{cj}(x_j)\big)$, where $\mu_{cij}(x_i, x_j)$ can be calculated like $\mu_{ci}(x_i)$ by fixing the labels of two nodes. This leads to a block diagonal matrix, with as many unique symmetric blocks as there are cliques. The elements of component two can be written as $H_{st,i}(x_i, x_i) = \tau\mu_i(x_i)(1 - \mu_i(x_i))$ and $H_{st,i}(x_i, x_j) = -\tau\mu_i(x_i)\mu_i(x_j)$. It has as many unique symmetric blocks as there are nodes and a given row or column has repeated copies of the same block. Here c, s, t are cliques, i, j member nodes, x_i, x_j node labels and $\boldsymbol{x}_c$ clique labelling. Also, it is component two, which contains off-diagonal blocks, which is caused by overlapping cliques.

Hence, the elements of the Hessian can be computed by only iterating once through all cliques and nodes. Iterations corresponding to pairs of overlapping cliques is avoided. For the gradient, arrays of values need to be computed by iterating once through all cliques and nodes. For the Hessian, we need to compute (symmetric) blocks of values at each clique and node. Practically due to cache re-use, this is achieved overall with the overhead time of one gradient computation only. These blocks can be readily used for parallelizing Hessian-vector multiplication and we do it with simple OpenMP code. Thus, efficient Hessian-vector products leads to an efficient CG routine in our case.

For several problems, the Hessian is sparse because a clique overlaps only with a few other cliques. Nevertheless, because of the special structure, there is no need to exploit this sparsity for computing the unique blocks of the Hessian. Also, if there are many overlapping cliques, computing Hessian-vector products is not adversely affected because we will only have more copies of the same block along each row of the component two matrix, corresponding to the shared node. Hence, data movement in computer memory gets limited. Moreover, these data structures can be readily used for constructing preconditioners.

3.2. Damping matrix approach

The smooth dual (4) is only strictly, not strongly, convex, i.e., away from the optimum, the Hessian is only positive semidefinite. In such a situation a trust-region approach is necessary to take meaningful Newton steps. Here, the same quadratic approximation (6) is minimized with the constraint $\|\boldsymbol{p}\| \leq \Delta$, where Δ is a trust radius. While developing our trust-region Newton method for MRF inference (TRN-MRF), we addressed several issues: how to enforce the trust-region, how to improve speed of convergence by coping with the ill-conditioning and how to design a suitable preconditioner. It is a combination of these choices that lead to an usable algorithm and we note that, what works for MAP inference, may not work for other tasks and vice-versa.

The Steihaug method seems like the first method to try for large problems. It has the nice property of shaping the trust-region into an ellipsoid, according to the landscape. This is achieved by minimizing (6), with the constraint $\|\boldsymbol{p}\|_{\boldsymbol{M}} \leq \Delta$, where $\boldsymbol{M}$ is a preconditioner and $\|\boldsymbol{p}\|_{\boldsymbol{M}} = \boldsymbol{p}^T\boldsymbol{M}\boldsymbol{p}$ [7]. However, in the absense of a good preconditioner, in a given outer Newton iteration, the algorithm quickly reaches the trust radius, before computing a good direction, leading to several outer iterations.

The Levenberg-Marquardt algorithm, which is generally used in least squares problems, offers another approach to impose a trust-region. The idea that we borrow is the damping matrix, which is a regularizer to address the strict convexity and the ill-conditioning. This matrix is added to the Hessian to obtain the modified Hessian in 8. The damping matrix restricts the Newton direction returned by CG to a trust-region. The Levenberg-Marquardt algorithm uses the scaled Hessian diagonal as the damping matrix. This choice gave very poor results for MAP inference. Instead, we modify the Hessian as follows, $\boldsymbol{B}(\boldsymbol{\delta}) = \nabla^2 g(\boldsymbol{\delta}) + \lambda\boldsymbol{I}$, where $\lambda > 0$. While Steihaug works explicitly with a trust radius Δ, we implicitly impose it through λ. This can be seen through these equations that tie the damping parameter λ to a trust radius Δ,

$$\begin{aligned} (\nabla^2 g(\boldsymbol{\delta}) + \lambda\boldsymbol{I})\boldsymbol{p}^* &= -\nabla g(\boldsymbol{\delta}) \\ \lambda(\Delta - \|\boldsymbol{p}^*\|) &= 0 \end{aligned} \tag{11}$$

In a sufficiently positive definite region (close to the optimum), λ is close to zero and the Newton direction $\boldsymbol{p}^*$ is computed with the true Hessian. If $\lambda > 0$ then $\|\boldsymbol{p}^*\| = \Delta$, i.e., the direction is restricted by the trust-region. In ill-conditioned regions, λ will be large and the enforced trust radius will be small. Thus, after every iteration, λ is adapted and it can be done as follows,

$$\begin{gathered} \rho < 0.25 : \lambda \leftarrow 2\lambda; \;\; 0.25 < \rho < 0.5 : \lambda \leftarrow \lambda \\ 0.5 < \rho < 0.9 : \lambda \leftarrow 0.5\lambda; \;\; 0.9 < \rho : \lambda \leftarrow 0.25\lambda \\ \text{where,} \quad \rho = \frac{g(\boldsymbol{\delta} + \boldsymbol{p}) - g(\boldsymbol{\delta})}{q(\boldsymbol{p}) - q(\boldsymbol{0})} \end{gathered} \tag{12}$$

ρ signifies how well the quadratic of equation (6) approximates the dual ($g(\boldsymbol{\delta})$). As the algorithm approaches the optimum, λ becomes vanishingly small and the algorithm reaches the true optimum without any perturbation.

3.3. Forcing sequence for CG truncation

Having found out the suitability of damping matrix based approach, it is still necessary to properly address the ill-conditioning caused by annealing. Trust-region Newton proceeds by taking approximate Newton steps, where in each outer iteration the run of CG iterations is truncated by a suitable criterion. However, as the algorithm approaches optimality, it is critical to solve the linear systems to greater accuracy and get better Newton steps [35]. Otherwise, the algorithm will take too long to converge or will not converge at all. Generally, at an outer iteration k, CG can be truncated at iteration j according to the following condition, $\|\boldsymbol{r}^j\| \leq \eta_k \|\nabla g(\boldsymbol{\delta}_k)\|$. Here, $\boldsymbol{r}^j = \boldsymbol{B}(\boldsymbol{\delta}_k)\boldsymbol{\delta}_k^j + \nabla g(\boldsymbol{\delta}_k)$, is the residual of equation 8, at iteration j of CG and η_k is referred to as the forcing sequence. Through η_k we reduce the residual and achieve more accurate Newton direction. For MAP inference, we found textbook choices of the forcing sequence leading to Newton iterations not converging because the residual never becomes low enough to achieve the more accurate directions required for further progress. Hence, for TRN-MRF, the following criterion has been designed: $\eta_k = min(\frac{\epsilon_\tau}{k}, \sqrt{\|\nabla g(\boldsymbol{\delta}_k)\|})$. This condition naturally imposes stronger condition on the residual for later iterations and also, the term (ϵ_τ), which takes smaller values as the annealing progresses, ensures that sufficiently accurate Newton steps are obtained, as smoothing reduces.

3.4. Clique based Preconditioner and Backtracking search

In order to further improve the efficiency of TRN-MRF, we will address one important aspect that influences trust-region Newton methods. This concerns the computational efficiency of the Conjugate Gradient routine. Convergence of CG iterations is a function of the number of distinct eigen value clusters of the linear system and a well preconditioned system ($\boldsymbol{M}^{-1}\boldsymbol{A}\boldsymbol{x} = \boldsymbol{M}^{-1}\boldsymbol{b}$) will have fewer of clusters. Matrix M should be as similar to A as possible and should be efficient to construct and invert.

Standard preconditioning approaches like incomplete Cholesky, quasi-Newton and multigrid, fail to cope with generic MAP inference problems. A clique based block diagonal preconditioner has along the diagonal, clique specific blocks ($\boldsymbol{H}^c$) of double derivative terms, i.e., $H^c_{i,x_i,j,x_j} = \frac{\partial^2 g(\boldsymbol{\delta})}{\partial \delta_{ci}(x_i)\partial \delta_{cj}(x_j)}$, where i, j are nodes belonging to the clique and x_i, x_j are their labels. Since its structure is closely related to the MAP inference problem, it performs quite well. The computational cost is of computing and inverting these blocks individually. This is done at the same time as computing the gradient and the Hessian. Applying this preconditioner corresponds to matrix-vector multiplications, involving the block inverses.

For sufficiently large values of λ, the modified Hessian is automatically well-conditioned and CG will converge fast. It is towards the optimum, when λ reduces to vanishing values, that CG runs for large number of iterations. We observed considerable improvement in CG performance with this preconditioner and the results shown in this paper are based on this. Also, close to the optimum, there will be CG runs taking the maximum allowed number of iterations. We have set it as 250 for all our experiments.

The last aspect concerns backtracking search, once the Newton direction has been computed by the CG routine. When ρ in equation (12) is less than a small value (say, ϵ_ρ), it means a step of either very less decrease or an increase in the function value. So, we cannot directly take a step along this direction. However, it is desirable to take a step of sufficient decrease in every outer iteration of TRN-MRF and hence, we perform a backtracking search in these cases [30]. The backtracking can be either done along a straight line or along a curved path (a subset of CG iterations). Although backtracking along a curved path gives good results in other problems [24], we observed poor results for MAP inference: the final direction was very close to the steepest descent direction. On the other hand, performing a backtracking line search along the direction computed by CG, gave a huge speed-up for TRN-MRF. We have implemented a cubic interpolation based search, which is very efficient.

3.5. Annealing schedule and stopping condition

[34] suggests annealing τ by periodically computing the primal-dual (PD) gap of the smooth problem. For intermediate dual variables, they recover feasible primal variables by solving a small LP (called a transportation problem) for each clique. Since, their approach is adapted to pairwise graphs, they achieve results with less than thousand oracle calls (an oracle call is either an iteration or computation of the PD gap). However, for higher-order MRFs, computing feasible primal variables and the primal objective is costly. Hence, computing PD gap of the smooth problem, every few iterations greatly affects computational efficiency.

We have used a simple but intuitive criterion for judging when to anneal. Since, we are working with a concave function, regions with lesser values of gradient euclidean norm are guaranteed to be closer to the optimum than regions with much greater values. Hence, if $\nabla g(\boldsymbol{\delta}_k)$ signifies the gradient after running k iterations with a given τ, we update $\tau \leftarrow \alpha\tau, 1 < \alpha$, if $\|\nabla g(\boldsymbol{\delta}_k)\|_2 < \gamma_\tau$. If we impose a strong enough threshold γ_τ, we are guaranteed to achieve sufficient improvement for that particular τ and annealing can be performed. We have defined, $\gamma_\tau = \beta\|\nabla g(\boldsymbol{\delta})\|$, just after τ is annealed. Similar to the proof in [34], this annealing approach will work with any optimization algorithm that will converge to the global optimum for a fixed value of τ. All the algorithms tested by us, have this guarantee for smooth, concave problems. Also, we ensure τ has reached a

large enough value τ_{max} in order to obtain accurate results.

In order to exit the least smooth problem, [34] use the non-smooth PD gap. We have observed that TRN-MRF, due to its quadratic convergence rate, can exit based on classical gradient based condition itself. More precisely, with the l_∞ norm and a threshold of $\zeta = 10^{-3}$ (§8, [12]), TRN-MRF achieves good exit behaviour. However, first order methods take too long a time to achieve this gradient based exit condition and many times never do so. Hence, we have implemented a PD gap based approach, so that all algorithms can exit gracefully. The approach in [34], is available only for pairwise graphs in the openGM library and implementing small LP solvers for all the higher order cliques looks challenging. [28] proposed a method involving only closed form calculations and we have implemented their approach.

Our complete trust-region Newton method, within an annealing framework, is described in Algorithm 1. In our experiments, we set, $\lambda_0 = 1$, $\alpha = 2$, $\beta = \frac{1}{6}$, $\epsilon_\rho = 10^{-4}$, $\zeta = 10^{-3}$, $\tau_{max} = 2^{13}$ and $\epsilon_\tau = 0.1$ if $\tau < \frac{\tau_{max}}{4}$, 0.01 if $\frac{\tau_{max}}{4} < \tau < \frac{\tau_{max}}{2}$, 0.001 if $\frac{\tau_{max}}{2} < \tau$.

Algorithm 1 TRN-MRF: Trust-region Newton for MAP inference

Input: $\lambda_0, \tau, \tau_{max} > 0; \boldsymbol{\delta}_0 \in \mathbb{R}^{\mathcal{N}}; \alpha > 1$.
$\lambda \leftarrow \lambda_0, \gamma_\tau = \beta \|\nabla g(\boldsymbol{\delta}_0)\|_2$
while $\|\nabla g(\boldsymbol{\delta}_k)\|_\infty > \zeta$ or $\tau < \tau_{max}$ **do**
  if $\|\nabla g(\boldsymbol{\delta}_k)\|_2 \leq \gamma_\tau$ and $\tau < \tau_{max}$ **then**
    $\tau \leftarrow \alpha\tau$; $\gamma_\tau = \beta\|\nabla g(\boldsymbol{\delta}_k)\|_2$; adjust ϵ_τ
  end if
  $\boldsymbol{B}(\boldsymbol{\delta}_k) = \nabla^2 g(\boldsymbol{\delta}_k) + \lambda \boldsymbol{I}$
  set $\eta_k = min(\frac{\epsilon_\tau}{k}, \sqrt{\|\nabla g(\boldsymbol{\delta}_k)\|})$
  Run CG while $\|\boldsymbol{r}^j\| > \eta_k \|\nabla g(\boldsymbol{\delta}_k)\|$
  obtain Newton direction $\boldsymbol{p}$ and calculate ρ
  update λ according to equation (12)
  if $\rho < \epsilon_\rho$ then backtracking line search along $\boldsymbol{p}$ to obtain Newton step $\boldsymbol{p}_k$
  $\boldsymbol{\delta}_{k+1} = \boldsymbol{\delta}_k + \boldsymbol{p}_k$
end while

4. Scalable Smoothing based approach

Even though smoothing based approach has been scaled for large pairwise graphs [34], higher order MRFs with large label spaces and/or large graphs are less studied. In this section we show an efficient way to compute the smoothing operation with large label spaces for a very useful class of clique potentials. Next we demonstrate the use of quasi-Newton methods for problems with large graphs.

4.1. Pattern-based Smoothing

In the smooth dual (4), we denote the first term as $g_1(\boldsymbol{\delta})$. It corresponds to cliques and involves log-sum-exp calculations over all possible labellings for each clique. This scales exponentially (l^k) with clique size k, where each node takes l labels. Hence, computing the gradient of this dual is computationally heavy, especially for higher order cliques. [16] observed that these terms in the gradient correspond to marginal probabilities in a suitably defined graphical model. Hence, they can be computed using the sum-product algorithm [19]. Still, $O(l^k)$ complexity remains.

Sparse, pattern-based clique potentials are very useful in computer vision [21], [32]. In these potentials, a big majority of the labellings take a constant (usually high) value and a small subset (of size s) take other significant values. Many clique potentials fit this description. [21] showed an efficient way to perform max-product computation in chains of such cliques. It is not clear how to extend their work to sum-product. [8] showed a sum-product approach in pairwise MRFs. They achieved a complexity of $O(2l + s)$, instead of $O(l^2)$. They mention that their idea can be used for higher order cliques with a factor graph representation but don't go into details. We have found that with a factor graph representation, their approach has a complexity of $O(l^{k-1}+l+s)$. Instead of a factor graph, if a clique tree representation (§10, [19]) is used, it is possible to achieve much better complexity of $O(l^{k-n}+l^n+s)$, where n is the size of a subset of clique nodes. For individual cliques, these subsets will be (nearly) equal sized partitions of the clique. For clique chains, this is the separator set between two overlapping cliques. The separator set is the nodes shared by two cliques. For example, for cliques of size 4, with subset of size 2, one can quickly see the efficiency gain.

In the following, we will consider clique chains. This is for simple notation and these results are applicable to trees. Also, these results can be reduced to the case of individual cliques. While computing the gradient, the quantity $\frac{\partial g_1}{\partial \delta_{ci}(x_i)}$ is equal to $p_i^c(x_i)$, i.e., the marginal probability of node i taking label x_i in clique c of a suitably defined graphical model. Let $\psi_c(\boldsymbol{x}_c)$ and $\psi_i(x_i)$ denote the clique and node potentials, respectively and a, b, c be three neighbouring cliques in a chain, with m and n being the separator set between a and b, b and c, respectively. Sum-product message is passed from b to c, as follows,

$$\begin{aligned} m_{bc}(\boldsymbol{x}_n) &= \sum_{\boldsymbol{x}_{b\setminus n}} \psi_b(\boldsymbol{x}_b)\nu(\boldsymbol{x}_{b\setminus n}) \\ \text{where, } \nu(\boldsymbol{x}_{b\setminus n}) &= \Big(\prod_{i\in b\setminus n} \psi_i(x_i)\Big) m_{ab}(\boldsymbol{x}_{b\setminus n}) \end{aligned} \tag{13}$$

$m_{ab}(\boldsymbol{x}_{b\setminus n})$ is derived from the message sent by clique a to b. This message was for the nodes in m and $m_{ab}(\boldsymbol{x}_{b\setminus n})$ is obtained by marginalization (summation). Sometimes, $m = b \setminus n$. Let $\bar{\psi}$ denote the constant clique potential and let $Pat(\boldsymbol{x}_n)$ denote the pattern-based clique labellings corresponding to the seperator set labelling $\boldsymbol{x}_n$, then the first

equation of (13) can be written as,

$$m_{bc}(\boldsymbol{x}_n) = \sum_{\substack{\boldsymbol{x}_{b\setminus n} \\ \in Pat(\boldsymbol{x}_n)}} \psi_b(\boldsymbol{x}_b)\nu(\boldsymbol{x}_{b\setminus n}) + \sum_{\substack{\boldsymbol{x}_{b\setminus n} \\ \notin Pat(\boldsymbol{x}_n)}} \bar{\psi}\nu(\boldsymbol{x}_{b\setminus n})$$
$$= \sum_{\substack{\boldsymbol{x}_{b\setminus n} \\ \in Pat(\boldsymbol{x}_n)}} \big(\psi_b(\boldsymbol{x}_b) - \bar{\psi}\big)\nu(\boldsymbol{x}_{b\setminus n}) + \bar{\psi}\sum_{\boldsymbol{x}_{b\setminus n}} \nu(\boldsymbol{x}_{b\setminus n}) \tag{14}$$

In (14), the second summation is computed only once ($O(l^{k-n})$) and used for all elements of $\boldsymbol{x}_n$. With a loop of complexity $O(s)$, the set of first summations for each element of $\boldsymbol{x}_n$ can be computed. Then iterating once through $\boldsymbol{x}_n$, gives the message in $O(l^n)$. Within this last loop to compute the message, the probabilities $p_i^c(x_i)$ of the corresponding nodes are computed. Also, the Hessian requires node pair marginals ($\mu_{c,ij}(x_i, x_j)$, section 3.1) and they are calculated by parallelized sweeps of the sum-product algorithm working on independent subsets of the cliques.

4.2. Quasi-Newton approach

In section 2, we pointed out the trade-off between the sizes of the graph and the sub-graphs in dual decomposition. This becomes critical for large problems like stereo, where a decomposition based on individual cliques does not lead to practical convergence [21], [29]. Larger sub-graphs are needed and chains of higher order cliques is suitable for grid graphs. As mentioned in section 4.1, the Hessian for MAP inference requires node pair marginals and it is very costly to calculate node pair marginals for clique chains. Hence, it is not practical to populate the Hessian for problems which require large sub-graphs. Since we have promising results with trust-region Newton for medium sized problems with individual clique based decomposition, we explore quasi-Newton methods for large problems. Note that quasi-Newton methods have only super-linear convergence rate when sufficiently close to the optimum.

Quasi-Newton methods build an approximate Hessian by low rank updates of an initial approximation (usually, a scaled Identity matrix) with vector pairs, $\boldsymbol{s}_k \triangleq \boldsymbol{\delta}_{k+1} - \boldsymbol{\delta}_k$ and $\boldsymbol{y}_k \triangleq \nabla g(\boldsymbol{\delta}_{k+1}) - \nabla g(\boldsymbol{\delta}_k)$. For large problems a limited memory variant is required, based on the most recent m pairs of vectors. One can either approximate the Hessian or the Hessian inverse. For not strongly convex and/or ill-conditioned problem, the Hessian inverse approximation leads to large Newton steps. Downscaling them to a trust region radius does not help, as the direction itself is poor to start with. A trust-region based approach using the Hessian approximation is the better choice. Hence, we take the same approach as Algorithm 1 with an Hessian approximation (based on L-BFGS) in the place of $\nabla^2 g(\boldsymbol{\delta}_k)$.

The Hessian approximation is of the form of an Identity + rank-r matrix. The Conjugate Gradient algorithm converges in $O(r)$ iterations for such a linear sytem (§11.3.4, [13]). Morever, matrix-vector products in quasi-Newton can be obtained through a compact representation [6]. Thus the cost for the CG routine is a very small fraction of the cost of computing the gradient (0.5% in our experiments). Thus, given the iteration cost of quasi-Newton being comparable to first order methods, it is worthwhile to check whether quasi-Newton converges faster to the optimum.

5. Experiments

We present results based on higher-order MRF models. As our baseline, we used FISTA with backtracking line search [3], Smooth Coordinate Descent (SCD) based on Star update [28] and AD3 [25]. Note that Star update based SCD, consistently performed better than Max-Sum Diffusion based SCD in our experiments.

First, we tested on medium sized problems and compared TRN-MRF, FISTA, SCD and AD3. We formulate the problem of matching two sets of interest points as MAP inference, based on [9]. For n points, each point in the source can be matched to any of the points in the target, i.e., n labels. The MRF is constructed by generating $4n$ triangles in the source and each triangle in the source and target are characterized by tuples of side length. For each source tuple, we find the top 30K nearest neighbours among the target tuples. 30K is much lesser than all possible triangles in the target and this is a sparse, pattern-based clique potential. The higher-order cliques potentials are defined as $\exp(\frac{-1}{\gamma}((s_a - t_a)^2 + (s_b - t_b)^2 + (s_c - t_c)^2))$, where s and t refer to source and target, respectively and γ is the average squared differences between source tuple and its 30K nearest neighbours in the target. To get state-of-the-art results additional terms to disallow many-to-one mapping will be needed. For the sake of simplicity we ignore this issue.

We first tested with a synthetic problem, with $n = 81$ on the 2D plane. We added Gaussian noise to these points to create the target image. The unary potentials are defined by assigning the value i to node i in both the images and taking the absolute differences. The results are presented in table 1, where two levels of added noise were tested.

We next tested with matching points on the House dataset [2]. $n = 74$ points are marked in all the frames and the points in the first frame are matched with points in later frames (110 being the last frame). The unaries are set to zero. The camera rotates considerably around the object and we show results for three frames in table 2.

The main reason for being able to tackle such problems is the efficient computation of log-sum-exp (section 4.1). TRN-MRF, FISTA and SCD, all benefit from this. AD3 can also exploit sparse, pattern-based potentials, since, in its inner loop, max-product computations take place. The same steps shown for sum-product, can be used for max-product by replacing summing operation by max operation. Since,

	$\sigma = 0.5$				$\sigma = 0.8$			
Algorithm	TRN	SCD	FISTA	AD3	TRN	SCD	FISTA	AD3
time (seconds)	948.9	1742.7	4368.1	369.35 (738.7)	2513.8	6884.2	9037.1	No convergence
Non-smooth dual	-279.69	-279.69	-279.69	-279.69	-259.28	-258.27	-258.7	-260.94
Non-smooth primal	-279.67	-279.69	-279.66	N/A	-261.04	-258.36	-258.86	N/A
Integer primal	-279.69	-279.69	-279.69	-279.69	-247.86	-248.24	-250.3	-249.82

Table 1. Results for synthetic problem.

	Frame 70				Frame 90				Frame 110			
Algorithm	TRN	SCD	FISTA	AD3	TRN	SCD	FISTA	AD3	TRN	SCD	FISTA	AD3
time (seconds)	2374.5	3702.9	11964.5	1428.7 (2857.4)	4731.6	4206.4	12561.2	2303.05 (4606.1)	4451.8	10498.8	21171.1	No convergence
Non-smooth dual	-368.59	-368.59	-368.59	-368.59	-337.81	-337.81	-337.81	-337.81	-333.03	-331.51	-331.31	-335.5
Non-smooth primal	-368.56	-368.57	-368.37	N/A	-337.77	-337.78	-337.36	N/A	-336.16	-331.95	-330.65	N/A
Integer primal	-368.59	-368.59	-368.59	-368.59	-337.81	-337.81	337.81	-337.81	-315.93	-317.69	-317.78	314.16

Table 2. Results for House dataset.

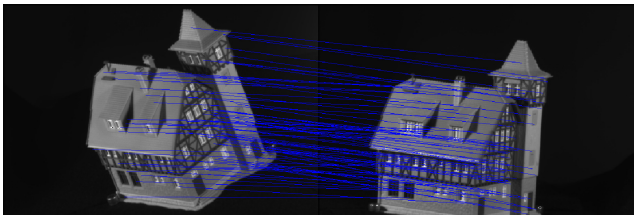

Figure 2. Matching 1st frame to 90th frame.

these modifications cannot be made to the AD3 version in openGM, we show within brackets the actual time taken by openGM's AD3. Since, our current implementation of sparse sum-product gives two times speed-up, the outside figure is half that value.

An important observation is that TRN-MRF, FISTA and SCD have reliable convergence compared to AD3 (an ADMM based approach). Among these three, TRN-MRF is very competitive and is the fastest in many cases. The quadratic convergence rate guarantee of TRN-MRF is evident because in many cases the stricter gradient based exit condition is reached at the same time or before the PD gap based exit condition. The gradient based exit condition is never reached before the PD gap condition by any of the first order methods. We do simple rounding of the primal variables to recover the labels. This rounding leads to energies that can be slightly better or worse. The crux of this paper is about getting closer to the global optimum of the non-smooth dual. Improved rounding schemes based on local search, will surely lead to better final labelling. AD3 has its own rounding scheme and outputs its results.

Next, we present results for stereo with curvature prior on 1×3 and 3×1 cliques. The clique energy is truncated, i.e., pattern-based. The unaries are based on absolute difference. We present results for Tsukuba, image size 144×192, 16 depth levels. This is a large problem and pattern-based sum-product was computed on clique chains. We compare quasi-Newton with FISTA and AD3. For large problems, the PD gap based criterion [28] never leads to convergence for both quasi-Newton and FISTA. So, a simultaneous criterion on function value difference, variable difference and gradient has been used (adapted from §8.2.3.2 [12]). AD3 showed poor convergence behavior for this large problem.

Algorithm	Quasi-Newton	FISTA
Iterations	357	594
Non-smooth dual	29105.9	29105.5
Integer primal	29347	29282.5

Table 3. Stereo estimation: Tsukuba.

Figure 3. Tsukuba result for quasi-Newton.

6. Discussion

We presented Newton-type methods that offer convergence guarantee and very good convergence rates, through appropriate choices made concerning their algorithmic components. Specifically, for problems in which sum-product computation is efficient, these Newton-type methods are very suitable. We showed promising results with higher-order MRFs of medium and large sizes. We hope this work spurs further research on exploiting curvature information within optimization algorithms for MAP inference.

7. Acknowledgments

Thanks to the reviewers for their helpful comments.

References

[1] Complex step differentiation. http://blogs.mathworks.com/cleve/2013/10/14/complex-step-differentiation/. Accessed: 2016-10-30. 3

[2] House dataset. http://vasc.ri.cmu.edu//idb/html/motion/house/index.html. Accessed: 2016-10-30. 7

[3] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009. 7

[4] D. P. Bertsekas. *Nonlinear programming*. Athena scientific Belmont, 1999. 3

[5] R. H. Byrd, S. Hansen, J. Nocedal, and Y. Singer. A stochastic quasi-newton method for large-scale optimization. *SIAM Journal on Optimization*, 26(2):1008–1031, 2016. 2

[6] R. H. Byrd, J. Nocedal, and R. B. Schnabel. Representations of quasi-newton matrices and their use in limited memory methods. *Mathematical Programming*, 63(1-3):129–156, 1994. 7

[7] A. R. Conn, N. I. Gould, and P. L. Toint. *Trust region methods*. SIAM, 2000. 4

[8] J. M. Coughlan and H. Shen. An embarrassingly simple speed-up of belief propagation with robust potentials. *arXiv preprint arXiv:1010.0012*, 2010. 6

[9] O. Duchenne, F. Bach, I.-S. Kweon, and J. Ponce. A tensor-based algorithm for high-order graph matching. *IEEE transactions on pattern analysis and machine intelligence*, 33(12):2383–2395, 2011. 7

[10] A. Fix, C. Wang, and R. Zabih. A primal-dual algorithm for higher-order multilabel markov random fields. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1138–1145. IEEE, 2014. 1

[11] K. Fountoulakis and J. Gondzio. Performance of first- and second-order methods for big data optimization. *arXiv preprint arXiv:1503.03520*, 2015. 2

[12] P. E. Gill, W. Murray, and M. H. Wright. Practical optimization. 1981. 6, 8

[13] G. H. Golub and C. F. Van Loan. *Matrix computations*, volume 3. JHU Press, 2012. 7

[14] T. Hazan and A. Shashua. Norm-product belief propagation: Primal-dual message-passing for approximate inference. *IEEE Transactions on Information Theory*, 56(12):6294–6316, 2010. 1

[15] H. Ishikawa. Transformation of general binary mrf minimization to the first-order case. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(6):1234–1249, 2011. 1

[16] V. Jojic, S. Gould, and D. Koller. Accelerated dual decomposition for map inference. In *International Conference on Machine Learning (ICML)*, pages 503–510, 2010. 1, 3, 6

[17] J. Kappes, B. Andres, F. Hamprecht, C. Schnörr, S. Nowozin, D. Batra, S. Kim, B. Kausler, T. Kröger, J. Lellmann, N. Komodakis, B. Savchynskyy, and C. Rother. A comparative study of modern inference techniques for structured discrete energy minimization problems. *International Journal of Computer Vision*, 115(2):155–184, 2015. 1

[18] P. Kohli, M. P. Kumar, and P. H. Torr. P^3 & beyond: Move making algorithms for solving higher order functions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(9):1645–1656, 2009. 1

[19] D. Koller and N. Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009. 6

[20] V. Kolmogorov. A new look at reweighted message passing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(5):919–930, 2015. 1

[21] N. Komodakis and N. Paragios. Beyond pairwise energies: Efficient optimization for higher-order mrfs. In *Computer Vision and Pattern Recognition (CVPR)*, pages 2985–2992. IEEE, 2009. 1, 2, 6, 7

[22] N. Komodakis, N. Paragios, and G. Tziritas. MRF energy minimization and beyond via dual decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(3):531–552, 2011. 1, 2

[23] J. Lee, Y. Sun, and M. Saunders. Proximal newton-type methods for convex optimization. In *Advances in Neural Information Processing Systems (NIPS)*, pages 836–844, 2012. 1

[24] J. Martens. Deep learning via hessian-free optimization. In *International Conference on Machine Learning (ICML)*, pages 735–742, 2010. 1, 5

[25] A. F. Martins, M. A. Figueiredo, P. M. Aguiar, N. A. Smith, and E. P. Xing. Ad3: Alternating directions dual decomposition for map inference in graphical models. *Journal of Machine Learning Research*, 16:495–545, 2015. 1, 7

[26] T. Meltzer, A. Globerson, and Y. Weiss. Convergent message passing algorithms: a unifying view. In *Uncertainty in Artificial Intelligence (UAI)*, pages 393–401, 2009. 1

[27] O. Meshi and A. Globerson. An alternating direction method for dual map lp relaxation. In *Machine Learning and Knowledge Discovery in Databases*, pages 470–483. Springer, 2011. 1

[28] O. Meshi, T. Jaakkola, and A. Globerson. Smoothed coordinate descent for map inference. *Advanced Structured Prediction*, pages 103–131, 2014. 1, 3, 6, 7, 8

[29] O. Miksik, V. Vineet, P. Pérez, P. H. Torr, and F. Cesson Sévigné. Distributed non-convex admm-inference in large-scale random fields. In *British Machine Vision Conference (BMVC)*, 2014. 2, 7

[30] J. Nocedal and Y.-x. Yuan. *Combining Trust Region and Line Search Techniques*, volume 14 of *Applied Optimization*, pages 153–175. Springer, 1998. 5

[31] R. T. Rockafellar and R. J.-B. Wets. *Variational analysis*, volume 317. Springer Science & Business Media, 2009. 3

[32] C. Rother, P. Kohli, W. Feng, and J. Jia. Minimizing sparse higher order energy functions of discrete variables. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1382–1389. IEEE, 2009. 6

[33] B. Savchynskyy, S. Schmidt, J. Kappes, and C. Schnörr. A study of nesterov's scheme for lagrangian decomposition and map labeling. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1817–1823. IEEE, 2011. 1

[34] B. Savchynskyy, S. Schmidt, J. Kappes, and C. Schnörr. Efficient mrf energy minimization via adaptive diminish-

ing smoothing. *Uncertainty in Artificial Intelligence (UAI)*, pages 746–755, 2012. 1, 3, 5, 6

[35] T. Schlick and M. Overton. A powerful truncated newton method for potential energy minimization. *Journal of Computational Chemistry*, 8(7):1025–1039, 1987. 5

[36] M. Schmidt. *Graphical model structure learning using L1-regularization*. PhD thesis, 2010. 1, 2

[37] M. Shlezinger. Syntactic analysis of two-dimensional visual signals in the presence of noise. *Cybernetics and systems analysis*, 12(4):612–628, 1976. 1

[38] D. Sontag, A. Globerson, and T. Jaakkola. Introduction to dual decomposition for inference. *Optimization for Machine Learning*, 1:219–254, 2011. 2

[39] D. Sontag, T. Meltzer, A. Globerson, T. Jaakkola, and Y. Weiss. Tightening lp relaxations for map using message passing. In *Uncertainty in Artificial Intelligence (UAI)*, pages 503–510, 2008. 1

[40] V. Vineet, J. Warrell, and P. H. Torr. Filter-based mean-field inference for random fields with higher-order terms and product label-spaces. *International Journal of Computer Vision*, 110(3):290–307, 2014. 1

[41] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. Map estimation via agreement on trees: message-passing and linear programming. *IEEE Transactions on Information Theory*, 51(11):3697–3717, 2005. 1

[42] H. Wang and D. Koller. A fast and exact energy minimization algorithm for cycle mrfs. In *Proceedings of the 30th International Conference on*, volume 1, page 1, 2013. 2