Abstract. Dense conditional random fields (CRF) with Gaussian pairwise potentials have emerged as a popular framework for several computer vision applications such as stereo correspondence and semantic segmentation. By modeling long-range interactions, dense CRFs provide a more detailed labelling compared to their sparse counterparts. Variational inference in these dense models is performed using a filtering-based mean-field algorithm in order to obtain a fully-factorized distribution minimizing the Kullback-Leibler divergence to the true distribution. In contrast to the continuous relaxation-based energy minimisation algorithms used for sparse CRFs, the mean-field algorithm fails to provide strong theoretical guarantees on the quality of its solutions. To address this deficiency, we show that it is possible to use the same filtering approach to speed-up the optimisation of several continuous relaxations. Specifically, we solve a convex quadratic programming (QP) relaxation using the efficient Frank-Wolfe algorithm. This also allows us to solve difference-of-convex relaxations via the iterative concave-convex procedure where each iteration requires solving a convex QP. Finally, we develop a novel divide-and-conquer method to compute the subgradients of a linear programming relaxation that provides the best theoretical bounds for energy minimisation. We demonstrate the advantage of continuous relaxations over the widely used mean-field algorithm on publicly available datasets.

Keywords: Energy minimisation, Dense CRF, Inference, Linear Programming, Quadratic Programming

1 Introduction

Discrete pairwise conditional random fields (CRFs) are a popular framework for modelling several problems in computer vision. In order to use them in practice, one requires an energy minimisation algorithm that obtains the most likely output for a given input. The energy function consists of a sum of two types of terms: unary potentials that depend on the label for one random variable at a time and pairwise potentials that depend on the labels of two random variables.
Traditionally, computer vision methods have employed sparse connectivity structures, such as 4 or 8 connected grid CRFs. Their popularity lead to a considerable research effort in efficient energy minimisation algorithms. One of the biggest successes of this effort was the development of several accurate continuous relaxations of the underlying discrete optimisation problem \cite{1,2}. An important advantage of such relaxations is that they lend themselves easily to analysis, which allows us to compare them theoretically \cite{3}, as well as establish bounds on the quality of their solutions \cite{4}.

Recently, the influential work of Krähenbühl and Koltun \cite{5} has popularised the use of dense CRFs, where each pair of random variables is connected by an edge. Dense CRFs capture useful long-range interactions thereby providing finer details on the labelling. However, modeling long-range interactions comes at the cost of a significant increase in the complexity of energy minimisation. In order to operationalise dense CRFs, Krähenbühl and Koltun \cite{5} made two key observations. First, the pairwise potentials used in computer vision typically encourage smooth labelling. This enabled them to restrict themselves to the special case of Gaussian pairwise potentials introduced by Tappen et al. \cite{6}. Second, for this special case, it is possible to obtain a labelling efficiently by using the mean-field algorithm \cite{7}. Specifically, the message computation required at each iteration of mean-field can be carried out in $O(N)$ operations where $N$ is the number of random variables (of the order of hundreds of thousands). This is in contrast to a naïve implementation that requires $O(N^2)$ operations. The significant speed-up is made possible by the fact that the messages can be computed using the filtering approach of Adams et al. \cite{8}.

While the mean-field algorithm does not provide any theoretical guarantees on the energy of the solutions, the use of a richer model, namely dense CRFs, still allows us to obtain a significant improvement in the accuracy of several computer vision applications compared to sparse CRFs \cite{5}. However, this still leaves open the intriguing possibility that the same filtering approach that enabled the efficient mean-field algorithm can also be used to speed-up energy minimisation algorithms based on continuous relaxations. In this work, we show that this is indeed possible.

In more detail, we make three contributions to the problem of energy minimisation in dense CRFs. First, we show that the conditional gradient of a convex quadratic programming (QP) relaxation \cite{1} can be computed in $O(N)$ complexity. Together with our observation that the optimal step-size of a descent direction can be computed analytically, this allows us to minimise the QP relaxation efficiently using the Frank-Wolfe algorithm \cite{9}. Second, we show that difference-of-convex (DC) relaxations of the energy minimisation problem can be optimised efficiently using an iterative concave-convex procedure (CCCP). Each iteration of CCCP requires solving a convex QP, for which we can once again employ the Frank-Wolfe algorithm. Third, we show that a linear programming (LP) relaxation \cite{2} of the energy minimisation problem can be optimised efficiently via subgradient descent. Specifically, we design a novel divide-and-conquer method to compute the subgradient of the LP. Each subproblem of our
method requires one call to the filtering approach. This results in an overall run-time of $O(N \log(N))$ per iteration as opposed to an $O(N^2)$ complexity of a naïve implementation. It is worth noting that the LP relaxation is known to provide the best theoretical bounds for energy minimisation with metric pairwise potentials \cite{2}.

Using standard publicly available datasets, we demonstrate the efficacy of our continuous relaxations by comparing them to the widely used mean-field baseline for dense CRFs.

2 Related works

Krähenbühl and Koltun popularised the use of densely connected CRFs at the pixel level \cite{5}, resulting in significant improvements both in terms of the quantitative performance and in terms of the visual quality of their results. By restricting themselves to Gaussian edge potentials, they made the computation of the message in parallel mean-field feasible. This was achieved by formulating message computation as a convolution in a higher-dimensional space, which enabled the use of an efficient filter-based method \cite{8}.

While the original work \cite{5} used a version of mean-field that is not guaranteed to converge, their follow-up paper \cite{10} proposed a convergent mean-field algorithm for negative semi-definite label compatibility functions. Recently, Baqué et al. \cite{11} presented a new algorithm that has convergence guarantees in the general case. Vineet et al. \cite{12} extended the mean-field model to allow the addition of higher-order terms on top of the dense pairwise potentials, enabling the use of co-occurrence potentials \cite{13} and $P^n$-Potts models \cite{14}.

The success of the inference algorithms naturally lead to research in learning the parameters of dense CRFs. Combining them with Fully Convolutional Neural Networks \cite{15} has resulted in high performance on semantic segmentation applications \cite{16}. Several works \cite{17,18} showed independently how to jointly learn the parameters of the unary and pairwise potentials of the CRF. These methods led to significant improvements on various computer vision applications, by increasing the quality of the energy function to be minimised by mean-field.

Independently from the mean-field work, Zhang et al. \cite{19} designed a different set of constraints that lends itself to a QP relaxation of the original problem. Their approach is similar to ours in that they use continuous relaxation to approximate the solution of the original problem but differ in the form of the pairwise potentials. The algorithm they propose to solve the QP relaxation has linearithmic complexity while ours is linear. Furthermore, it is not clear whether their approach can be easily generalised to tighter relaxations such as the LP.

Wang et al. \cite{20} derived a semi-definite programming relaxation of the energy minimisation problem, allowing them to reach better energy than mean-field. Their approach has the advantage of not being restricted to Gaussian pairwise potentials. The inference is made feasible by performing low-rank approximation of the Gram matrix of the kernel, instead of using the filter-based method. However, while the complexity of their algorithm is the same as our QP or DC relaxation, the runtime is significantly higher. Furthermore, while the SDP
relaxation has been shown to be accurate for repulsive pairwise potentials (encouraging neighbouring variables to take different labels) [21], our LP relaxation provides the best guarantees for attractive pairwise potentials [2].

In this paper, we use the same filter-based method [8] as the one employed in mean-field. We build on it to solve continuous relaxations of the original problem that have both convergence and quality guarantees. Our work can be viewed as a complementary direction to previous research trends in dense CRFs. While [10,11,12] improved mean-field and [17,18] learnt the parameters, we focus on the energy minimisation problem.

3 Preliminaries

Before describing our methods for energy minimisation on dense CRF, we establish the necessary notation and background information.

Dense CRF Energy Function. We define a dense CRF on a set of \( N \) random variables \( X = \{X_1, \ldots, X_N\} \) each of which can take one label from a set of \( M \) labels \( L = \{l_1, \ldots, l_M\} \). To describe a labelling, we use a vector \( x \) of size \( N \) such that its element \( x_a \) is the label taken by the random variable \( X_a \). The energy associated with a given labelling is defined as:

\[
E(x) = \sum_{a=1}^{N} \phi_a(x_a) + \sum_{a=1}^{N} \sum_{\substack{b=1 \\ b \neq a}}^{N} \psi_{a,b}(x_a, x_b). \tag{1}
\]

Here, \( \phi_a(x_a) \) is called the unary potential for the random variable \( X_a \) taking the label \( x_a \). The term \( \psi_{a,b}(x_a, x_b) \) is called the pairwise potential for the random variables \( X_a \) and \( X_b \) taking the labels \( x_a \) and \( x_b \) respectively. The energy minimisation problem on this CRF can be written as:

\[
x^* = \arg\min_{x} E(x). \tag{2}
\]

Gaussian Pairwise Potentials. Similar to previous work [5], we consider arbitrary unary potentials and Gaussian pairwise potentials. Specifically, the form of the pairwise potentials is given by:

\[
\psi_{a,b}(i, j) = \mu(i, j) \sum_m w^{(m)} k(f^{(m)}_a, f^{(m)}_b), \tag{3}
\]

\[
k(f_a, f_b) = \exp \left( -\frac{\|f_a - f_b\|^2}{2} \right) \tag{4}
\]

We refer to the term \( \mu(i, j) \) as a label compatibility function between the labels \( i \) and \( j \). An example of a label compatibility function is the Potts model, where \( \mu_{\text{potts}}(i, j) = [i \neq j] \), that is \( \mu_{\text{potts}}(i, j) = 1 \) if \( i \neq j \) and 0 otherwise. Note that the label compatibility does not depend on the image. The other term, called the pixel compatibility function, is a mixture of gaussian kernels \( k(\cdot, \cdot) \). The coefficients of the mixture are the weights \( w^{(m)} \). The \( f^{(m)}_a \) are the features describing the random variable \( X_a \). Note that the pixel compatibility does not
depend on the labelling. In practice, similar to [5], we use the position and RGB values of a pixel as features.

**IP Formulation.** We now introduce a formulation of the energy minimisation problem that is more amenable to continuous relaxations. Specifically, we formulate it as an Integer Program (IP) and then relax it to obtain a continuous optimisation problem. To this end, we define the vector \( y \) whose components \( y_a(i) \) are indicator variables specifying whether or not the random variable \( X_a \) takes the label \( i \). Using this notation, we can rewrite the energy minimisation problem as an IP:

\[
\min \sum_{a=1}^{N} \sum_{i \in \mathcal{L}} \phi_a(i)y_a(i) + \sum_{a=1}^{N} \sum_{b \neq a}^{N} \sum_{i,j \in \mathcal{L}} \psi_{a,b}(i,j)y_a(i)y_b(j),
\]

\[
\text{s.t.} \sum_{i \in \mathcal{L}} y_a(i) = 1 \quad \forall a \in [1,N],
\]

\[
y_a(i) \in \{0,1\} \quad \forall a \in [1,N] \quad \forall i \in \mathcal{L}.
\]

(5)

The first set of constraints model the fact that each random variable has to be assigned exactly one label. The second set of constraints enforce the optimisation variables \( y_a(i) \) to be binary. Note that the objective function is equal to the energy of the labelling encoded by \( y \).

**Filter-based Method.** Similar to [5], a key component of our algorithms is the filter-based method of Adams et al. [8]. It computes the following operation:

\[
\forall a \in [1,N], \quad v'_a = \sum_{b=1}^{N} k(f_a, f_b)v_b,
\]

(6)

where \( v'_a, v_b \in \mathbb{R} \) and \( k(\cdot, \cdot) \) is a Gaussian kernel. Performing this operation the naïve way would result in computing a sum on \( N \) elements for each of the \( N \) terms that we want to compute. The resulting complexity would be \( \mathcal{O}(N^2) \). The filter-based method allows us to perform it approximately with \( \mathcal{O}(N) \) complexity. We refer the interested reader to [8] for details. The accuracy of the approximation made by the filter-based method is explored in the Appendix A.

## 4 Quadratic Programming Relaxation

We are now ready to demonstrate how the filter-based method [8] can be used to optimise our first continuous relaxation, namely the convex quadratic programming (QP) relaxation.

**Notation.** In order to concisely specify the QP relaxation, we require some additional notation. Similar to [10], we rewrite the objective function with linear algebra operations. The vector \( \phi \) contains the unary terms. The matrix \( \mu \) corresponds to the label compatibility function. The Gaussian kernels associated with the \( m \)-th features are represented by their Gram matrix \( \mathbf{K}_{a,b}^{(m)} = k(f_a^{(m)}, f_b^{(m)}) \). The Kronecker product is denoted by \( \otimes \). The matrix \( \Psi \) represents the pairwise
Desmaison, Bunel, Kohli, Torr, Kumar

6

The terms and is defined as follows:

$$
\Psi = \mu \otimes \left( \sum_{m} K^{(m)} - I_N \right),
$$

where $I_N$ is the identity matrix. Under this notation, the IP (5) can be concisely written as

$$
\min \phi^T y + y^T \Psi y,
$$

s.t. $y \in \mathcal{I}$, (8)

with $\mathcal{I}$ being the feasible set of integer solution, as defined in equation (5).

Relaxation. In general, IP such as (8) are NP-hard problems. Relaxing the integer constraint on the indicator variables to allow fractional values between 0 and 1 results in the QP formulation. Formally, the feasible set of our minimisation problem becomes:

$$
M = \left\{ y \text{ such that } \sum_{i \in \mathcal{L}} y_a(i) = 1 \quad \forall a \in [1, N], \\
y_a(i) \geq 0 \quad \forall a \in [1, N], \forall i \in \mathcal{L} \right\}.
$$

(9)

Ravikumar and Lafferty \cite{1} showed that this relaxation is tight and that solving the QP will result in solving the IP. However, this QP is still NP-hard, as the objective function is non-convex. To alleviate this difficulty, Ravikumar and Lafferty \cite{1} relaxed the QP minimisation to the following convex problem:

$$
\min S_{cvx}(y) = (\phi - d)^T y + y^T (\Psi + D)y,
$$

s.t. $y \in M$, (10)

where the vector $d$ is defined as follows

$$
d_a(i) = \sum_{b=1}^{N} \sum_{\substack{j \in \mathcal{L} \\ b \neq a}} |\psi_{a,b}(i, j)|,
$$

and $D$ is the square diagonal matrix with $d$ as its diagonal.

Minimisation. We now introduce a new method based on the Frank-Wolfe algorithm \cite{9} to minimise problem (23). The Frank-Wolfe algorithm allows to minimise a convex function $f$ over a convex feasible set $M$. The key steps of the algorithm are shown in Algorithm 1. To be able to use the Frank-Wolfe algorithm, we need a way to compute the gradient of the objective function (Step 3), a method to compute the conditional gradient (Step 4) and a strategy to choose the step size (Step 5).

Gradient computation

Since the objective function is quadratic, its gradient can be computed as

$$
\nabla S_{cvx}(y) = (\phi - d) + 2(\Psi + D)y.
$$

(12)

What makes this equation expensive to compute in a naive way is the matrix product with $\Psi$. We observe that this operation can be performed using the
Algorithm 1 Frank-Wolfe algorithm

1: Get $y^0 \in \mathcal{M}$
2: while not converged do
3:  Compute the gradient at $y^t$ as $g = \nabla f(y^t)$
4:  Compute the conditional gradient as $s = \arg\min_{s \in \mathcal{M}} \langle s, g \rangle$
5:  Compute a step-size $\alpha = \arg\min_{\alpha \in [0,1]} f(\alpha y^t + (1 - \alpha) s)$
6:  Move towards the negative conditional gradient $y^{t+1} = \alpha y^t + (1 - \alpha) s$
7: end while

filter-based method in linear time. Note that the other matrix-vector product, $Dy$, is not expensive (linear in $N$) since $D$ is a diagonal matrix.

Conditional gradient

The conditional gradient is obtained by solving

$$\arg\min_{s \in \mathcal{M}} \langle s, \nabla S_{cvx}(y) \rangle. \quad (13)$$

Minimising such an LP would usually be an expensive operation for problems of this dimension. However, we remark that, once the gradient has been computed, exploiting the properties of our problem allows us to solve problem (13) in a time linear in the number of random variables ($N$) and labels ($M$). Specifically, the following is an optimal solution to problem (13).

$$s_a(i) = \begin{cases} 1 & \text{if } i = \arg\min_{i \in \mathcal{L}} \frac{\partial S_{cvx}}{\partial y_a(i)} \\ 0 & \text{else.} \end{cases} \quad (14)$$

Step size determination

In the original Frank-Wolfe algorithm, the step size $\alpha$ is simply chosen using line search. However we observe that, in our case, the optimal $\alpha$ can be computed by solving a second-order polynomial function of a single variable, which has a closed form solution that can be obtained efficiently. This observation has been previously exploited in the context of Structural SVM [22]. The derivations for this closed form solution can be found in Appendix B. With careful reutilisation of computations, this step can be performed without additional filter-based method calls. By choosing the optimal step size at each iteration, we reduce the number of iterations needed to reach convergence.

The above procedure converges to the global minimum of the convex relaxation and resorts to the filter-based method only once per iteration during the computation of the gradient and is therefore efficient. However, this solution has no guarantees to be even a local minimum of the original QP relaxation. To alleviate this, we will now introduce a difference-of-convex (DC) relaxation.

5 Difference of Convex Relaxation

5.1 DC relaxation: General case

The objective function of a general DC program can be specified as

$$S_{CCCP}(y) = p(y) - q(y). \quad (15)$$
One can obtain one of its local minima using the Concave-Convex Procedure (CCCP) \cite{23}. The key steps of this algorithm are described in Algorithm \ref{alg:cccp}. Briefly, Step \ref{step:linearize_concave} computes the gradient of the concave part. Step \ref{step:convex_upper_bound} minimises a convex upper bound on the DC objective, which is tight at $y^t$.

In order to exploit the CCCP algorithm for DC programs, we observe that the QP (8) can be rewritten as

$$\min_y \phi^T y + y^T (\Psi + D)y - y^T Dy,$$

s.t. $y \in M$. \hfill (16)

Formally, we can define $p(y) = \phi^T y + y^T (\Psi + D)y$ and $q(y) = y^T Dy$, which are both convex in $y$.

\begin{algorithm}
\caption{CCCP Algorithm}
1: Get $y^0 \in M$
2: \textbf{while} not converged \textbf{do}
3: \hspace{1em} Linearise the concave part $g = \nabla q(y^t)$
4: \hspace{1em} Minimise a convex upper-bound $y^{t+1} = \arg\min_{y \in M} p(y) - g^T y$
5: \textbf{end while}
\end{algorithm}

We observe that, since $D$ is diagonal and the matrix product with $\Psi$ can be computed using the filter based method, the gradient $\nabla q(y^t) = 2Dy$ (Step \ref{step:linearize_concave}) is efficient to compute. The minimisation of the convex problem (Step \ref{step:convex_upper_bound}) is analogous to the convex QP formulation \cite{23} presented above with different unary potentials. Since we do not place any restrictions on the form of the unary potentials, (Step \ref{step:convex_upper_bound}) can be implemented using the method described in Section \ref{sec:filter_based_method}.

The CCCP algorithm provides a monotonous decrease in the objective function and will converge to a local minimum \cite{24}. However, the above method will take several iterations to converge, each necessitating the solution of a convex QP, and thus requiring multiple calls to the filter-based method. While the filter-based method \cite{8} allows us to compute operations on the pixel compatibility function in linear time, it still remains an expensive operation to perform. As we show next, if we introduce some additional restriction on our potentials, we can obtain a more efficient difference of convex decomposition.

### 5.2 DC relaxation: negative semi-definite compatibility

We now introduce a new DC relaxation of our objective function that takes advantage of the structure of the problem. Specifically, the convex problem to solve at each iteration does not depend on the filter-based method computations, which are the expensive steps in the previous method. Following the example of Krähenbühl and Koltun \cite{10}, we look at the specific case of negative semi-definite label compatibility function, such as the commonly used Potts model. Taking advantage of the specific form of our pairwise terms \cite{7}, we can rewrite the problem as

$$S(y) = \phi^T y - y^T (\mu \otimes I_N)y^T + y^T (\mu \otimes \sum_m K^{(m)})y.$$ \hfill (17)
The first two terms can be verified as being convex. The Gaussian kernel is positive semi-definite, so the Gram matrices $K^{(m)}$ are positive semi-definite. By assumption, the label compatibility function is also negative semi-definite. The results from the Kronecker product between the Gram matrix and $\mu$ is therefore negative semi-definite.

**Minimisation.** Once again we use the CCCP Algorithm. The main difference between the generic DC relaxation and this specific one is that Step 3 now requires a call to the filter-based method, while the iterations required to solve Step 4 do not. In other words, each iteration of CCCP only requires one call to the filter based method. This results in a significant improvement in speed. More details about this operation are available in Appendix C.

### 6 LP relaxation

This section presents an accurate LP relaxation of the energy minimisation problem and our method to optimise it efficiently using subgradient descent.

**Relaxation.** To simplify the description, we focus on the Potts model. However, our approach can easily be extended to more general pairwise potentials by approximating them using a hierarchical Potts model. Such an extension, inspired by [25], is presented in Appendix F. We define the following notation:

$$K_{a,b} = \sum_m w^{(m)}_a k^{(m)}(f^{(m)}_a, f^{(m)}_b), \quad \sum_a = \sum_{a=1}^{N_a} \quad \text{and} \quad \sum_{b<a} = \sum_{b=1}^{a-1}.$$

With these notations, a LP relaxation of (5) is:

$$\min \quad S_{LP}(y) = \sum_a \sum_i \phi_a(i)y_a(i) + \sum_a \sum_{b \neq a} \sum_i K_{a,b} \frac{|y_a(i) - y_b(i)|}{2},$$

s.t. $y \in M$. 

The feasible set remains the same as the one we had for the QP and DC relaxations. In the case of integer solutions, $S_{LP}(y)$ has the same value as the objective function of the IP described in (5). The unary term is the same for both formulations. The pairwise term ensures that for every pair of random variables $X_a, X_b$, we add the cost $K_{a,b}$ associated with this edge only if they are not associated with the same labels.

**Minimisation.** Kleinberg and Tardos [2] solve this problem by introducing extra variables for each pair of pixels to get a standard LP, with a linear objective function and linear constraints. In the case of a dense CRF, this is infeasible because it would introduce a number of variables quadratic in the number of pixels. We will instead use projected subgradient descent to solve this LP. To do so, we will reformulate the objective function, derive the subgradient, and present an algorithm to compute it efficiently.

**Reformulation**

The absolute value in the pairwise term of (5) prevents us from using the filtering approach. To address this issue, we consider that for any given label $i$, the
variables $y_a(i)$ can be sorted in a descending order: $a \geq b \implies y_a(i) \leq y_b(i)$. This allows us to rewrite the pairwise term of the objective function (18) as:

$$\sum_i \sum_a \sum_{a \neq b} K_{a,b} \frac{|y_a(i) - y_b(i)|}{2} = \sum_i \sum_a \sum_{b > a} K_{a,b} y_a(i) - \sum_i \sum_a \sum_{b < a} K_{a,b} y_a(i).$$

(19)

A formal derivation of this equality can be found in Appendix D.

**Subgradient**

From (19), we rewrite the subgradient:

$$\frac{\partial S_{LP}}{\partial y_c(k)}(y) = \phi_c(k) + \sum_{a > c} K_{a,c} - \sum_{a < c} K_{a,c}. \tag{20}$$

Note that in this expression, the dependency on the variable $y$ is hidden in the bounds of the sum because we assumed that $y_a(k) \leq y_c(k)$ for all $a > c$. For a different value of $y$, the elements of $y$ would induce a different ordering and the terms involved in each summation would not be the same.

**Subgradient computation**

What prevents us from evaluating (20) efficiently are the two sums, one over an upper triangular matrix ($\sum_{a > c} K_{a,c}$) and one over a lower triangular matrix ($\sum_{a < c} K_{a,c}$). As opposed to (6), which computes terms $\sum_{a,b} K_{a,b} v_b$ for all $a$ using the filter-based method, the summation bounds here depend on the random variable we are computing the partial derivative for. While it would seems that the added sparsity provided by the upper and lower triangular matrices would simplify the operation, it is this sparsity itself that prevents us from interpreting the summations as convolution operations. Thus, we cannot use the filter-based method as described by Adams et al. [8].

We alleviate this difficulty by designing a novel divide-and-conquer algorithm. We describe our algorithm for the case of the upper triangular matrix. However, it can easily be adapted to compute the summation corresponding to the lower triangular matrix. We present the intuition behind the algorithm using an example. A rigorous development can be found in Appendix E. If we consider $N = 6$ then $a, c \in \{1, 2, 3, 4, 5, 6\}$ and the terms we need to compute for a given label are:

$$\begin{pmatrix}
\sum_{a > 1} K_{a,1} \\
\sum_{a > 2} K_{a,2} \\
\sum_{a > 3} K_{a,3} \\
\sum_{a > 4} K_{a,4} \\
\sum_{a > 5} K_{a,5} \\
\sum_{a > 6} K_{a,6}
\end{pmatrix} = \begin{pmatrix}
0 & K_{2,1} & K_{3,1} & K_{4,1} & K_{5,1} & K_{6,1} \\
0 & 0 & K_{3,2} & K_{4,2} & K_{5,2} & K_{6,2} \\
0 & 0 & 0 & K_{4,3} & K_{5,3} & K_{6,3} \\
0 & 0 & 0 & 0 & K_{5,4} & K_{6,4} \\
0 & 0 & 0 & 0 & 0 & K_{6,5} \\
0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix} \cdot \begin{pmatrix}
1 \\
1 \\
1 \\
1 \\
1 \\
1
\end{pmatrix}. \tag{21}$$

We propose a divide and conquer approach that solves this problem by splitting the upper triangular matrix $U$. The top-left and bottom-right parts are upper triangular matrices with half the size. We solve these subproblems recursively.
The top-right part can be computed with the original filter based method. Using this approach, the total complexity to compute this sum is $O(N \log(N))$.

With this algorithm, we have made feasible the computation of the subgradient. We can therefore perform projected subgradient descent on the LP objective efficiently. Since we need to compute the subgradient for each label separately due to the necessity of having sorted elements, the complexity associated with taking a gradient step is $O(MN \log(N))$. To ensure the convergence, we choose as learning rate $(\beta^t)_{t=1}^\infty$ that is a square summable but not a summable sequence such as $(\frac{1}{1+t})_{t=1}^\infty$. We also make use of the work by Condat [26] to perform fast projection on the feasible set. The complete procedure can be found in Algorithm 3. Step 3 to 7 present the subgradient computation for each label. Using this subgradient, Step 8 shows the update rule for $y^t$. Finally, Step 9 project this new estimate onto the feasible space.

Algorithm 3  LP subgradient descent

1: Get $y^0 \in \mathcal{M}$
2: while not converged do
3: for $i \in \mathcal{L}$ do
4: Sort $y_a(i)$ \quad $\forall a \in [1, N]$
5: Reorder $K$
6: $g(i) = \nabla S_{LP}(y^t(i))$
7: end for
8: $y^{t+1} = y^t - \beta^t \cdot g$
9: Project $y^{t+1}$ on the feasible space
10: end while

The algorithm that we introduced converges to a global minimum of the LP relaxation. By using the rounding procedure introduced by Kleinberg and Tardos [2], it has a multiplicative bound of 2 for the dense CRF labelling problem on Potts models and $O(\log(M))$ for metric pairwise potentials.

7 Experiments

We now demonstrate the benefits of using continuous relaxations of the energy minimisation problem on two applications: stereo matching and semantic segmentation. We provide results for the following methods: the Convex QP relaxation ($\text{QP}_{\text{cvx}}$), the generic and negative semi-definite specific DC relaxations ($\text{DC}_{\text{gen}}$ and $\text{DC}_{\text{neg}}$) and the LP relaxation ($\text{LP}$). We compare solutions obtained by our methods with the mean-field baseline ($\text{MF}$).

7.1 Stereo matching

Data. We compare these methods on images extracted from the Middlebury stereo matching dataset [27]. The unary terms are obtained using the absolute difference matching function of [27]. The pixel compatibility function is similar to the one used by Krähenbühl and Koltun [5] and is described in Appendix G. The label compatibility function is a Potts model.
Results. We present a comparison of runtime in Figure (1a), as well as the associated final energies for each method in Table (1b). Similar results for other problem instances can be found in Appendix H.

![Runtime comparisons](image)

(a) Runtime comparisons

![Final Energy achieved](image)

(b) Final Energy achieved

Fig. 1: Evolution of achieved energies as a function of time on a stereo matching problem (Teddy Image). While the QP method leads to the worst result, using it as an initialisation greatly improves results. In the case of negative semi-definite potentials, the specific DC\(_{neg}\) method is as fast as mean-field, while additionally providing guarantees of monotonous decrease. (Best viewed in colour)

We observe that continuous relaxations obtain better energies than their mean-field counterparts. For a very limited time-budget, MF is the fastest method, although DC\(_{neg}\) is competitive and reach lower energies. When using LP, optimising a better objective function allows us to escape the local minima to which DC\(_{neg}\) converges. However, due to the higher complexity and the fact that we need to perform divide-and-conquer separately for all labels, the method is slower. This is particularly visible for problems with a high number of labels. This indicates that the LP relaxation might be better suited to fine-tune accurate solutions obtained by faster alternatives. For example, this can be achieved by restricting the LP to optimise over a subset of relevant labels, that is, labels that are present in the solutions provided by other methods. Qualitative results for the Teddy image can be found in Figure 2 and additional outputs are present in Appendix H. We can see that lower energy translates to better visual results: note the removal of the artifacts in otherwise smooth regions (for example, in the middle of the sloped surface on the left of the image).

7.2 Image Segmentation

Data. We now consider an image segmentation task evaluated on the PASCAL VOC 2010 [28] dataset. For the sake of comparison, we use the same data splits and unary potentials as the one used by Krähenbühl and Koltun [5]. We perform
Fig. 2: Stereo matching results on the Teddy image. Continuous relaxation achieve smoother labeling, as expected by their lower energies.

cross-validation to select the best parameters of the pixel compatibility function for each method using Spearmint [29].

**Results.** The energy results obtained using the parameters cross validated for \( \text{DC}_{\text{neg}} \) are given in Table 1. \( \text{MF5} \) corresponds to mean-field ran for 5 iterations as it is often the case in practice [5,12].

| Unary | MF5 | MF | \( \text{QP}_{\text{cvx}} \) | \( \text{DC}_{\text{gen}} \) | \( \text{DC}_{\text{neg}} \) | LP | Avg. E | Acc  | IoU  |
|-------|-----|----|-----------------|-----------------|-----------------|----|-------|------|------|
| Unary | -   | 0  | 0               | 0               | 0               | 0  | 0     | 79.04| 27.43|
| MF5   | 99  | -  | 13              | 0               | 0               | 0  | -600  | 79.13| 27.53|
| MF    | 99  | 0  | -               | 0               | 0               | 0  | -600  | 79.13| 27.53|
| \( \text{QP}_{\text{cvx}} \) | 99  | 99 | 99              | -               | 0               | 0  | -6014 | 80.38| 28.56|
| \( \text{DC}_{\text{gen}} \) | 99  | 99 | 99              | 85              | -               | 0  | 1     | 80.41| 28.59|
| \( \text{DC}_{\text{neg}} \) | 99  | 99 | 99              | 98              | 97              | -  | 4     | 80.43| 28.60|
| LP    | 99  | 99 | 99              | 98              | 97              | 87 | -8697 | 80.49| 28.68|

Table 1: Percentage of images the row method outperforms the column method on final energy, average energy over the test set and Segmentation performance. Continuous relaxations dominate mean-field approaches on almost all images and improve significantly more compared to the Unary baseline. Parameters tuned for \( \text{DC}_{\text{neg}} \).

Once again, we observe that continuous relaxations provide lower energies than mean-field based approaches. To add significance to this result, we also compare energies image-wise. In all but a few cases, the energies obtained by the continuous relaxations are better or equal to the mean-field ones. This provides conclusive evidence for our central hypothesis that continuous relaxations are better suited to the problem of energy minimisation in dense CRFs.

For completeness, we also provide energy and segmentation results for the parameters tuned for \( \text{MF} \) in Appendix I. Even in that unfavourable setting, continuous relaxations still provide better energies. Note that, due to time con-
straints, we run the LP subgradient descent for only 5 iterations of subgradient descent. Moreover, to be able to run more experiments, we also restricted the number of labels by discarding labels that have a very small probability to appear given the initialisation.

Some qualitative results can be found in Figure 3. When comparing the segmentations for MF and DC\textsubscript{neg}, we can see that the best one is always the one we tune parameters for. A further interesting caveat is that although we always find a solution with better energy, it does not appear to be reflected in the quality of the segmentation. While in the previous case with stereo vision, better energy implied qualitatively better reconstruction it is not so here. Similar observation was made by Wang et al [20].

8 Discussion

Our main contribution are four efficient algorithms for the dense CRF energy minimisation problem based on QP, DC and LP relaxations. We showed that continuous relaxations give better energies than the mean-field based approaches. Our best performing method, the LP relaxation, suffers from its high runtime. To go beyond this limit, move making algorithms such as $\alpha$-expansion [30] could be used and take advantage of the fact that this relaxation solves exactly the original IP for the two label problem. In future work, we also want to investigate the effect of learning specific parameters for these new inference methods using the framework of [18].

Acknowledgments

This work was supported by the EPSRC, Leverhulme Trust, Clarendon Fund and the ERC grant ERC-2012-AdG 321162-HELIOS, EPSRC/MURI grant ref EP/N019474/1, EPSRC grant EP/M013774/1, EPSRC Programme Grant Seebibyte EP/M013774/1 and Microsoft Research PhD Scolarship Program. We thank Philip Krähenbühl for making his code available and Vibhav Vineet for his help.
References

1. Ravikumar, P., Lafferty, J.: Quadratic programming relaxations for metric labeling and Markov random field MAP estimation. In: ICML. (2006)
2. Kleinberg, J., Tardos, E.: Approximation algorithms for classification problems with pairwise relationships: Metric labeling and Markov random fields. JACM (2002)
3. Kumar, P., Kolmogorov, V., Torr, P.: An analysis of convex relaxations for MAP estimation. In: NIPS. (2008)
4. Chekuri, C., Khanna, S., Naor, J., Zosin, L.: Approximation algorithms for the metric labeling problem via a new linear programming formulation. In: SODA. (2001)
5. Krähenbühl, P., Koltun, V.: Efficient inference in fully connected CRFs with gaussian edge potentials. In: NIPS. (2011)
6. Tappen, M., Liu, C., Adelson, E., Freeman, W.: Learning gaussian conditional random fields for low-level vision. In: CVPR. (2007)
7. Koller, D., Friedman, N.: Probabilistic graphical models: principles and techniques. (2009)
8. Adams, A., Baek, J., Abraham, M.: Fast high-dimensional filtering using the permutohedral lattice. Eurographics (2010)
9. Frank, M., Wolfe, P.: An algorithm for quadratic programming. Naval research logistics quarterly (1956)
10. Krähenbühl, P., Koltun, V.: Parameter learning and convergent inference for dense random fields. In: ICML. (2013)
11. Baqué, P., Bagautdinov, T., Fleuret, F., Fua, P.: Principled parallel mean-field inference for discrete random fields. In: CVPR. (2016)
12. Vineet, V., Warrell, J., Torr, P.: Filter-based mean-field inference for random fields with higher-order terms and product label-spaces. IJCV (2014)
13. Ladicky, L., Russell, C., Kohli, P., Torr, P.: Graph cut based inference with co-occurrence statistics. In: ECCV. (2010)
14. Kohli, P., Kumar, P., Torr, P.: P3 & beyond: Solving energies with higher order cliques. In: CVPR. (2007)
15. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: CVPR. (2015)
16. Chen, L., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.: Semantic image segmentation with deep convolutional nets and fully connected CRFs. In: ICLR. (2015)
17. Schwing, A., Urtasun, R.: Fully connected deep structured networks. CoRR (2015)
18. Zheng, S., Jayasumana, S., Romera-Paredes, B., Vineet, V., Su, Z., Du, D., Huang, C., Torr, P.: Conditional random fields as recurrent neural networks. In: ICCV. (2015)
19. Zhang, Y., Chen, T.: Efficient inference for fully-connected CRFs with stationarity. In: CVPR. (2012)
20. Wang, P., Shen, C., van den Hengel, A.: Efficient SDP inference for fully-connected CRFs based on low-rank decomposition. In: CVPR. (2015)
21. Goemans, M., Williamson, D.: Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. JACM (1995)
22. Lacoste-Julien, S., Jaggi, M., Schmidt, M., Pletscher, P.: Block-coordinate frank-wolfe optimization for structural svms. In: ICML. (2013)
23. Yuille, A., Rangarajan, A.: The concave-convex procedure (CCCP). NIPS (2002)
9 Appendix

A Filter-based method approximation

In this paper, the filter based method that we use for our experiments is the one by Adams et al. [8]. In this method, the original computation is approximated by a convolution in a higher dimensional space. The original points are associated to a set of vertices on which the convolution is performed. The considered vertices are the one from the permutohedral lattice. Krähenbühl and Koltun [5] provided an implementation of this method. In their implementation, they added a pixel-wise normalisation of the output of the permutohedral lattice and say that it performs well in practice.

We observe that for the variances considered in this paper and without using the normalisation by Krähenbühl and Koltun, the results given by the permutohedral lattice is a constant factor away from the value computed by brute force in most cases. As can be seen in Figure 4, in the case where we compute \( \sum_{a,b} K_{a,b} \), the left graph, the ratio between the value obtained by brute force and the value obtained using the permutohedral lattice is 0.6 for large enough images. On the other hand, for a different value of the input points where we compute \( \sum_{b>a} K_{a,b} - \sum_{b<a} K_{a,b} \), the right graph, we get a ratio of 0.48 between the two results. The case where we consider a variance of 50 is special. We know that the highest the variance value, the worst the approximation of the permutohedral is. If the experience on the full computation is conducted on an image of size \( 320 \times 213 \), the ratio between the brute force approach and the permutohedral lattice is 0.633. At the same time is also worth noting that in all these results, if we consider the outputs as vectors, as is done when computing our gradients, the vectors given by the brute force and the ones given by the permutohedral lattice are collinear for all image size and all variances. We can thus expect that for other input values, the direction of gradient provided by the permutohedral lattice is correct, but the norm of this vector may be incorrect.
B Optimal step size in the Frank-Wolfe algorithm

Solving the convex relaxation of the QP is performed using the Frank-Wolfe algorithm \[9\]. The gradient is computed efficiently using the filter-based method \[8\]. An efficient method is available to compute the conditional gradient, based on the gradient. Once this conditional gradient is obtained, a step size needs to be determined to update the value of the current parameters. We show that the optimal step size can be computed and that this does not introduce any additional call to the filter-based method.

The problem to solve is

$$
\arg\min_{\alpha \in [0,1]} S_{\text{cvx}}(y + \alpha(s - y)).
$$

The definition of $S_{\text{cvx}}$ is

$$
S_{\text{cvx}}(y) = (\phi - d)^T y + y^T (\Psi + D)y.
$$

Solving for the optimal value of $\alpha$ amounts to solving a second order polynomial:

$$
S_{\text{cvx}}(y + \alpha(s - y)) = (\phi - d)^T (y + \alpha(s - y))
+ (y + \alpha(s - y))^T (\Psi + D)(y + \alpha(s - y)),
= \alpha^2 \left[ (s - y)^T (\Psi + D)(s - y) \right]
+ \alpha \left[ (\phi - d)^T (s - y) + 2y^T (\Psi + D)(s - y) \right]
+ \left[ (\phi - d)^T y + y^T (\Psi + D)y \right],
$$

whose optimal value is given by

$$
\alpha^* = -\frac{1}{2} \frac{(\phi - d)^T (s - y) + 2y^T (\Psi + D)(s - y)}{(s - y)^T (\Psi + D)(s - y)}
$$

The dot products are going to be linear in complexity and efficient. Using the filtering approach, the matrix-vector operation are also linear in complexity. In terms of run-time, they represent the costliest step so minimising the number of times that we are going to perform them will gives us the best performance for
our algorithm. We remind the reader that the expression of the gradient used at an iteration is:

$$\nabla S_{cvx}(y) = (\phi - d) + 2(\Psi + D)y. \quad (26)$$

so by keeping intermediary results of the gradient’s computation, we don’t need to compute $$(\Psi + D)y$$, having already performed this operation once. The other matrix-vector product that is necessary for obtaining the optimal step-size is $$(\Psi + D)s$$. During the first iteration, we will need to compute is using filter-based methods, which means using them twice in the same iteration. However, this computation can be reused. The update rules that we follow are:

$$y^{t+1} = y^t + \alpha(s - y^t). \quad (27)$$

At the following iteration, to obtain the gradient, we will need to compute:

$$(\Psi + D)y^{t+1} = (\Psi + D)(y^t + \alpha(s - y^t)),$$

$$= (1 - \alpha)(\Psi + D)y^t + \alpha(\Psi + D)s. \quad (28)$$

All the matrix-vector product of this equation have already been computed. This means that no call to the filter-based method will be required.

At each iteration, we will only need to perform the expensive matrix-vector products on the conditional gradient. Using linearity and keeping track of our previous computations, we can then obtain all the other terms that we need.

### C Convex problem in the restricted DC relaxation

Two difference-of-convex decompositions of the objective function are presented in the paper. The first one is based on diagonally dominant matrices to ensure convexity and would be applicable to any QP objective function. However, using this decomposition, the terms involving the pixel-compatibility function, and therefore requiring filter-based convolutions, need to be computed several times per CCCP iteration.

On the other hand, in the case of negative semi-definite compatibility functions, a decomposition suited to the structure of the problem is available. Using this decomposition, similar to the one proposed by Krähenbühl [10], the convex problem to solve CCCP will be the following:

$$\min (\phi^T - g^T)y - y^T(\mu \otimes I_N)y,$$

s.t. $$y \in \mathcal{M}. \quad (29)$$

The filter-method has been used to compute the gradient of the concave part $g$. The Kronecker product with the identity matrix will make this problem completely de-correlated between pixels. This means that instead of solving one problem involving $N \times L$ variables, we will have to solve $N$ problems of $L$ variables, which is much faster. The problem to solve for each pixel are, using the $a$ subscript to refer to the subset of the vector elements that correspond to the
random variable \( a \):
\[
\min \ (\phi^T_a - g^T_a) y_a - y^T_a \mu, \\
\text{s.t.} \quad y_a \geq 0 \\
\quad y^T_a 1 = 1.
\] (30)

These problems can also be solved using the Frank-Wolfe algorithm, with efficient conditional gradient computation and optimal step size. The only difference is that in that case, no filter-based method will need to be used for computation. CCCP on this DC relaxation will therefore be much faster than on the generic case, an improvement gained at the cost of generality.

We also remark that the guarantees of CCCP to provide better results at each iteration does not require to solve the convex problem exactly. It is sufficient to obtain a value of the convex problem lower that the initial estimate. Therefore, the inference may eventually be sped-up by solving the convex problem approximately instead of reaching the optimal solution.

D LP objective reformulation

This section presents the reformulation of the pairwise part of the LP objective. We first introduce the following equality:
\[
\sum_a \sum_{b > a} K_{a,b} y_b(i) = \sum_a \sum_{b < a} K_{a,b} y_a(i), \tag{31}
\]

It comes from the symmetry of \( K \).

Using the above formula, considering the reordering has already been done, we can rewrite the pairwise term of (18) as:
\[
\sum_a \sum_{b \neq a} \sum_i K_{a,b} \frac{|y_a(i) - y_b(j)|}{2}, \\
= \sum_i \sum_a \sum_{b > a} K_{a,b} \frac{y_a(i) - y_b(i)}{2} - \sum_i \sum_a \sum_{b < a} K_{a,b} \frac{y_a(i) - y_b(i)}{2}, \tag{32}
\]

It is important to note that in these equations, the ordering between \( a \) and \( b \) used in the summations is dependent on the considered label \( i \).

E LP Divide and conquer

We are going to present an algorithm to efficiently compute the following:
\[
\forall k \sum_{j > k} K_{k,j} , \tag{33}
\]

for \( j \) and \( k \) being between 1 and \( N \). For the sake of simplicity, we are going to consider \( N \) as being even. The odd case is very similar. Considering \( h = N/2 \), we can rewrite the original sum as:
\[
\sum_{j > k} K_{k,j} = \begin{cases} 
\sum_{j > k} K_{k,j} & \text{if } k > h \\
\sum_{j > k} K_{k,j} & \text{if } k \leq h 
\end{cases}
\]

\[
= \begin{cases} 
\sum_{j > k} K_{k,j} & \text{if } k > h \\
\sum_{j > k} K_{k,j} + \sum_{j > h} K_{k,j} & \text{if } k \leq h 
\end{cases}
\]

(34)

We can see that both \(A\) and \(B\) corresponds to the cases where respectively \(k, j > h\) and \(k, j \leq h\). These two elements can be obtained by recursion using sub-matrices of \(K\) which have half the size of the current size of the problem. To compute the \(C\) part, we consider the following variable:

\[
v_j = \begin{cases} 
0 & \text{if } j \leq h \\
1 & \text{if } j > h 
\end{cases}
\]

(35)

We can now rewrite the \(C\) part as \(\sum_j K_{k,j}v_j\). We can compute this sum efficiently for all \(k\) using the filter-based method. Since this term contribute to the original sum only when \(k \leq h\), we will only consider a subset of the output from the filter based method.

So we have a recursive algorithm that will have a depth of \(\log(N)\) and for which all level takes \(\mathcal{O}(N)\) to compute. We can use it to compute the requested sum \(\forall k\) in \(\mathcal{O}(N \log(N))\).

**F LP generalisation beyond Potts models**

In this section, we consider the case where the label compatibility \(\mu(x_a, x_b)\) is any semi-metric. We recall that \(\mu(\cdot, \cdot)\) is a semi-metric if and only if \(d(i, i) = 0, \forall i\) and \(d(i, j) = d(j, i) > 0, \forall i \neq j\).

To solve this problem, we are going to reduce the semi-metric labelling problem to a \(r\)-hierarchically well-separated tree (r-HST) labelling problem that we can then reduce to a uniform labelling problem.

As described in [25], an r-HST metric [31] \(d^t(\cdot, \cdot)\) is specified by a rooted tree whose edge lengths are non-negative and satisfy the following properties: (i) the edge lengths from any node to all of its children are the same; and (ii) the edge lengths along any path from the root to a leaf decrease by a factor of at least \(r > 1\). Given such a tree, known as r-HST, the distance \(d^t(i, j)\) is the sum of the edge lengths on the unique path between them.
F.1 Approximate the semi-metric with r-HST metric

Fakcharoenphol et al. [32] present an algorithm to get in polynomial time a mixture of r-HST that approximate any semi-metric using a fixed number of trees. This algorithm generate a collection \( D = d^t(\cdot, \cdot), t = 1, ..., n \) where each \( d^t \) is a r-HST metric.

Since this mixture of r-HST metric is an approximation to the original metric, we can approximate the original labelling problem by solving the labelling problem on each of these r-HST metrics and combining them with the method presented in [25]. We are now going to present an efficient algorithm to solve the problem on an r-HST. This algorithm can then be used to solve the problem in the semi-metric case.

F.2 Solve the r-HST labelling problem

We now consider a given r-HST metric \( d^t \) and we note \( T \) all the sub-trees corresponding to this metric and we will use \( T \) as one of these sub-trees. This problem has been formulated by Kleinberg and Tardos [2], but we are not using their method to solve it because the density of our CRF makes their method unfeasible. We are going to solve their original LP directly:

\[
\begin{align*}
\min & \quad \sum_a \sum_i \phi_a(i) y_a(i) + \sum_{a,b \neq a} \sum_T K_{a,b,c_T} \frac{|y_a(T) - y_b(T)|}{2} \\
\text{such that} & \quad \sum_i y_a(i) = 1 \quad \forall a \\
& \quad y_a(T) = \sum_{i \in L(T)} y_a(i) \quad \forall a \forall T \\
& \quad y_a(i) \in \{0, 1\} \quad \forall a, i
\end{align*}
\]

where \( L(T) \) is the set of all labels associated with a sub-tree \( T \).

This problem is the same the Potts model case except for two points:

- In the pairwise term, the labels have been replaced by sub-trees. Since the assignment of the labels to the trees does not change, this won’t prevent us from using the same method to compute the gradient.
- There is a factor \( c_T \) corresponding to the weights in the tree. This does not prevents us from computing this efficiently since we can move this out of the inner loop with the summation over the trees.

Using the same trick where we sort the \( y_a(T) \) for all \( T \), we can rewrite the pairwise part of the above problem and compute its sub-gradient. Using the fact that \( \frac{\partial y_a(T)}{\partial y_c(k)} \) is 0 if \( a \neq c \) or \( k \notin L(T) \) and 1 otherwise and noting \( T_k \) all the
sub-trees that contains \( k \) as one of their label.

\[
\frac{\partial}{\partial y_{c,k}} \left( \sum_{T} \sum_{a,b \neq a} K_{a,b} c_T \frac{|y_a(T) - y_b(T)|}{2} \right)
\]

\[
= \frac{\partial}{\partial y_{c,k}} \left( \sum_{T} \sum_{a,b} K_{a,b} c_T \frac{y_a(T) - y_b(T)}{2} \right) - 2 \sum_{T} \sum_{a,b < a} K_{a,b} c_T \frac{y_a(T) - y_b(T)}{2}
\]

\[
= \sum_{T} \sum_{b} c_T \frac{K_{c,b}}{2} - \sum_{T} \sum_{a} c_T \frac{K_{a,c}}{2} - 2 \sum_{T} \sum_{b < c} c_T \frac{K_{c,b}}{2} + 2 \sum_{T} \sum_{a > c} c_T \frac{K_{a,c}}{2}
\]

\[
= - \sum_{T} \sum_{a < c} c_T K_{a,c} + \sum_{T} \sum_{a > c} c_T K_{a,c}
\]

\[
= \sum_{T} c_T (\sum_{a > c} K_{a,c} - \sum_{a < c} K_{a,c})
\]  

(37)

Since the sorting is done for each sub-tree \( T \), the sums where we consider relative ordering of the indices cannot be switched and thus we cannot simplify this expression further.

We present a method to solve this problem in Algorithm 4. State 3 initialise the subgradient with the unaries. The loop at State 4 is used to compute the participation of each sub-tree to the total subgradient. To do so, we first precompute the \( y_a(T) \) terms for all pixel in the loop starting at State 5. We then sort these \( y_a(T) \) in State 8 and reorder the \( K \) matrix accordingly in State 9. Using this, we can use the divide and conquer approach from the Potts model section to compute the participation of this tree to the subgradient for each pixel using the formula from State 10. The vector \( g_t \) is of size \( N \) and contains one value per pixel. We can now update the subgradient with the partial one on this tree for all labels associated with this tree. The notation \( g \cdot k \) corresponds to a single row of the gradient matrix. This is done in the for loop starting at State 11. We can perform one step of subgradient descent in State 17. Finally we need to project the new point on the feasible set at State 18.
Algorithm 4  r-HST labelling problem

1: Get $y^0$
2: while not converged do
3:     Initialise the subgradient $g = \phi$
4:     for all subtree $T$ do
5:         for all pixel $a$ do
6:             $y_a(T) = \sum_{i \in L(T)} y_a(i)$
7:         end for
8:     Sort $y_a(T)$
9:     Reorder $K$
10:    Gradient for this subtree $g_{T} = \sum_{k > c} K_{a,c} - \sum_{k < c} K_{a,c}$
11:    for all label $k$ do
12:        if $k \in T$ then
13:            Update $g \cdot k += g_t$
14:        end if
15:    end for
16: y$^{t+1} = y^t - \beta g$
17: Project $y^{t+1}$ such that it is a feasible point
18: end while

G  Model used in the experiments section

All the results that we presented in the paper are valid for pixel compatibility functions composed of a mixture of Gaussian kernels:

$$\sum_{m} w^{(m)} k(f^{(m)}_a, f^{(m)}_b).$$

(38)

In practice, for our experiments, we use the same form as Krähenbühl and Koltun [5]. It is a mixture of two gaussian kernels defined using the position vectors $p_a$ and $p_b$, and the colour vectors $I_a$ and $I_b$ associated with each pixel $a$ and $b$. The complete formula is the following:

$$K_{a,b} = w^{(1)} \exp \left(-\frac{|p_a - p_b|^2}{\sigma_1}\right) + w^{(2)} \exp \left(-\frac{|p_a - p_b|^2}{\sigma_{2,spc}} - \frac{|I_a - I_b|^2}{\sigma_{2,col}}\right).$$

(39)

We note that this pixel compatibility function contains 5 learnable parameters $w^{(1)}, \sigma_1, w^{(2)}, \sigma_{2,spc}, \sigma_{2,col}$.

H  More results on stereo

In the following Figures 8, 10 and 9, we observe that the continuous relaxations give consistently better results. We can also note that even though it runs only for a few iterations, the LP improves the visual quality of the solution.
Fig. 5: Evolution of achieved energies as a function of time on a stereo matching problem (Venus Image).

| Method      | Final energy |
|-------------|--------------|
| MF          | -1.341e+07   |
| DC$_{neg}$  | -1.344e+07   |
| QP          | -1.341e+07   |
| QP-DC$_{neg}$ | -1.354e+07  |
| QP-DC$_{gen}$ | -1.352e+07  |
| QP-DC$_{neg}$-LP | -1.354e+07 |

Fig. 6: Evolution of achieved energies as a function of time on a stereo matching problem (Cones Image). Note that the LP in that case is not improving the results.

| Method      | Final energy |
|-------------|--------------|
| MF          | -9.286e+06   |
| DC$_{neg}$  | -9.388e+06   |
| QP          | -8.881e+06   |
| QP-DC$_{neg}$ | -9.868e+06  |
| QP-DC$_{gen}$ | -9.757e+06  |
| QP-DC$_{neg}$-LP | -9.758e+06 |
Fig. 7: Evolution of achieved energies as a function of time on a stereo matching problem (Tsukuba Image). Note that the LP in that case is not improving the results.

| Method         | Final energy |
|----------------|--------------|
| MF             | -8.927e+06   |
| DC<sub>neg</sub> | -8.920e+06   |
| QP             | -9.101e+06   |
| QP-DC<sub>neg</sub> | -9.215e+06   |
| QP-DC<sub>gen</sub> | -9.177e+06   |
| QP-DC<sub>neg</sub>-LP | -9.186e+06   |

Fig. 8: Stereo matching results on the Tsukuba image and corresponding timings. We see that the LP method allows to improve the smoothness from its initialisation DC<sub>neg</sub>. For this set of parameters, the mean-field methods performs poorly.
Fig. 9: Stereo matching results on the Cones image and corresponding timings. Here again, we can see that the MF solution is significantly better than the MF5 solution. The continuous relaxations improve even further by reducing the number of artifacts.

Fig. 10: Stereo matching results on the Venus image. Note that the smoothness of the reconstructions improves with methods reaching lower energies. The LP result does not artefacts anymore, only the non contiguous borders due to the Potts model.
I More results on segmentation

Additional results for parameters cross-validated for MF are presented in Table 2.

We see that in this case where the parameters are tuned for MF, the continuous relaxations still reach lower energy than the mean-field approaches on average. Furthermore, we observe that in almost all images, the energy is strictly lower than the one provided by the mean-field methods. However, with the parameters that were tuned for MF using cross-validation, we note that the segmentation performance is poor compared to mean-field approaches, and that the improved energy minimisation does not translate to better segmentation.

| Unary | MF5 | MF | QPcvx | DCgen | DCneg | LP | Avg. E | Acc | IoU |
|-------|-----|----|-------|-------|-------|----|--------|-----|-----|
| Unary | -   | 0  | 0     | 0     | 0     | 0  | 0      | 79.04| 27.43|
| MF5   | 99  | -  | 0     | 1     | 0     | 1  | 1      | -8.37e5| 80.42| 28.66|
| MF    | 99  | 93 | -     | 4     | 2     | 3  | 3      | -1.19e6| 80.95| 28.86|
| QPcvx | 99  | 93 | 86    | -     | 0     | 4  | 2      | -1.66e6| 77.75| 14.94|
| DCgen | 99  | 94 | 88    | 32    | -     | 29 | 29     | -1.68e6| 77.76| 14.96|
| DCneg | 99  | 93 | 87    | 27    | 2     | -  | 12     | -1.67e6| 77.76| 14.91|
| LP    | 99  | 93 | 87    | 29    | 2     | 17 | -      | -1.67e6| 77.77| 14.93|

Table 2: Percentage of images obtaining strictly lower energy values. Average energy results over the test set and Segmentation performance. Higher percentage is better and lower energy is better. Higher accuracy and IoU are better. Continuous relaxations dominate mean-field approaches on almost all images and improve significantly more compared to the Unary baseline. Parameters tuned for MF.