An Inference Algorithm for Multi-Label MRF-MAP Problems with Clique Size 100

Ishant Shanu[1], Siddhant Bharti[1], Chetan Arora[2], and S. N. Maheshwari[2]

[1] Indraprastha Institute of Information Technology, Delhi, India
[2] Indian Institute of Technology, Delhi, India

Abstract. In this paper, we propose an algorithm for optimal solutions to submodular higher order multi-label MRF-MAP energy functions which can handle practical computer vision problems with up to 16 labels and cliques of size 100. The algorithm uses a transformation which transforms a multi-label problem to a 2-label problem on a much larger clique. Earlier algorithms based on this transformation could not handle problems larger than 16 labels on cliques of size 4. The proposed algorithm optimizes the resultant 2-label problem using the submodular polyhedron based Min Norm Point algorithm. The task is challenging because the state space of the transformed problem has a very large number of invalid states. For polyhedral based algorithms the presence of invalid states poses a challenge as apart from numerical instability, the transformation also increases the dimension of the polyhedral space making the straightforward use of known algorithms impractical. The approach reported in this paper allows us to bypass the large costs associated with invalid configurations, resulting in a stable, practical, optimal and efficient inference algorithm that, in our experiments, gives high quality outputs on problems like pixel-wise object segmentation and stereo matching.

Keywords: Submodular Minimization, Discrete Optimization, Hybrid Methods, MRF-MAP, Image Segmentation.

1 Introduction

Many problems in computer vision can be formulated as pixel labeling problems, in which each pixel $p \in \mathcal{P}$ needs to be assigned a label $l_p \in \mathcal{L}$. Finding the joint labeling configuration, $\mathbf{l}_{\mathcal{P}}$, over all pixels, with maximum posterior probability can then be formulated as a MRF-MAP inference problem [25,48]. The formulation involves solving the following optimization problem: $\mathbf{l}^*_{\mathcal{P}} = \arg\min_{\mathbf{l}_{\mathcal{P}} \in \mathcal{L}^{|\mathcal{P}|}} \sum_{\mathbb{c} \in \mathcal{C}} f_{\mathbb{c}}(\mathbf{l}_{\mathbb{c}})$. Here, $\mathbb{c}$, also called a *clique*, is defined as a set of pixels whose labels are contextually dependent on each other. A labeling configuration on a clique $\mathbb{c}$ is denoted as $\mathbf{l}_{\mathbb{c}}$, $\mathcal{P}$ denotes the set of all pixels and $\mathcal{C}$ denotes the set of all cliques. The *order* of the MRF-MAP problem is considered as one less than the size of the maximal clique, $k = \max_{\mathbb{c} \in \mathcal{C}} |\mathbb{c}|$. Each term, $f_{\mathbb{c}}(\mathbf{l}_{\mathbb{c}})$, also called the *clique potential*, measures the cost of the labeling configuration $\mathbf{l}_{\mathbb{c}}$ of a clique $\mathbb{c}$, depending on how consistent the labeling is with respect to the observation and prior knowledge.

Optimal inference problem, in general, is NP hard even for first order MRFs. Therefore, researchers have explored approximate solutions to the inference problem for first order [9,28,32,52] as well as higher order MRFs [7,33,50]. Another line of research has been to identify sub-classes of clique potentials which model vision problems well and for which optimal inference algorithms can be devised with polynomial time complexity. The MRF-MAP problems with *submodular* clique potentials is one such popular sub-class [2,11,32], which is also the focus of this paper.

Use of higher-order cliques in an MRF-MAP problem is important because it has been established that they can capture more complex dependencies between pixels thereby significantly improving the quality of a labeling solution [21,26,33,40,41,46,51,53]. Our experiments also show improvement over state of the art techniques based on the deep neural networks. Note that MRF-MAP formulation allows one to use the output of deep neural networks as the likelihood term in the objective function. Therefore, performing posterior inference, even using the manually defined priors, helps exploit the problem structure, and improves performance further.

Inference algorithms for higher-order MRF-MAP with general clique potentials output approximate solutions, and are generally based on either message passing/dual decomposition [18,31,33,37,38,47,49] or reduction to first-order potentials frameworks [10,12,15,21,19,24,32,39,41]. The focus of this paper is on developing optimal inference algorithm for multi-label, submodular, higher-order MRF-MAP problems.

One approach to handle multi-label potentials is to use encodings [2,20,53] to convert a multi-label problem to an equivalent 2-label problem while preserving submodularity. However there are some practical challenges. For a multi-label problem of order k with m labels, the encoding blows the problem to cliques of size mk and exploding the size of the solution space to 2^{mk} [2]. Note that only m^k of the 2^{mk} binary configurations resulting from the encoding correspond to the original m^k labeling configurations. The rest are *invalid* in the problem context. Note that if potentials for invalid states are kept very large and those for valid states the same as in the original multi-label version, the minimum is always among the valid states.

The use of Block Co-ordinate Descent (BCD) based techniques to handle the Min Norm Point polyhedral algorithm[45,46] is also possible in principle for such transformed problems. But the encoding based transformations pose new challenges. As explained in the next section, these techniques maintain the current feasible base vector as a convex combination of a set of extreme bases. For the 2-label problems arising out of encoding multi-label versions, some of the values in the extreme bases can correspond to energy of the invalid states. Giving a large or effectively an infinite value to the invalid states creates numerical challenges in maintaining/updating these convex combinations. Also, encoding increases the size of the cliques by m times, which increases the dimensions of the polyhedral space to an extent that cannot be handled by the algorithm in [46].

The main contribution of this paper is to show that there is enough structure in the submodular polyhedron to handle invalid extreme bases arising out of converted 2-label problems efficiently. The proposed algorithm raises the bar significantly in that using it we can handle multi-label MRF-MAP problems with 16 labels, and clique size upto 100. In comparison the current state of the art [2] can only work with cliques of size up to 4.

At this stage we would like to contrast out mapping technique with that of [29], which has exploited the linear relationship between a tree and order in labels, to map multi-label submodular functions to the more general class of tree based $L^\natural$-convex functions. However, these algorithms have high degree polynomial time complexity (based on [35,22,36]), limiting them to be of theoretical interest only. Our focus on the other hand is to extend the frontiers of practical optimal algorithms.

Finally, we would like to point out that when the case for higher-order potential was first made, the then existing algorithms could only work with small cliques. Solutions were approximate and potentials many times were decomposable [26,33,41]. It is only with [45] and [46] that experiments could be done with cliques of size 100 or larger. Experiments reported in [46] established that quality of object segmentation improves with larger clique sizes [46]. We extend that exercise further here by focusing on quality of multi object segmentation as a function of clique size.

2 Background

We briefly describe the basic terminology and results from *submodular function minimization* (SFM) literature required to follow the discussion in this paper. We direct the reader to [44] for more details. The objective of a SFM problem is to find a minimizer set, $S^* = \min_{S \subseteq \mathcal{V}} f(S)$ of a submodular function f, where $\mathcal{V}$ is the set of all the elements. W.l.o.g. we assume $f(\phi) = 0$. We associate two polyhedra in $\mathbb{R}^{|\mathcal{V}|}$ with f, the *submodular polyhedron*, $P(f)$, and the *base polyhedron*, $B(f)$, such that

$$\begin{aligned} P(f) =& \{x \mid x \in \mathbb{R}^{|\mathcal{V}|},\ \forall\ U \subseteq \mathcal{V} : x(U) \leq f(U)\},\ \text{and} \\ B(f) =& \{x \mid x \in P(f), x(\mathcal{V}) = f(\mathcal{V})\}, \end{aligned}$$

where $x(v)$ denotes the element at index v in the vector x, and $x(U) = \sum_{v \in U} x(v)$. A vector in the base polyhedron $B(f)$ is called a *base*, and an extreme point of $B(f)$ is called an *extreme base*. *Edmond's greedy algorithm* gives a procedure to create an extreme base, $b^\prec$, given a total order $\prec$ of elements of $\mathcal{V}$ such that $\prec : v_1 \prec \ldots \prec v_n$, where $n = |\mathcal{V}|$. Denoting the first k elements in the ordered set $\{v_1, \ldots, v_k, \ldots, v_n\}$ by $k_\prec$, the algorithm initializes the first element as $b^\prec(1) = f(\{v_1\})$ and rest of the elements as $b^\prec(k) = f(k_\prec) - f((k-1)_\prec)$. There is a one to one mapping between an ordering of the elements, and an extreme base. The *Min Max Theorem*, states that $\max\{x^-(\mathcal{V}) \mid x \in B(f)\} = \min\{f(U) \mid U \subseteq \mathcal{V}\}$. Here, $x^-(\mathcal{V})$ gives the sum of negative elements of x.

The min-norm equivalence result shows that $\arg\max_{x\in B(f)} x^{-}(\mathcal{V}) = \arg\min_{x\in B(f)} \|x\|_2$. Fujishige and Isotani's [14] *Min Norm Point* (MNP) algorithm uses the equivalence and solves the problem using Wolfe's algorithm [13]. The algorithm has been shown empirically to be the fastest among all base polyhedron based algorithms [23,46]. The algorithm maintains a set of extreme bases, $\{b^{\prec_i}\}$, and a minimum norm base vector, x, in their convex hull, s.t.:

$$x = \sum_i \lambda_i b^{\prec_i} \quad \lambda_i \geq 0, \text{ and } \sum_i \lambda_i = 1. \tag{1}$$

At a high level, an iteration in the MNP/Wolfe's algorithm comprises of two stages. In the first stage, given the current base vector, x, an extreme base, q, that minimizes $x^{\intercal}q$ is added to the current set. The algorithm terminates in case $\|x\| = x^{\intercal}q$. Otherwise it finds a new x, with smaller norm, in the convex hull of the updated set of extreme bases.

The MRF-MAP inference problem can be seen as minimizing a sum of submodular functions [30,46]. Shanu et al. [46] have suggested a block coordinate descent framework to implement the Min Norm Point algorithm in the sum of submodular functions environment when cliques are large. A very broad overview of that scheme is as follows.

With each $f_{\mathbb{c}}$, the submodular clique potential of clique $\mathbb{c}$, one can associate a base polyhedron such that:

$$B(f_{\mathbb{c}}) := \Big\{ y_{\mathbb{c}} \in \mathbb{R}^{|\mathbb{c}|} \mid \; y_{\mathbb{c}}(U) \leq f_{\mathbb{c}}(U), \; \forall U \subseteq \mathbb{c} \; ; \; y_{\mathbb{c}}(\mathbb{c}) = f_{\mathbb{c}}(\mathbb{c}) \Big\}. \tag{2}$$

The following results [46] relate a base vector x of function f, and a set of base vectors $y_{\mathbb{c}}$ of a $f_{\mathbb{c}}$:

Lemma 1. *Let $x(S) = \sum_{\mathbb{c}} y_{\mathbb{c}}(\mathbb{c} \cap S)$ where each $y_{\mathbb{c}}$ belongs to base polyhedra $B(f_{\mathbb{c}})$. Then the vector x belongs to base polyhedron $B(f)$.*

Lemma 2. *Let x be a vector belonging to the base polyhedron $B(f)$. Then, x can be expressed as the sum: $x(S) = \sum_{\mathbb{c}} y_{\mathbb{c}}(S \cap \mathbb{c})$, where each $y_{\mathbb{c}}$ belongs to the submodular polyhedron $B(f_{\mathbb{c}})$ i.e., $y_{\mathbb{c}} \in B(f_{\mathbb{c}})$ $\forall$ $\mathbb{c}$.*

The block coordinate descent approach based on the results requires each block to represent a base vector $y_{\mathbb{c}}$ as defined above (c.f. [46]). Note that a base vector $y_{\mathbb{c}}$ is of dimension $|\mathbb{c}|$ (clique size), whereas a base x is of dimension $|\mathcal{V}|$ (number of pixels in an image). Since $|\mathbb{c}| \ll |\mathcal{V}|$, minimizing the norm of $y_{\mathbb{c}}$ over its submodular polyhedron $B(f_{\mathbb{c}})$ is much more efficient than minimizing the norm of x by just applying the MNP algorithm. However, for reasons already given and discussed in the Introduction, the algorithm based on the above fails to converge on multi-label submodular MRF-MAP problems when transformed to a 2-label MRF-MAP problems using an extension of the encoding given in [2] that preserves submodularity.

We now show how these problems can be overcome by performing block coordinate descent over two blocks: one block has convex combination of only extreme

bases corresponding to valid states and the other has the convex combination of extreme bases corresponding to the invalid states. The block corresponding to valid states is small enough for the traditional MNP algorithm to output optimal solutions. For the larger block corresponding to the invalid states we develop a flow based algorithm to find a vector with minimum ℓ_2 norm. This results in an algorithm which is numerically stable and practically efficient.

3 Properties of the Multi-label to 2-Label Transformation

Let F be a multi-label submodular function defined over the set of n pixels $\mathcal{P}$. Let X and Y stand for the n-tuples of parameters. Let $\vee$ and $\wedge$ be max and min operators and let $(X \vee Y), (X \wedge Y)$ denote the n-tuples resulting from element wise application of the max and min operators over n-tuples X and Y. F is called submodular if:

$$F(X) + F(Y) \geq F(X \vee Y) + F(X \wedge Y). \tag{3}$$

We now summarize the transformation to convert a multi-label to a 2-label problem as suggested in [2,20]. Consider an *unordered* set of pixels $\mathcal{P} = \{p_1, \ldots, p_i, \ldots, p_n\}$, and an *ordered* set of labels $\mathcal{L} = \{1, \ldots, m\}$. To save the notation clutter, whenever obvious, we denote a pixel simply using variables p, q without the subscript index.

Definition 1 (Binary Encoding). *The encoding $\mathcal{E} : \mathcal{L} \rightarrow \mathbb{B}^m$ maps a label $i \in \mathcal{L}$ to a m dimensional binary vector such that its first $m - i$ elements are 0 and the remaining elements are 1.*

For example, $\mathcal{E}(1) = (0, \ldots, 0, 0, 1)$, and $\mathcal{E}(2) = (0, \ldots, 0, 1, 1)$. Let us denote the encoded label vector corresponding to a pixel p_i as $\gamma_i = (p_i^1, \ldots, p_i^m), p_i^j \in \{0, 1\}$. We denote by $\Gamma \in \mathbb{B}^{mn}$, the vector obtained by concatenating all encoded vectors $\gamma : \Gamma = (\gamma_1, \ldots, \gamma_i, \ldots, \gamma_n)$. The vector Γ represents encoding of labeling configuration over all the pixels. We also define a *universal set* containing all elements of $\Gamma : \mathcal{V} = \{p_1^1, \ldots, p_1^m, \ldots, p_n^1, \ldots, p_n^m\}$.

Definition 2 (Universal Ordering). *Assuming an arbitrary ordering among the pixels, the universal ordering, defines a total ordering of the elements p_i^j, $i \in \mathcal{Z}_{1:n}, j \in \mathcal{Z}_{1:m}$:*

$$\prec_0 : p_1^1 \prec \cdots \prec p_1^m \prec \cdots \prec p_n^1 \cdots \prec p_n^m.$$

We denote by $S \subseteq \mathcal{V}$, called *state*, set of all the elements, p_i^j of Γ labeled as 1. Note that there are 2^{mn} possible states, however only m^n of them correspond to valid Γ vector obtained by encoding labeling configurations over the pixels. We call such states as *valid states*. If label of a pixel p_i is denoted as $l_i \in \mathcal{L}$, a valid state may be represented as: $S = \{\mathcal{E}(l_1), \ldots, \mathcal{E}(l_i), \ldots, \mathcal{E}(l_n)\}$. Similarly $S_p = \{\mathcal{E}(l_p)\}$ includes elements corresponding to pixel p.

Definition 3 (Valid Ordering/Extreme Base). *An ordering $\prec$ is called a valid ordering, if for any $p_i^j, p_i^k \in \mathcal{V}$, $j > k \Rightarrow p_i^j \prec p_i^k$. An extreme base $b^{\prec}$ is called a valid extreme base, if it corresponds to a valid ordering.*

The states, orderings or extreme-bases which are not valid are called *invalid*. We denote the set of all valid states by $\mathcal{S}$.

Definition 4 (Covering State, Minimal Covering State). *For an arbitrary state, S, a valid state, $\hat{S} \in \mathcal{S}$, is called covering if $S \subseteq \hat{S}$. There may be multiple covering states corresponding to a S. The one with the smallest cardinality among them is referred to as the minimal covering state, and is denoted by $\overline{S}$. There is a unique minimal covering state corresponding to any S. For a valid state $S = \overline{S}$.*

We are now ready to show that the above transformation can be used to define a binary set function which is not only submodular but is also identical to the multi-label submodular function on valid states. We encode the multi-label function to a submodular pseudo-Boolen function f defined over set $\mathcal{V}$ of size mn as follows:

Definition 5 (The Extended Binary Set Function).

$$f(S) = \begin{cases} F(\ldots, l_i, \ldots), & \text{if } S = \{\ldots, \mathcal{E}(l_i), \ldots\} \\ f(\overline{S}) + (|\overline{S}| - |S|)L & \text{otherwise} \end{cases}$$

Here $l_i \in \mathcal{L}$ is label of pixel p_i, and $L \gg M = [\max_{S \in \mathcal{S}} f(S) - \min_{S \in \mathcal{S}} f(S)]$.

It is easy to see that $f(S)$ can also be defined as follows:

Definition 6 (The Extended Binary Set Function: Alternate Definition).

$$f(S) = f(\overline{S}) + \sum_{p \in \mathcal{P}} (|\overline{S}_p| - |S_p|)L, \tag{4}$$

where $\overline{S}_p \subset \overline{S}$, and $S_p \subset S$ are the subsets containing elements corresponding to pixel p in $\overline{S}$ and S respectively.

Theorem 1. *The extended binary set function f, as given by Definition 5, is submodular, and* $\min f(\cdot) = \min F(\cdot)$.

To prevent the breaking of thought flow, and due to restrictions on length, the detailed proof of this theorem as well as those following are given in the supplementary material.

The reader may, at this stage, wonder whether it is at all possible to limit to working only with the valid states in the submodular mode, perhaps using one-hot encoding as in [53]. The answer is no, since in a one-hot encoding the set of all valid states is not a ring family [34], and hence the encoded function is not submodular.

Note that in the proposed encoding, any value of $L \gg M$, keeps the function, f, submodular. However, as we show later, choosing such a large value of L, makes the contribution of some extreme bases very small causing precision issues in the computation. We also show that including those extreme bases with very small contribution is extremely important for achieving the optimal inference. The major contribution of this paper is in showing that one can perform an efficient inference bypassing L altogether. Therefore, the use of L is merely conceptual in our framework. There is no impact of actual value of L on the algorithm's performance.

4 Representing Invalid Extreme Bases

In the discussion that follows, we refer to any scalar as *small* or *finite* if its absolute value is $\ll L$, and *large* or *infinite* if the absolute value is $\propto L$. We write Eq. (1) as:

$$x = x_v + x_i = \sum_{b^{\prec_j} \in R} \lambda_j b^{\prec_j} + \sum_{b^{\prec_i} \in Q} \lambda_i b^{\prec_i}. \tag{5}$$

Here, R and Q are the sets of valid and invalid extreme bases, and x_v, and x_i, their contribution in x respectively. It is easy to see that, all the elements of x_i must be much smaller than L [3]. We first focus on the relationship between λ and L in the block of invalid extreme bases.

Lemma 3. *For any element, e, of an invalid extreme base, $b^{\prec} : b^{\prec}(e) = a_e L + b_e$, where $|a_e|, |b_e| \ll L$ and $a_e \in I$.*

Lemma 4. *Consider two base vectors x_1 and x_2 such that $\|x_1\|^2, \|x_2\|^2 < |\mathcal{V}|M^2$. If $x_2 = (1-\lambda)x_1 + \lambda b^{\prec}$ and $b^{\prec}$ is an invalid extreme base, then $\lambda \leq |\mathcal{V}|\frac{M}{L}$.*

Conceptually, Lemma 3 shows that all elements of an invalid extreme base are either small or are proportional to L (and not proportional to, say L^2, or other higher powers of L). Whereas, Lemma 4 shows that since $\mathcal{V}$ and M are effectively constants, λ the multiplicative factor associated with in the contribution of invalid extreme bases, λ is proportional to $1/L$. Therefore, for $L \approx \infty$, the value of $\lambda \approx 0$. However, it is important to note that the value of $\lambda b^{\prec}(e)$, is always finite. It is easy to see that, whenever $a_e = 0$, $\lambda b^{\prec}(e) \approx 0$, and when $a_e \neq 0$, the L present in the $b^{\prec}(e)$ and $1/L$ present in λ cancel each other, leading to a finite contribution. The argument as given above motivates our overall approach in this paper that, for a numerically stable norm minimization algorithm, focus should be on manipulating the finite valued product $\lambda b^{\prec}$, and not the individual λ and $b^{\prec}(e)$. We show in the following sections that this is indeed possible.

We start by showing that it is possible to find a small set of what we call *elementary* invalid extreme bases whose linear combination contains as a subset the space of vectors x_i as given in Eq. (5). Crucial to doing this is the notion of *canonical orderings*.

[3] We start the algorithm with a valid extreme base, where the condition is satisfied. In all further iterations the norm of x decreases monotonically, and the condition continues to remain satisfied.

4.1 Canonical Ordering and Its Properties

In an arbitrary, valid or invalid, ordering $\prec$ consider two adjacent elements u and v such that $u \prec v$. We term swapping of order locally between u and v in $\prec$ as an *exchange operation*. The operation will result in a new ordering $\prec_{\text{new}}$ such that u and v are still adjacent but $v \prec_{\text{new}} u$.

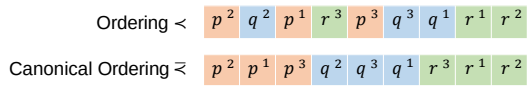

Fig. 1: Top: An ordering of elements in $\mathcal{P} = \{p, q, r\}$, for a label of size 3. Bottom: Corresponding canonical ordering.

Consider a strategy in which starting with $\prec$ we carry out exchange operations till all the elements corresponding to a pixel come together, and repeat this for all pixels. Note that we do not change the relative ordering between elements corresponding to the same pixel. We call the resultant ordering the *canonical* form of the original ordering $\prec$ and denote it by $\overline{\prec}$. The corresponding extreme base is called *canonical extreme base*. Note that there can be multiple canonical forms of an ordering. Figure 1 contains an example of an arbitrary ordering and one of its canonical orderings. We emphasize here that there may be more than one canonical orderings corresponding to $\prec$.

Note that a valid (invalid) ordering leads to a valid (invalid) canonical ordering. For any p^j and p^k, in a valid canonical ordering, if $j = k + 1$, then p^j, p^k are adjacent in the ordering and $p^k \overline{\prec} p^j$. Further, a canonical ordering is agnostic to any relative order among pixels. For example, for pixels p and q, a canonical ordering only requires that all elements of p (or q) are contiguous. An ordering in which elements corresponding to p come before those of q will define a different canonical ordering from the one in which the relative ordering of elements of p and q is vice-versa. In general a canonical ordering $\overline{\prec}$ corresponding to a $\prec$ can be any one of the possible canonical orderings.

Lemma 5. *Let $\prec$ be an invalid ordering and $\overline{\prec}$ be its canonical ordering. Then, $b^{\prec}(e) - b^{\overline{\prec}}(e) \ll L, \forall e \in \mathcal{V}$.*

The above result serves to indicate that by changing an invalid extreme base to canonical one, the change in value of any element of the extreme base is much less than L. Therefore, due to Lemma 4, one can conclude that the contribution of an invalid extreme base or its canonical extreme base in a base vector is going to be the same.

Lemma 6. *For a canonical invalid ordering $\overline{\prec}$, let p^i and p^j be two adjacent elements corresponding to a pixel p, s.t. $p^i \overline{\prec} p^j$. Let $\overline{\prec}_p^{i,j}$ be the ordering obtained by swapping p^i and p^j. Then: $b^{\overline{\prec}_p^{i,j}} - b^{\overline{\prec}} = (\chi_p^j - \chi_p^i)(aL + b)$, where χ_p^i is an indicator vector for the element p^i, and $a, b \ll L$.*

Lemma 6 relates the two extreme bases when one pair of their elements is swapped. It is useful to note that in a valid extreme base all elements have small values. With each swap in an invalid canonical ordering we either move

the canonical ordering towards validity or away from it. In each swap the change in the value of an element is proportional to L (positive or negative). Since conversion of an invalid canonical ordering to a valid one may involve swaps between a number of elements, the extreme base corresponding to the invalid ordering will contain multiple elements with values proportional to L. The special cases are the ones in which only one swap has been done. In these cases there will be only two elements with values proportional to L (positive and negative). We show that using such extreme bases as the basis to represent canonical invalid extreme bases. In the next section we show that it is indeed possible.

4.2 Elementary Invalid Extreme Base

Definition 7 (Elementary Invalid Extreme Base). *The ordering obtained by swapping two elements p^j and p^{j+1}, corresponding to a pixel p, in a canonical valid ordering, is called an* elementary invalid ordering. *Its corresponding extreme base is called* elementary invalid extreme base, *and is denoted as $b^{\widetilde{\prec}^j_p}$.*

Lemma 7. *Consider an elementary invalid extreme base $b^{\widetilde{\prec}^i_p}$, obtained by swapping two adjacent elements (p^{i+1}, p^i) in the universal ordering, $\prec_0$ (Def. 2). Then: $b^{\widetilde{\prec}^i_p} - b^{\prec_0} = (\chi^i_p - \chi^{i+1}_p)(L+b)$, where $b^{\prec_0}$ is the valid extreme base corresponding to $\prec_0$.*

Lemma 8. *An invalid canonical extreme base, $b^{\overline{\prec}}$, can be represented as a linear combination of elementary invalid extreme base vectors such that: $b^{\overline{\prec}} = \sum_{p\in\mathcal{P}}\sum_{i=1}^{m-1}\alpha^i_p b^{\widetilde{\prec}^i_p} + \Lambda$, where $0 < \alpha^i_p \ll L$, and Λ is a vector with all its elements much smaller than L.*

Due to Lemma 5, the above result is also true for representing the invalid extreme bases (and not only the canonical ones), with a different Λ. Lemma 7 allows us to further simplify the result of Lemma 8 to the following:

Lemma 9 (Invalid Extreme Base Representation). *An invalid extreme base can be represented as $b^{\prec} = \sum_{p\in\mathcal{P}}\sum_{i=1}^{m-1}\alpha^i_p L(\chi^i_p - \chi^{i+1}_p) + \Lambda$, where χ^i_p is an indicator vector corresponding to element p^i, $0 < \alpha^i_p \ll L$, and Λ is some vector whose all elements are $\ll L$.*

Recall from Eq. (5): $x = x_v + x_i$, where $x_v = \sum_{b^{\prec_j}\in R}\lambda_j b^{\prec_j}$, and $x_i = \sum_{b^{\prec_i}\in Q}\lambda_i b^{\prec_i}$. Using Lemma 9 to replace the second term, and noting that $L \approx \infty \Rightarrow \lambda_i \approx 0$, and $\sum\lambda_j \approx 1$, one observes that the term $\sum_{b^{\prec_i}\in Q}\lambda_i\Lambda_i$ in the expansion can be made smaller than the precision constant by increasing the value of L ($\lambda < |\mathcal{V}|M/L$ by Lemma 4) and can be dropped. As one of the final theoretical results of this paper, we can show the following:

Theorem 2 (Main Result).

$$\sum_{\forall b^{\prec_i}\in Q}\lambda_i b^{\prec_i} = \sum_{p\in\mathcal{P}}\sum_{k=1}^{m-1}\beta^k_p L(\chi^k_p - \chi^{k+1}_p), \tag{6}$$

where $\lambda_i \geq 0$, $\beta^k_p = \sum_{b_i\in Q}\alpha^k_p\lambda_i$.

Note that the above result incorporates *all* the invalid extreme bases, not merely the ones involved in the representation of base vector x in any iteration of MNP. Using the result in Eq. (5), we get: $\|x\|^2 = \left\|\sum_{b^{\prec_j} \in R} \lambda_j b^{\prec_j} + \sum_{p \in \mathcal{P}} \sum_{k=1}^{m-1} \beta_p^k L(\chi_p^k - \chi_p^{k+1})\right\|^2$.

5 The Multi-label Hybrid Algorithm

In this section we give the algorithm for minimizing the norm of the base vector corresponding to a single clique in the original MRF-MAP problem, where the pseudo-Boolean function is generated from encoding the multi-label function. For solving the overall MRF-MAP problem with multiple cliques, the proposed algorithm can be used in the inner loop of the BCD strategy as suggested in [46].

Theorem (2) opens up the possibility of minimizing $\|x\|^2$ for a single clique using the BCD strategy. We will have two blocks. The first block, called the *valid block*, is a convex combination of valid extreme bases $b^{\prec_j}$, where standard MNP algorithm can be used to optimize the block. The other block, called the *invalid block*, corresponds to the sum of the mn terms of type: $\beta_p^k L(\chi_p^k - \chi_p^{k+1})$, representing the invalid extreme bases. For minimizing the norm of the overall base vector using the invalid block, we hold the contribution from the valid block, x_v, constant [4]. Each vector $\beta_p^k L(\chi_p^k - \chi_p^{k+1})$ may be looked upon as capturing the β_p^k increase/decrease due to the exchange operation between the two adjacent elements which define an elementary extreme base. This exchange operation can be viewed as flow of $\beta_p^k L$ from the element p^{k+1} to p^k. We model the optimization problem for the invalid block using a flow graph whose nodes consists of $\{p^k \mid p \in \mathcal{P}, 1 \leq k \leq m-1\} \cup \{s, t\}$. We add two type of edges:

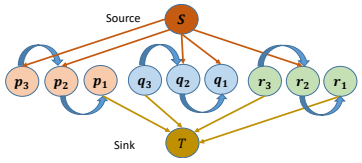

Fig. 2: Flow graph corresponding to the exchange operations for optimizing the block containing invalid extreme bases.

- **Type 1:** If $x_v(p^k)$, corresponding to the valid block contribution, is > 0, then we add a directed edge from $s \rightarrow p^k$, else we add the edge from $p^k \rightarrow t$ with capacity $x_v(p^k)$.
- **Type 2:** The directed edges p^{k+1} to p^k, $1 \leq k \leq (m-1)$ with capacity $|\mathcal{V}|M$ to ensure that the capacity is at least as large as $\beta_p^k L$: much larger than any permissible value of $x_v(p^k)$. Thus, any feasible flow augmentation in a path from from s to t can saturate only the first or the last edge in the augmenting path (i.e. the edge emanating from s or the edge incident at t in the path).

Figure 2 is an example of a flow graph for 3 pixel and 3 label problem. Since the starting state is x_v the "initial flow" prior to pushing flow for flow maximization requires setting flow in a type 1 edge incident at p^k equal to the value of $x_v(p^k)$ and that in type 2 edges as 0. This is because sum of flow on all

[4] Recall that we start from a valid extreme base. Therefore, at initialization $x = x_v$

Algorithm 1 Computing Min ℓ_2 Norm from the Flow Output

Input: Vector e the output of the max flow algorithm.
Output: The transformed vector e with minimum ℓ_2 norm.

for $\forall p \in \mathcal{P}$ **do**
  for $i = 2 : m$ **do**
    repeat
      Find smallest k, $i \geq k \geq 1$, such that
      $e(p^i) > e(p^{i-1}) = e(p^{i-2}) \cdots = e(p^k)$ or
      $e(p^i) = e(p^{i-1}) = e(p^{i-2}) \cdots = e(p^{k+1}) > e(p^k)$;
      Set $e(p^i), e(p^{i-1}), \ldots, e(p^k)$ equal to av_k,
      where av_k is the average of $e(p^i), e(p^{i-1}), \ldots, e(p^k)$;
    until $e(p^{k+1}) \leq e(p^k)$
  end for
end for

edges incident at a node may be looked upon as the value of the corresponding element in the base vector [5]. In effect initially there are non zero excesses on the non s, t nodes in the flow graph defined as the sum of net in-flow on all edges incident at a node. The excess at node p_k is denoted by $e(p_k)$. Max flow state can be looked upon as that resulting from repeatedly sending flow from a positive excess vertex to a negative excess vertex till that is no more possible. Values in the optimal base vector (optimal subject to the given x_v) at the end of this iteration will be the excesses at nodes when max flow state has been reached.

5.1 Computing Min ℓ_2 Norm By Flow

Since there is no edge between any two nodes corresponding to different pixels max flow can be calculated independently for each pixel. When max flow state is reached in the flow graph associated with a pixel, a vertex which still has a negative excess will be to the left of vertices with positive excess (planar flow graph laid out as in Figure 2) otherwise flow could be pushed from a positive excess vertex to a negative excess vertex.

Note that the optimal base vector is not unique. Consider two adjacent vertices, p^{k+1} and p^k, in the flow graph when the max flow state has been reached. If $e(p^{k+1})$ is larger than $e(p^k)$ then increasing the flow in the edge from p^{k+1} to p^k by δ decreases $e(p^{k+1})$ by δ and increases $e(p^k)$ by δ. The result of this "exchange operation" is to create another optimal base vector but with a smaller ℓ_2 norm.

An optimal base vector with minimum ℓ_2 norm will correspond to the max flow state in the flow graph in which $e(p^{k+1}) \leq e(p^k)$ for all adjacent pairs of type 2 vertices. If this is not so then there would exist at least a pair $e(p^{k+1})$ and $e(p^k)$ such that $e(p^{k+1}) > e(p^k)$. Doing an exchange operation between p^{k+1} and p^k involving setting $e(p^{k+1})$ and $e(p^k)$ to the average of the old values will create a new optimal base vector with lower value of the ℓ_2 norm. Algorithm 1 gives an efficient procedure to transform the optimal base vector outputted by the max

[5] we refer the reader to [45] for details about the flow to base vector correspondence

flow algorithm to one with minimum ℓ_2 norm. Note that the proposed algorithm simply updates the base vector in one pass without any explicit flow pushing. In contrast, the corresponding algorithm for general flow graphs given in [45] requires $O(n \log n)$ additional max flow iterations over an n vertex flow graph.

5.2 Overall Algorithm

The proposed Multi-label Hybrid (MLHybrid) algorithm is quite similar to the algorithm in [46] in its over all structure. Just like [46], we also create blocks corresponding to each clique, and optimize each block independently (taking the contribution of other blocks as suggested in [46]) in an overall block coordinate descent strategy. The only difference between SoSMNP and MLHybrid is the way we optimize one block. While SoSMNP uses standard MNP, we optimize using a special technique, as outlined in previous section, with (sub)blocks of valid and invalid extreme bases, within each block/clique. Hence, the convergence and correctness of overall algorithm follows from block coordinate descent similar to [46]. What we need to show is that for a single clique/block, the algorithmic strategy of alternating between valid and invalid blocks converges to the optimal for that clique/block.

Recall that in a standard MNP algorithm iteration, given the current base vector x, an extreme base q, that minimizes $x^\intercal q$ is added to the current set. Hence, steps to convergence of MNP is bounded by the number of extreme bases that may be added. In our case we have shown in the Supplementary Section that when we start with a valid extreme base, the extreme base generated in the valid block after using the latest contribution from the invalid block, will come out to be a valid extreme base. This implies that the number of iterations involving invalid blocks can not exceed the number of valid extreme bases added as in the standard MNP algorithm. This ensures convergence of the optimization step for each block. The formal convergence proof for the MLHybrid algorithm is given in the Supplementary Section.

The correctness of our optimization for each block follows from the fact that the optimization for valid blocks proceeds in the standard way, and results in a new extreme base given the current base vector. The correctness of the optimization step of the invalid block, which finds a minimum norm base vector given a valid block, has already been explained in the previous section.

6 Experiments

We have experimented with pixel-wise object segmentation and stereo correspondence problems. All experiments have been conducted on a computer with Intel Core i7 CPU, 8 GB of RAM running Windows 10. Implementation of our algorithm is in C++ (`https://github.com/ishantshanu/ML-Minnorm`). For the segmentation experiments, the input images are from Pascal VOC dataset [8] with a small amount of Gaussian noise added. We have experimented with two types of submodular clique potentials:

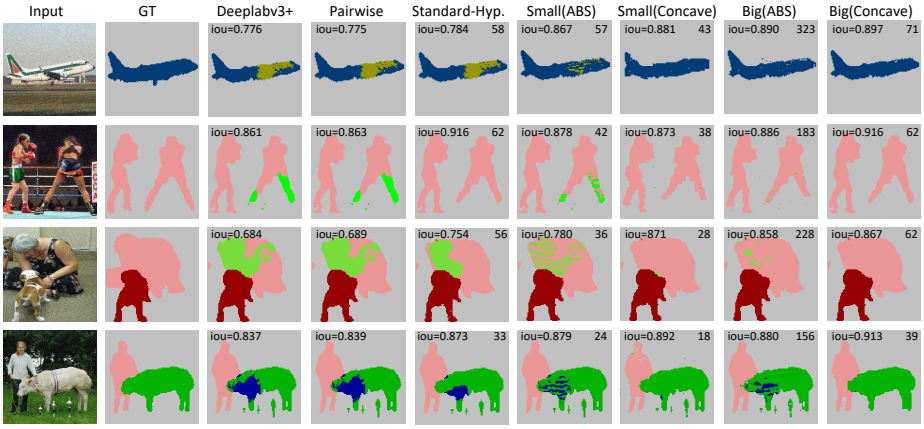

Fig. 3: Pixel-wise object segmentation comparison. Input images from the Pascal VOC dataset.

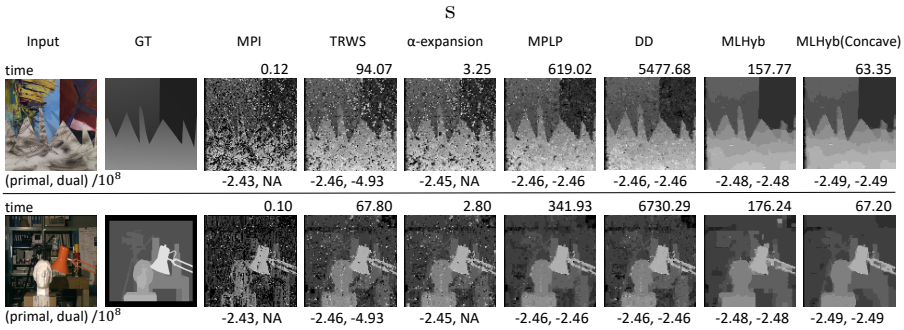

Fig. 4: Stereo matching problem. Input images from the Middlebury dataset.

– Decomposable: Sum of absolute difference of labels for all pixel pairs in a clique. Denoted by ABS.
– Non-decomposable: Concave-of-Cardinality potential defined in [53] as: $\sum_{l\in\mathcal{L}}(\text{number of pixels} - \text{number of pixels which have their label as } l)^{\alpha}$. We have used $\alpha = 0.5$ in our experiments.

For both the potentials, two types of clique sizes namely "Small" (cliques ranging from 60 to 80 elements) and "Big" (cliques ranging from 300 to 400 elements) have been used for the experiments. Overlapping of cliques has been ensured by running SLIC algorithm [1] with different seeds.

Figure 5 shows the IOU values as bars for Deeplabv3+ [6] fine-tuned on noisy images (red), running MLHybrid with small cliques (green) and with big cliques (blue) on all the classes of the VOC dataset for the segmentation problem. The likelihood of a label on each pixel, required for our algorithm, is estimated using the scaled score from the Deeplabv3+. The scaling factors are specific to labels and are the hyper-parameters in our algorithm. We use the pre-trained version

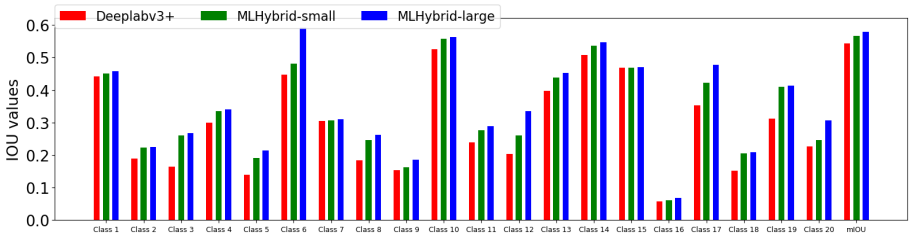

Fig. 5: Shows IOU values across all the classes of PASCAL VOC dataset.

of Deeplabv3+ from [6]. Deeplabv3+ gives overall pixel accuracy of 82.79 and with MLHybrid we get pixel accuracy of 84.07 and 85.11 respectively for small and big cliques. Mean IOU values (three bars at the right end) are 0.544, 0.566, and 0.579 respectively. MLHybrid has been run with non-decomposable clique potentials and the same standard fixed hyper parameters on the VOC dataset.

The performance of MLHybrid improves with fine tuning of hyper parameters. Figure 3 shows the visual results on four pictures from the data set when the hyper parameters have been tuned. To show the extent of improvement we have also included in Figure 3 the MlHybrid output with the standard hyper parameters (standard-hyp). We have also included the IOU values in the images (upper left hand corner) corresponding to Deeplabv3+, MlHybrid (Big(concave)) run with standard and fine tuned hyper parameters respectively. For all the four images IOU values hover around 0.9 when MLHybrid is run with big cliques and concave potentials. Run time for MLhybrid in seconds are shown at the upper right corner of the respective images. Deeplabv3+ takes approximately 0.5 seconds per image excluding the training time. Hyper parameters for α-expansion running on pairwise cliques (4^{th} column in Figure 3) are the optimized parameters used for MLHybrid as are the likelihood labels for the pixels.

Note that the quality of output is distinctly better for the non-decomposable concave potential in comparison to the decomposable ABS potential for both Small and Big clique configurations. The output for Big(Concave) matches the ground truth significantly. The time taken for concave potentials is distinctly less than ABS potentials for the same size and number of cliques. This difference is because the number of iterations taken for convergence is proportionately less for non-decomposable potentials. It is reasonable to infer that the segmentation quality improves with clique size. Since for large cliques, potentials will need to be predefined and not learnt, designing clique potentials calls for further investigation. Also, since fine tuning of hyper parameters improves quality of segmentation results significantly an area of research with high pay off is how to automate the process of fine tuning the hyper parameters for the segmentation problem.

For stereo correspondence, the images are from Middelbury dataset [42] and are of size 200×200. The cliques are generated, as earlier, using SLIC algorithm. Label likelihood is calculated using Birchfield/Tomasi cost given in [3]. There are 16 disparity labels considered and clique potential used is the same as for the segmentation problem. Figure 4 shows the output. We have compared with

implementations of Max Product Inference (MPI) [27], TRWS [28], MPLP [16], α-expansion [4] available in Darwin framework [17]. We use the pair wise absolute difference of labels potential with a pixel covered by maximum of four cliques. Other than α-expansion, other methods could not handle pairwise potentials emanating out of all pairs of variables in a clique of size 50 or larger. Primal/Dual values are shown below the images and their corresponding running times on the top.

Our final experiments are to show efficacy of convergence of the MLHybrid algorithm. Table 1 shows the performance of SOS-MNP [46] on the extended pseudo-boolean submodular function. Since [46] do not bypass L therefore we run it for different values of L. Note that primal and dual do not converge even when the value of L is as large as 10^{15} after running the algorithm for approximately 50 minutes. SOS-MNP not only takes huge amount of time but do not even converge to the right point.

$L =$	10^9	10^{11}	10^{13}	10^{15}
Primal	$1.26(10^{15})$	$1.26(10^{17})$	$-1.75(10^8)$	$-1.77(10^8)$
Dual	$-5.37(10^8)$	$-5.37(10^8)$	$-5.58(10^8)$	$-5.60(10^8)$

Table 1: Primal dual for SoS-MNP [46] for different values of L.

In contrast Figure 6 shows the convergence performance of the MLHybrid algorithm for solving a stereo problem on the sawtooth sample with sum of absolute difference potential. The figure shows that on the same potential function and same problem size, time taken for effective convergence by the MLHybrid algorithm is only around 28 seconds. It must be pointed out that one of the factors contributing to speed gain is the way invalid extreme bases are being handled. The flow graph created at each iteration handles a fixed number of (only $n(m-1)$) elementary extreme bases which span the space of all invalid extreme bases. The run-time at each iteration is essentially independent of the number of invalid extreme bases added by Wolfe's algorithm.

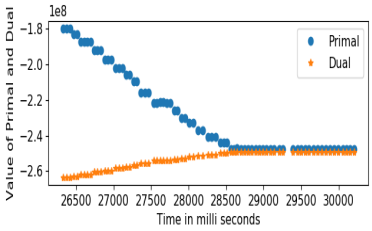

Fig. 6: Convergence of MLHybrid.

7 Conclusions

In this paper, we have proposed a new efficient inference algorithm for higher-order multi-label MRF-MAP problems, which enables obtaining optimal solution to such problems when potentials are submodular, and even when the cliques are of size upto 100 (for a 16 label problem). This has been made possible by exploiting the structure of the potentials used to make the extension function submodular. The min ℓ_2 norm solution to the block of invalid extreme bases can be found by max flow techniques on a particularly simple flow graph. What takes a series of max flow iterations in [45] requires only two linear time passes on the resultant flow graph.

References

1. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Süsstrunk, S.: Slic superpixels compared to state-of-the-art superpixel methods. PAMI **34**(11), 2274–2282 (2012)
2. Arora, C., Maheshwari, S.: Multi label generic cuts: Optimal inference in multi label multi clique MRF-MAP problems. In: CVPR. pp. 1346–1353 (2014)
3. Birchfield, S., Tomasi, C.: A pixel dissimilarity measure that is insensitive to image sampling. PAMI **20**(4), 401–406 (1998)
4. Boykov, Y., Veksler, O., Zabih, R.: Fast approximate energy minimization via graph cuts. IEEE Transactions on pattern analysis and machine intelligence **23**(11), 1222–1239 (2001)
5. Chakrabarty, D., Jain, P., Kothari, P.: Provable submodular minimization using wolfe's algorithm. In: NIPS. pp. 802–809 (2014)
6. Chen, L.C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: ECCV (2018)
7. Delong, A., Osokin, A., Isack, H.N., Boykov, Y.: Fast approximate energy minimization with label costs. International journal of computer vision **96**(1), 1–27 (2012)
8. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html
9. Felzenszwalb, P.F., Huttenlocher, D.P.: Efficient belief propagation for early vision. International journal of computer vision **70**(1), 41–54 (2006)
10. Fix, A., Gruber, A., Boros, E., Zabih, R.: A graph cut algorithm for higher-order markov random fields. In: ICCV. pp. 1020–1027 (2011)
11. Fix, A., Wang, C., Zabih, R.: A primal-dual algorithm for higher-order multilabel markov random fields. In: CVPR. pp. 1138–1145 (2014)
12. Freedman, D., Drineas, P.: Energy minimization via graph cuts: Settling what is possible. In: CVPR. pp. 939–946 (2005)
13. Fujishige, S., Hayashi, T., Isotani, S.: The minimum-norm-point algorithm applied to submodular function minimization and linear programming (2006)
14. Fujishige, S., Isotani, S.: A submodular function minimization algorithm based on the minimum-norm base. Pacific Journal of Optimization **7**, 3–17 (2011)
15. Gallagher, A.C., Batra, D., Parikh, D.: Inference for order reduction in Markov random fields. In: CVPR. pp. 1857–1864 (2011)
16. Globerson, A., Jaakkola, T.S.: Fixing max-product: Convergent message passing algorithms for map lp-relaxations. In: NIPS. pp. 553–560 (2008)
17. Gould, S.: Darwin: A framework for machine learning and computer vision research and development. JMLR **13**(Dec), 3533–3537 (2012)
18. Hazan, T., Shashua, A.: Norm-product belief propagation: Primal-dual message-passing for approximate inference. Information Theory **56**(12), 6294–6316 (2010)
19. Ishikawa, H.: Higher-order clique reduction without auxiliary variables. In: CVPR. pp. 1362–1369 (2014)
20. Ishikawa, H.: Exact optimization for Markov Random Fields with convex priors. PAMI **25**(10), 1333–1336 (2003)
21. Ishikawa, H.: Transformation of general binary MRF minimization to the first-order case. TPAMI **33**(6), 1234–1249 (2011)
22. Iwata, S., Fleischer, L., Fujishige, S.: A combinatorial strongly polynomial algorithm for minimizing submodular functions. JACM **48**(4), 761–777 (2001)

23. Jegelka, S., Bach, F., Sra, S.: Reflection methods for user-friendly submodular optimization. In: NIPS. pp. 1313–1321 (2013)
24. Kahl, F., Strandmark, P.: Generalized roof duality for pseudo-boolean optimization. In: ICCV. pp. 255–262 (2011)
25. Kappes, J.H., Andres, B., Hamprecht, F.A., Schnörr, C., Nowozin, S., Batra, D., Kim, S., Kausler, B.X., Kröger, T., Lellmann, J., Komodakis, N., Savchynskyy, B., Rother, C.: A comparative study of modern inference techniques for structured discrete energy minimization problems. IJCV **115**(2), 155–184 (2015)
26. Kohli, P., Torr, P.H., et al.: Robust higher order potentials for enforcing label consistency. IJCV **82**(3), 302–324 (2009)
27. Koller, D., Friedman, N., Bach, F.: Probabilistic graphical models: principles and techniques. MIT press (2009)
28. Kolmogorov, V.: Convergent tree-reweighted message passing for energy minimization. PAMI **28**(10), 1568–1583 (2006)
29. Kolmogorov, V.: Submodularity on a tree: Unifying l^1-convex and bisubmodular functions. In: International Symposium on Mathematical Foundations of Computer Science. pp. 400–411. Springer (2011)
30. Kolmogorov, V.: Minimizing a sum of submodular functions. Discrete Applied Mathematics **160**(15), 2246–2258 (2012)
31. Kolmogorov, V.: A new look at reweighted message passing. TPAMI **37**(5), 919–930 (2015)
32. Kolmogorov, V., Zabin, R.: What energy functions can be minimized via graph cuts? TPAMI **26**(2), 147–159 (2004)
33. Komodakis, N., Paragios, N.: Beyond pairwise energies: Efficient optimization for higher-order MRFs. In: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on. pp. 2985–2992. IEEE (2009)
34. McCormick, S.T.: Submodular function minimization (2005)
35. Murota, K.: On steepest descent algorithms for discrete convex functions. SIAM Journal on Optimization **14**(3), 699–707 (2004)
36. Orlin, J.B.: A faster strongly polynomial time algorithm for submodular function minimization. Mathematical Programming **118**(2), 237–251 (2009)
37. Pearl, J.: Probabilistic reasoning in intelligent systems: networks of plausible inference. Morgan Kaufmann (2014)
38. Potetz, B., Lee, T.S.: Efficient belief propagation for higher-order cliques using linear constraint nodes. CVIU **112**(1), 39–54 (Oct 2008)
39. Ramalingam, S., Russell, C., Ladicky, L., Torr, P.H.: Efficient minimization of higher order submodular functions using monotonic boolean functions. arXiv preprint arXiv:1109.2304 (2011)
40. Roth, S., Black, M.J.: Fields of experts. IJCV **82**(2), 205–229 (2009)
41. Rother, C., Kohli, P., Feng, W., Jia, J.: Minimizing sparse higher order energy functions of discrete variables. In: CVPR. pp. 1382–1389 (2009)
42. Scharstein, D., Szeliski, R.: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. IJCV **47**(1-3), 7–42 (2002)
43. Schrijver, A.: A combinatorial algorithm minimizing submodular functions in strongly polynomial time. Journal of Combinatorial Theory, Series B **80**(2), 346–355 (2000)
44. Schrijver, A.: Combinatorial optimization: polyhedra and efficiency, vol. 24. Springer Science & Business Media (2003)
45. Shanu, I., Arora, C., Maheshwari, S.: Inference in higher order mrf-map problems with small and large cliques. In: CVPR. pp. 7883–7891 (2018)

46. Shanu, I., Arora, C., Singla, P.: Min norm point algorithm for higher order MRF-MAP inference. In: CVPR. pp. 5365–5374 (2016)
47. Sontag, D., Globerson, A., Jaakkola, T.: Introduction to dual decomposition for inference. Optimization for Machine Learning **1**, 219–254 (2011)
48. Szeliski, R., Zabih, R., Scharstein, D., Veksler, O., Kolmogorov, V., Agarwala, A., Tappen, M., Rother, C.: A comparative study of energy minimization methods for Markov random fields with smoothness-based priors. TPAMI **30**(6), 1068–1080 (Jun 2008)
49. Tarlow, D., Givoni, I.E., Zemel, R.S.: HOP-MAP: Efficient message passing with high order potentials. In: AISTATS (2010)
50. Windheuser, T., Ishikawa, H., Cremers, D.: Generalized roof duality for multi-label optimization: Optimal lower bounds and persistency. In: European Conference on Computer Vision. pp. 400–413. Springer (2012)
51. Woodford, O., Torr, P., Reid, I., Fitzgibbon, A.: Global stereo reconstruction under second order smoothness priors. In: CVPR. pp. 1–8 (2008)
52. Yedidia, J.S., Freeman, W.T., Weiss, Y.: Generalized belief propagation. In: Advances in neural information processing systems. pp. 689–695 (2001)
53. Zhang, J., Djolonga, J., Krause, A.: Higher-order inference for multi-class log-supermodular models. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1859–1867 (2015)

Supplementary Material

A Proofs of Lemmas and Theorems

A.1 Proof of Theorem 1

Theorem. *The extended binary set function f as given by Definition 5 is submodular.*

Recall that we define the extended binary submodular function for the valid states as equal to the original multi-label function and for the invalid states as the following:

$$f(S) = f(\overline{S}) + (|\overline{S}| - |S|)L.$$

Here $\overline{S}$ is the minimum covering state for an invalid state, S, which is defined as the smallest cardinality valid state, $\overline{S} \in Z$, such that $S \subset \overline{S}$. For a valid state $S = \overline{S}$.

Let us factorize $f(S) = g(S) + h(S)$, where $h(S) = f(\overline{S}) + |\overline{S}|L$, and $g(S) = -|S|L$. Since, g is modular, it is sufficient to show that h is submodular. We will need the following result to prove the Theorem.

Lemma 10. *For sets X, Y, and $(X \cap Y) \subseteq \mathcal{V}$ and their minimum covering states $\overline{X}, \overline{Y}$, and $\overline{X \cap Y}$ respectively:*

$$f(\overline{X \cap Y}) \leq f(\overline{X} \cap \overline{Y}))$$

Proof. Recall that for any valid state S, $\overline{S} = S$. Consider, two valid states $A, B \subseteq V$ with $A \subseteq B$. It is easy to see that:

$$h(B) - h(A) = L(|B| - |A|) + f(B) - f(A) \geq 0. \tag{7}$$

Since, $A \subseteq B$, therefore, $|B| - |A| \geq 0$. Also $L \gg f(B) - f(A)$ by definition. Therefore, $h(A) \leq h(B)$. Further, it has been shown in section 6 of [43] that for

two $X, Y \in \mathcal{V}$, $\overline{X \cup Y} = (\overline{X} \cup \overline{Y})$ and $\overline{X \cap Y} \subseteq (\overline{X} \cap \overline{Y})$ holds. Therefore, using Eq. (7), $f(\overline{X \cap Y}) \leq f(\overline{X} \cap \overline{Y})$). □

We can now give the proof of the theorem as follows. For the valid states, the extended function f, has been shown to be submodular in [2]. Therefore, here, we show only for the cases when S is an invalid state. Now, for two arbitrary (valid or invalid) sets, $X, Y \subseteq \mathcal{V}$

$$
\begin{aligned}
h(X) + h(Y) &= f(\overline{X}) + f(\overline{Y}) + |\overline{X}|L + |\overline{Y}|L \\
&\geq f(\overline{X} \cup \overline{Y}) + f(\overline{X} \cap \overline{Y}) + |\overline{X}|L + |\overline{Y}|L \\
&\qquad \text{(Using submodularity over } \overline{X}\text{, and } \overline{Y}\text{)} \\
&= f(\overline{X} \cup \overline{Y}) + f(\overline{X} \cap \overline{Y}) + |\overline{X} \cup \overline{Y}|L + |\overline{X} \cap \overline{Y}|L \\
&\qquad \text{(Since } |\overline{X}| + |\overline{Y}| = |\overline{X} \cup \overline{Y}| + |\overline{X} \cap \overline{Y}|\text{)} \\
&= f(\overline{X \cup Y}) + |\overline{X \cup Y}|L + f(\overline{X} \cap \overline{Y}) + |\overline{X} \cap \overline{Y}|L \\
&\qquad \text{(Since } \overline{X \cup Y} = (\overline{X} \cup \overline{Y})\text{)} \\
&\geq f(\overline{X \cup Y}) + |\overline{X \cup Y}|L + f(\overline{X \cap Y}) + |\overline{X \cap Y}|L \\
&\qquad \text{(Using Lemma 10)} \\
&= h(X \cup Y) + h(X \cap Y).
\end{aligned}
$$

The above shows that h is submodular. It is easy to see that, g, as defined above is modular. Since addition of a modular function and a submodular function is submodular, therefore, $f = g + h$ is submodular.

A.2 Proof of Lemma 3

Lemma. *For any element, e, of an invalid extreme base, $b^{\prec} : b^{\prec}(e) = a_e L + b_e$, where $|a_e|, |b_e| \ll L$ and $a_e \in I$.*

Proof. Let S_2 be the set of all elements smaller than e as per $\prec$. Let $S_1 = S_2 \cup \{e\}$.

$$\begin{aligned} b^{\prec}(e) &= f(S_1) - f(S_2) && \text{(Definition of extreme base)} \\ &= \big(f(\overline{S}_1) + (|\overline{S}_1| - |S_1|)L\big) - \big(f(\overline{S}_2) + (|\overline{S}_2| - |S_2|)L\big) && \text{(Definition 5)} \\ &= \big(f(\overline{S}_1) - f(\overline{S}_2)\big) + \big(|\overline{S}_1| - |S_1| - |\overline{S}_2| + |S_2|\big)\, L \\ &= a_e L + b && \text{(where } |a_e|, |b_e| \ll L\text{)} \end{aligned}$$

□

A.3 Proof of Lemma 4

Lemma. *Consider two base vectors x_1 and x_2 such that $\|x_1\|^2, \|x_2\|^2 < |\mathcal{V}|M^2$. If $x_2 = (1-\lambda)x_1 + \lambda b^{\prec}$ and $b^{\prec}$ is an invalid extreme base, then $\lambda \leq |\mathcal{V}|\frac{M}{L}$.*

Proof. Recall that in our algorithm, base vector is represented as the sum of contributions from valid and invalid extreme bases separately: $x = x_v + x_i$, where x_v and x_i are the base vectors collecting contributions of valid and invalid extreme bases respectively. Further, we start from a valid extreme and in each iteration of the algorithm, keep on decreasing the norm of the overall base vector. Note that, all the elements of a valid extreme base are smaller than M. Therefore the squared ℓ_2 norm of the overall base vector is less than $|\mathcal{V}|M^2$ at any point in the algorithm.

We will prove the lemma by contradiction, and show that unless the λ for the invalid extreme base is less than $|\mathcal{V}|M/L$, the squared norm of the overall base vector is more than $|\mathcal{V}|M^2$, which is a contradiction.

We will first need to prove the following result:

Lemma 11. *Consider an invalid ordering $\prec$, and its corresponding invalid extreme base $b^{\prec}$. Let e be the smallest element (as per $\prec$), for which validity condition is violated. Then, $\exists\ a_e \in \mathbb{R}$, and $a_e \geq (1 - M/L)$, s.t. $b^{\prec}(e) = a_e L$.*

Proof. Let S_2 be the set of all elements smaller than e as per $\prec$. Let $S_1 = S_2 \cup \{e\}$. Notice that S_2 is a valid and S_1 is an invalid state.

$$
\begin{aligned}
b^{\prec}(e) &= f(S_1) - f(S_2) && \text{(Definition of extreme base)} \\
&= \big(f(\overline{S}_1) + (|\overline{S}_1| - |S_1|)L\big) - f(S_2) && \text{(Definition 5)} \\
&\geq \min_{S \in Z} f(S) - f(S_2) + (|\overline{S}_1| - |S_1|)L \\
&\qquad (\overline{S}_1 \text{ is a valid state, therefore } f(\overline{S}_1) \geq \min_{S \in Z} f(S)) \\
&\geq \min_{S \in Z} f(S) - \max_{S \in Z} f(S) + (|\overline{S}_1| - |S_1|)L \\
&\qquad (S_2 \text{ is a valid state, therefore } f(S_2) \leq \max_{S \in Z} f(S)) \\
&= (|\overline{S}_1| - |S_1| - M/L)L && \text{(Defintion of } M\text{)}
\end{aligned}
$$

Note that for any invalid state S_1, $(|\overline{S}_1| - |S_1|) \geq 1$. Therefore there exists $a_e \geq (1 - M/L)$ such that $b^{\prec}(e) = a_e L$. □

To prove our main result by contradiction, assume $\lambda > |\mathcal{V}|M/L$. Let e be the smallest element (as per $\prec$ of invalid extreme base $b^{\prec}$), for which validity condition is violated. Consider:

$$
\begin{aligned}
(x_2(e))^2 &= ((1-\lambda)x_1(e) + \lambda b^{\prec}(e))^2 \\
&> ((1-\lambda)x_1(e) + b^{\prec}(e)|\mathcal{V}|M/L)^2 && (\lambda > |\mathcal{V}|M/L) \\
&= ((1-\lambda)x_1(e) + a_e|\mathcal{V}|M))^2. && \text{(Using lemma 11)}
\end{aligned}
$$

Two cases are possible:

1. $x_1(e) \geq 0$:

$$
\begin{aligned}
(x_2(e))^2 &\geq ((1-\lambda)x_1(e) + a_e|\mathcal{V}|M))^2 \\
&\geq (a_e|\mathcal{V}|M)^2 && \text{(Since } (1-\lambda) \geq 0\text{)} \\
&= (a_e|\mathcal{V}|)(|\mathcal{V}|M^2)
\end{aligned}
$$

Since $M \ll L$, and $a_e \geq (1 - M/L)$, therefore $a_e \approx 1$. The smallest problem size that we consider is of 3 pixels and 2 labels for which $|\mathcal{V}| = 6$. Hence, for our case, $a_e|\mathcal{V}| > a_e\sqrt{|\mathcal{V}|} > 2$. This implies:

$$(x_2(e))^2 > |\mathcal{V}|M^2.$$

2. $x_1(e) < 0$:

Note that for x_1 and any element $e \in \mathcal{V}$ we have $x_1(e)^2 \leq \|x_1\|^2 \leq |\mathcal{V}|M^2$. This implies that $x_1(e) \geq -\sqrt{|\mathcal{V}|}M$.

$$\begin{aligned}
(x_2(e))^2 &\geq ((1-\lambda)x_1(e) + a_e|\mathcal{V}|M)^2 \\
&\geq (-(1-\lambda)\sqrt{|\mathcal{V}|}M + a_e|\mathcal{V}|M)^2 \\
&\geq (-\sqrt{|\mathcal{V}|}M + a_e|\mathcal{V}|M)^2 && \text{(Since } 1 \geq (1-\lambda) \geq 0\text{)} \\
&= M^2|\mathcal{V}|(a_e\sqrt{|\mathcal{V}|} - 1) && \text{(As described in the first case } a_e\sqrt{|\mathcal{V}|} > 2\text{)} \\
&> |\mathcal{V}|M^2.
\end{aligned}$$

Both the cases imply that if $\lambda > |\mathcal{V}|M/L$ then norm $||x_2||^2 > |\mathcal{V}|M^2$ which is a contradiction. Hence for any invalid extreme base its contribution λ in the overall base vector must be less than $|\mathcal{V}|M/L$. □

A.4 Proof of Lemma 5

Lemma. *Let $\prec$ be an invalid ordering and $\overline{\prec}$ be its canonical ordering. Then, $b^{\prec}(e) - b^{\overline{\prec}}(e) \ll L, \forall e \in \mathcal{V}$.*

Proof. Let S and S' be the set of elements preceding $p^i, p \in \mathcal{P}$, in ordering $\prec$ and $\overline{\prec}$ respectively. Consider the term $b^{\prec}(p^i)$:

$$\begin{aligned} b^{\prec}(p^i) &= f(S \cup \{p^i\}) - f(S) \\ &= L \sum_{q \in \mathcal{P}} \Big(|(\overline{S \cup \{p^i\}})_q| - |(S \cup \{p^i\})_q| \Big) + f\Big(\overline{S \cup \{p^i\}}\Big) \\ &\quad - L \sum_{q \in \mathcal{P}} \Big(|\overline{S}_q| - |S_q| \Big) - f\Big(\overline{S}\Big) \qquad \text{(Def. 6)} \\ &= L\Big(|\overline{S_p \cup \{p^i\}}| - |S_p \cup \{p^i\}|\Big) - L\Big(|\overline{S}_p| - |S_p|\Big) \\ &\quad + f(\overline{S \cup \{p^i\}}) - f(\overline{S}) \end{aligned}$$

Similarly we obtain:

$$\begin{aligned} b^{\overline{\prec}}(p^i) &= L\Big(|\overline{S'_p \cup \{p^i\}}| - |S'_p \cup \{p^i\}|\Big) - L\Big(|\overline{S'}_p| - |S'_p|\Big) \\ &\quad + f\Big(\overline{S' \cup \{p^i\}}\Big) - f\Big(\overline{S'}\Big) \end{aligned}$$

Note that a canonical ordering does not change intersay ordering between elements corresponding to a particular pixel. Therefore, $S_p = S'_p$, and:

$$b^{\prec}(p^i) - b^{\overline{\prec}}(p^i) = (f(\overline{S \cup \{p^i\}}) - f(\overline{S}) - f(\overline{S' \cup \{p^i\}}) + f(\overline{S'})$$

Since, all terms in the r.h.s. of the equation above, correspond to valid sets, therefore $b^{\prec}(p^i) - b^{\overline{\prec}}(p^i) \ll L$. □

A.5 Proof of Lemma 6

Lemma. *For a canonical invalid ordering $\overline{\prec}$, let p^i and p^j be two adjacent elements corresponding to a pixel p, s.t. $p^i \overline{\prec} p^j$. Let $\overline{\prec}_p^{i,j}$ be the ordering obtained by swapping p^i and p^j. Then $b^{\overline{\prec}_p^{i,j}} - b^{\overline{\prec}} = (\chi_p^j - \chi_p^i)(aL + b)$, where χ_p^i is an indicator vector for the element p^i, and $a, b \ll L$.*

Proof. Recall that for an extreme base $b^{\prec} : b^{\prec}(k) = f(k_{\prec}) - f((k-1)_{\prec})$, where $k_{\prec}$ is the first k elements in the ordered set $\{v_1, \ldots, v_k, \ldots, v_n\}$. Since the swap

between p^i, and p^j leaves the set of preceding elements unchanged for all other elements, therefore, $b^{\overline{\prec}_p^{i,j}} - b^{\overline{\prec}}$ is non-zero corresponding to only p^i, and p^j.

Let S be the set of elements preceding p^i in $\overline{\prec}$. Now:

$$\begin{aligned} b^{\overline{\prec}}(p^j) &= f(S \cup \{p^j\} \cup \{p^i\}) - f(S \cup \{p^i\}) \\ &= L(|\overline{S \cup \{p^j\} \cup \{p^i\}}| - |S \cup \{p^j\} \cup \{p^i\}|) + f(\overline{S \cup \{p^j\} \cup \{p^i\}}) \\ &\quad - L(|\overline{S \cup \{p^i\}}| - |S \cup \{p^i\}|) - f(\overline{S \cup \{p^i\}}) \end{aligned} \tag{8a}$$

Similarly we have,

$$b^{\overline{\prec}_p^{i,j}}(p^j) = L(|\overline{S \cup \{p^j\}}| - |S \cup \{p^j\}|) - L(|\overline{S}| - |S|) + f(\overline{S \cup \{p^j\}}) - f(\overline{S}), \tag{8b}$$

Subtracting Eq. (8a) from Eq. (8b) and using $(|S \cup \{p^i\}| + |S \cup \{p^j\}| - |S \cup \{p^i\} \cup \{p^j\}| - |S|) = 0$ we have

$$\begin{aligned} b^{\overline{\prec}_p^{i,j}}(p^j) - b^{\overline{\prec}}(p^j) &= L(|\overline{S \cup \{p^i\}}| + |\overline{S \cup \{p^j\}}| - |\overline{S \cup \{p^i\} \cup \{p^j\}}| - |\overline{S}|) \\ &\quad + (f(\overline{S \cup \{p^j\}}) - f(\overline{S}) - f(\overline{S \cup \{p^j\} \cup \{p^i\}}) + f(\overline{S \cup \{p^i\}}), \\ &= aL + b, \end{aligned}$$

where:

$$\begin{aligned} a &= |\overline{S \cup \{p^i\}}| + |\overline{S \cup \{p^j\}}| - |\overline{S \cup \{p^i\} \cup \{p^j\}}| - |\overline{S}|, \text{ and} \\ b &= (f(\overline{S \cup \{p^j\}}) - f(\overline{S}) - f(\overline{S \cup \{p^j\} \cup \{p^i\}}) + f(\overline{S \cup \{p^i\}}). \end{aligned}$$

Note that b is sum of function values at valid states and is $\ll L$. Two cases arise for the value of a:

1. $i < j$:
   In this case $|\overline{S_p \cup \{p^i\} \cup \{p^j\}}| = |\overline{S_p \cup \{p^j\}}|$, and $a = |\overline{S \cup \{p^i\}}| - |\overline{S}|$. Therefore, $a \ll L$.

2. $j < i$:
   In this case $|\overline{S_p \cup \{p^i\} \cup \{p^j\}}| = |\overline{S_p \cup \{p^i\}}|$, and $a = |\overline{S \cup \{p^j\}}| - |\overline{S}|$. Therefore, $a \ll L$

Hence $b^{\overline{\prec}_p^{i,j}}(p^j) - b^{\overline{\prec}}(p^j) = aL + b$, such that $a, b \ll L$. Further, since $b^{\overline{\prec}_p^{i,j}}$ and $b^{\overline{\prec}}$ are extreme bases, and the sum of all the elements in them is constant, therefore, the reverse must hold for $b^{\overline{\prec}_p^{i,j}}(p^i) - b^{\overline{\prec}}(p^i)$. Hence $b^{\overline{\prec}_p^{i,j}} - b^{\overline{\prec}} = (\chi_p^j - \chi_p^i)(aL + b)$ □

A.6 Proof of Lemma 7

Lemma. *Consider an elementary invalid extreme base $b^{\widetilde{\prec}_p^i}$, obtained by swapping two adjacent elements (p^{i+1}, p^i) in the universal ordering, $\prec_0$ (Def. 2). Then:*

$$b^{\widetilde{\prec}_p^i} - b^{\prec_0} = (\chi_p^i - \chi_p^{i+1})(L + b),$$

where $b^{\prec_0}$ is the valid extreme base corresponding to $\prec_0$.

Proof. Recall:

- The universal ordered sequence, $\prec_0$, which is a valid ordering, and also defines a particular ordering among the pixels.
- The elementary invalid ordering, $\widetilde{\prec}$, which is defined as the ordering obtained by making one swap between adjacent elements of a valid ordering. The corresponding extreme base is denoted as $b^{\widetilde{\prec}}$.

Further, recall from Section A.5 where while proving Lemma 6, we showed that: $b^{\overline{\prec}_p^{i,j}} - b^{\overline{\prec}} = (\chi_p^j - \chi_p^i)(aL + b)$, such that $a = |\overline{S \cup \{p^i\}}| - |\overline{S}|$ (if $i < j$), or $a = |\overline{S \cup \{p^j\}}| - |\overline{S}|$ (if $j < i$). Now consider an elementary invalid extreme base $b^{\widetilde{\prec}_p^i}$, obtained by swapping two adjacent elements (p^{i+1}, p^i) in the universal ordering. The term $(\chi_p^i - \chi_p^{i+1})$ may be looked upon as corresponding to the creation of the elementary extreme base $b^{\widetilde{\prec}_p^i}$ from $b^{\prec_0}$. It is easy to see that for such special elementary invalid extreme bases created from universal ordering,

$a = 1$, and we have:

$$b^{\widetilde{\prec}^i_p} - b^{\prec_0} = (\chi^i_p - \chi^{i+1}_p)(L + b) \tag{9}$$

Hence, proved. □

A.7 Proof of Lemma 8

Lemma. *An invalid canonical extreme base, $b^{\overline{\prec}}$, can be represented as a linear combination of elementary invalid extreme base vectors such that:*

$$b^{\overline{\prec}} = \sum_{p\in\mathcal{P}} \sum_{i=1}^{m-1} \alpha^i_p b^{\widetilde{\prec}^i_p} + \Lambda,$$

where $0 < \alpha^i_p \ll L$, and Λ is a vector with all its elements much smaller than L.

Proof. Consider the canonical invalid ordering $\overline{\prec}$ and let $\prec_s$ be the starting canonical valid ordering from which it can be obtained by a series of swaps between adjacent elements. Note that since in the canonical ordering all the elements of a pixel are already together, therefore all the swaps required are between elements corresponding to same pixels. Let us assume that total number of such swaps required are T. Starting from $\prec_s$, let $\prec_j$ represents the ordering obtained after j such swaps. Hence, $\prec_T = \overline{\prec}$ by definition. Let j^{th} swap happens between elements p^{k_j} and p^{l_j}, where $p \in \mathcal{P}$.

$$\begin{aligned}
b^{\overline{\prec}} - b^{\prec_s} &= b^{\prec_T} - b^{\prec_s} \\
&= \sum_{j=1}^{T} \left(b^{\prec_j} - b^{\prec_{j-1}} \right) \\
&= \sum_{j=1}^{T} \left(\chi^{l_j}_p - \chi^{k_j}_p \right)(a_j L + b_j) && \text{(Using Lemma 6)} \\
&= \sum_{j=1}^{T} (\chi^{l_j}_p - \chi^{k_j}_p) a_j L + \sum_{j=1}^{T} (\chi^{l_j}_p - \chi^{k_j}_p) b_j.
\end{aligned}$$

Since $(\chi_p^{l_j} - \chi_p^{k_j}) = \sum_{i=l_j}^{k_j}(\chi_p^i - \chi_p^{i+1})$ we can write:

$$b^{\overline{\prec}} - b^{\prec_s} = \sum_{j=1}^{T} a_j \sum_{i=l_j}^{k_j}(\chi_p^i - \chi_p^{i+1})L + \sum_{j=1}^{T} b_j \sum_{i=l_j}^{k_j}(\chi_p^i - \chi_p^{i+1}) \tag{10}$$

Recall from Lemma 7:

$$\begin{aligned} & b^{\widetilde{\prec}_p^i} - b^{\prec_0} = (\chi_p^j - \chi_p^{i+1})(L + b) \\ \Rightarrow \quad & (\chi_p^i - \chi_p^{i+1})L = b^{\prec_p^i} - b^{\prec_0} - (\chi_p^i - \chi_p^{i+1})b_p^i. \qquad \text{(where } b_p^i \ll L\text{)} \end{aligned}$$

Substituting the value of $(\chi_p^i - \chi_p^{i+1})L$ in Eq. (10), we get:

$$b^{\overline{\prec}} - b^{\prec_s} = \sum_{j=1}^{T} a_j \sum_{i=l_j}^{k_j}(b^{\widetilde{\prec}_p^i} - b^{\prec_0} - (\chi_p^i - \chi_p^{i+1})b_p^i) + \sum_{j=1}^{T} b_j \sum_{i=l_j}^{k_j}(\chi_p^i - \chi_p^{i+1})$$

Since both $\prec_s$, and $\prec_0$ are valid orderings, we can write $b^{\prec_s} = b^{\prec_0} + \vec{d}$, where elements of $\vec{d}$ are much smaller than L. Therefore we get

$$b^{\overline{\prec}} = \sum_{j=1}^{T}\sum_{i=l_j}^{k_j} a_j b^{\widetilde{\prec}_p^i} + \Big(1 - \sum_{j=1}^{T}\sum_{i=l_j}^{k_j} a_j\Big) b^{\prec_0} - \sum_{j=1}^{T}\sum_{i=l_j}^{k_j} a_j(\chi_p^i - \chi_p^{i+1})b_p^i \tag{11}$$

$$+ \sum_{j=1}^{T}\sum_{i=l_j}^{k_j}(\chi_p^i - \chi_p^{i+1})b_j + \vec{d}$$

$$b^{\overline{\prec}} = \sum_{j=1}^{T}\sum_{i=l_j}^{k_j} a_j b^{\widetilde{\prec}_p^i} + \Lambda, \tag{12}$$

where Equation (12) has been derived summing the last 4 terms into a vector Λ. Note that all the elements of Λ are $\ll L$. It is easy to see that the first term in the equation essentially is a linear combination of some elementary invalid extreme bases, allowing us to simplify:

$$b^{\overline{\prec}} = \sum_{p \in P} \sum_{i=1}^{m-1} \alpha_p^i b^{\widetilde{\prec}_p^i} + \Lambda, \tag{13}$$

where coefficients α_p^i corresponding to elementary extreme bases not present in Equation (12) can be simply set to zero. □

A.8 Proof of Lemma 9

Theorem. *An invalid extreme base can be represented as* $b^{\prec} = \sum_{p \in \mathcal{P}} \sum_{i=1}^{m-1} \alpha_p^i L(\chi_p^i - \chi_p^{i+1}) + \Lambda$*, where* χ_p^i *is an indicator vector corresponding to element* p^i*,* $0 < \alpha_p^i \ll L$*, and* Λ *is some vector whose all elements are* $\ll L$*.*

Proof. Using Equation (13), we have:

$$b^{\overline{\prec}} = \sum_{p \in P} \sum_{i=1}^{m-1} \alpha_p^i b^{\widetilde{\prec}_p^i} + \Lambda.$$

Substituting representation of elementary extreme base from Equation (where $b_p^i \ll L$), we have:

$$\begin{aligned} b^{\prec} &= \sum_{p \in P} \sum_{i=1}^{m-1} \alpha_p^i \Big(b^{\prec_0} + (\chi_p^i - \chi_p^{i+1})(L + b_p^i) \Big) + \Lambda. \\ &= \sum_{p \in P} \sum_{i=1}^{m-1} \alpha_p^i b^{\prec_0} + \sum_{p \in P} \sum_{i=1}^{m-1} \alpha_p^i L(\chi_p^i - \chi_p^{i+1}) + \sum_{p \in P} \sum_{i=1}^{m-1} (\chi_p^i - \chi_p^{i+1}) \alpha_p^i b_p^i + \Lambda. \\ &= \sum_{p \in P} \sum_{i=1}^{m-1} \alpha_p^i L(\chi_p^i - \chi_p^{i+1}) + \Lambda. \end{aligned}$$

Note that we have replaced Λ with $\sum_{p \in P} \sum_{i=1}^{m-1} \alpha_p^i b^{\prec_0} + \sum_{p \in P} \sum_{i=1}^{m-1} (\chi_p^i - \chi_p^{i+1}) \alpha_p^i b_p^i + \Lambda$. □

A.9 Proof of Theorem 2

Theorem (Main Result)**.**

$$\sum_{\forall b^{\prec_i} \in Q} \lambda_i b^{\prec_i} = \sum_{p \in \mathcal{P}} \sum_{k=1}^{m-1} \beta_p^k L(\chi_p^k - \chi_p^{k+1}),$$

where $\lambda_i \geq 0$, $\beta_p^k = \sum_{b_i \in Q} \alpha_p^k \lambda_i$.

Consider the expansion of the term $x_i = \sum_{b^{\prec_i} \in Q} \lambda_i b^{\prec_i}$ in Eq.(5). Using Theorem (9) we get:

$$x_i = \sum_{b^{\prec_i} \in Q} \sum_{p \in \mathcal{P}} \sum_{k=1}^{m-1} \lambda_i \alpha_p^k L(\chi_p^k - \chi_p^{k+1}) + \sum_{b^{\prec_i} \in Q} \lambda_i \Lambda_i.$$

Recall from Lemma (4) that, for all $b^{\prec_i} \in Q$, the coefficient λ_i can be made arbitrarily small. Therefore, we can drop the term $\sum_{b^{\prec_i} \in Q} \lambda_i \Lambda_i$ and rewrite the above equation as:

$$x_i = \sum_{p \in P} \sum_{k=1}^{m-1} \sum_{b^{\prec_i} \in Q} \lambda_i \alpha_p^k L(\chi_p^k - \chi_p^{k+1}).$$

Replacing by $\beta_p^k = \sum_{b^{\prec_i} \in Q} \lambda_i \alpha_p^k$, we get:

$$x_i = \sum_{p \in P} \sum_{k=1}^{m-1} \beta_p^k L(\chi_p^k - \chi_p^{k+1}).$$

B Example for validating importance of invalid extreme bases

In this section we show that invalid extreme bases contribute to the representation of optimal vector x^*. We consider here small problem with only 2 pixels (p and q) with 3 labels. We consider the unary cost for labeling pixel p as [0, 1, -100]. Similarly assume unary cost for q as [0, -100, 200]. The clique potential is absolute difference between labels and $L = 1000$. It may be noted that in the proposed

encoding we showed conceptually that label for a pixel could be encoded using m binary elements. However, notice that the state of the last element corresponding to each encoding is always 1. Therefore, implementation-wise, one can encode label at each pixel using $m-1$ binary elements only, with the assumption that elements $p^m, \forall p \in \mathcal{P}$ have their labeling set to $1 : p^m = 1, \forall p \in \mathcal{P}$. Hence, in this section we work with the extreme bases of dimension 4, which is corresponding to 2 pixels and 2 binary elements (p^1, and p^2, and no p^3) per pixel only. In the following subsection we first compute the optimal minimizer x^* using all valid and invalid extreme bases.

B.1 Using All Valid and Invalid Extreme Bases

There are $4! = 24$, extreme bases for the example problem. We list below, all valid and invalid extreme base vectors:

$(\ 902, -1000,\ 1198, -1000)$	$(\ 902, -1000,\ 299, -101)$	$-(\ 902, -1000,\ 1198, -1000)$
$(\ 902, -1000,\ 1198, -1000)$	$(\ 902, -1000,\ 299, -101)$	$-(\ 902, -1000,\ 299, -101)$
$(-100,\ 2,\ 1198, -1000)$	$(-100,\ 2,\ 299, -101)$	$-(-102,\ 2,\ 1200, -1000)$
$(-102,\ 2,\ 1200, -1000)$	$(-100,\ 2,\ 299, -101)$	$-(-102,\ 2,\ 301, -101)$
$(\ 898, -1000, 1202, -1000)$	$-(\ 898, -1000, 1202, -1000)$	$-(-102,\ 0, 1202, -1000)$
$(-102,\ 0, 1202, -1000)$	$(\ 898, -1000, 1202, -1000)$	$-(-102,\ 0, 1202, -1000)$
$(\ 900, -1000,\ 299, -99)$	$-(\ 900, -1000,\ 299, -99)$	$-(-100,\ 0,\ 299, -99)$
$(-102,\ 0,\ 301, -99)$	$-(\ 898, -1000,\ 301, -99)$	$-(-102,\ 0,\ 301, -99)$

The values of corresponding lambda obtained for the optimal minimum norm point (x^*) in the convex hull of all the extreme points are as given below:

(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0.0250, 0.0250, 0.949, 0, 0, 0).

Observe that, as described in the main paper, the contribution of invalid extreme bases in the base vector is still finite and not dependent upon the L. Corresponding to the above convex combination, we get the optimal base vector as $x^* = (-50, -50, 299, -99)$. Below, we show that $x*$ can not be represented as the convex combination of valid extreme bases only.

B.2 Considering Only Valid Extreme Points

The 6 valid extreme base vectors corresponding to the example problem are given below:

$(-100,\ 2,\ 299,\ -101)$	$(-100,\ 2,\ 299,\ -101)$	$(-102,\ 2,\ 301,\ -101)$
$(-100,\ 0,\ 299,\ -99)$	$(-102,\ 0,\ 301,\ -99)$	$(-102,\ 0,\ 301,\ -99)$

Note that the first two elements of x^* are -50 and -50 which can never be represented as the convex combination of first two elements of only valid extreme points.

C Convergence of SoSMNP [46]

Our focus initially is to show the convergence to the optimal solution by the MNP algorithm running in the block co-ordinate descent mode as in [46]. The problem formally is to minimize the function $f(S) = \sum_{\mathbb{c} \in \mathcal{C}} f_{\mathbb{c}}(S \cap \mathbb{c}) \quad S \subseteq \mathcal{V}$, where $f_{\mathbb{c}} : 2^{|\mathbb{c}|} \rightarrow \mathcal{R}$ is a submodular function. It has been shown in [46] that f can be minimized by finding a point $x \in B(f)$ with the minimum ℓ_2-norm $\|x\|^2$. We write x as the sum $x = \sum_{\mathbb{c} \in \mathcal{C}} y_{\mathbb{c}}$ where $y_{\mathbb{c}} \in B(f_{\mathbb{c}})$.

We assume that a block corresponds to a clique in $\mathcal{C}$. Let $x_{\mathbb{c}}$ be the restriction of x to the elements in $\mathbb{c} \in \mathcal{C}$, and let $x_{\not\mathbb{c}}$ be the restriction of x on the remaining

elements. We can write $\|x\|^2 = \|x_{\mathbb{c}}\|^2 + \|x_{\not\mathbb{c}}\|^2$. The block co-ordinate descent algorithm in [46] minimizes $\|x_{\mathbb{c}}\|^2$ using MNP over all the cliques $\mathbb{c} \in \mathcal{C}$ cyclically. This norm minimization step can be viewed as MNP minimizing $f'_{\mathbb{c}}(S) = f_{\mathbb{c}}(S) + a_{\mathbb{c}}(S), \forall S \subseteq \mathbb{c}$ where $a_{\mathbb{c}} = x_c - y_c$, is a denoting the contribution of the other cliques which remains constant while running MNP over this clique/block. Note that $a_{\mathbb{c}}(S) = \sum_{e \in S} a_{\mathbb{c}}(e)$, and we can equivalently treat $a_{\mathbb{c}}$ as a modular function as well. Let $f'_{\mathbb{c}}(S) = f_{\mathbb{c}}(S) + a_{\mathbb{c}}(S), \forall S \subseteq \mathbb{c}$. Note that the f' as shown above is a sum of submodular (f), and a modular function ($a_{\mathbb{c}}$). Therefore, f' is submodular. It is easy to show the following result:

Lemma 12. *Let $q_{\mathbb{c}}$ be a extreme base vector in $B(f_{\mathbb{c}})$ corresponding to an ordering $\prec_{\mathbb{c}}$. Then the vector $q_{\mathbb{c}} + a_{\mathbb{c}}$ is an extreme base of $B(f'_{\mathbb{c}})$ corresponding to the same ordering $\prec_{\mathbb{c}}$.*

Proof. We can calculate the elements in extreme base vector ($q'_{\mathbb{c}} \in B(f')$) corresponding to ordering $\prec_{\mathbb{c}}$ by Edmond's Greedy Algorithm,

$$
\begin{aligned}
q'(e) &= f'(S_e \cup e) - f'(S_e), \qquad (S_e \text{ is the set of elements before } e \in \mathbb{c} \text{ in } \prec_{\mathbb{c}}.) \\
&= f(S_e \cup e) + a_{\mathbb{c}}(S_e \cup e) - (f(S_e) + a_{\mathbb{c}}(S_e)), \\
&= f(S_e \cup e) + a_{\mathbb{c}}(S_e) + a_{\mathbb{c}}(e) - (f(S_e) + a_{\mathbb{c}}(S_e)), \\
&\qquad\qquad (a_{\mathbb{c}} \text{ can be seen as a modular function.}) \\
&= f(S_e \cup e) - f(S_e) + a_{\mathbb{c}}(e), \\
&= q_{\mathbb{c}}(e) + a_{\mathbb{c}}(e). \qquad\qquad (\text{By Edmond's Greedy Algorithm.})
\end{aligned}
$$

Hence, $q'_{\mathbb{c}} = q_{\mathbb{c}} + a_{\mathbb{c}}$. □

It is easy to see that:

$$\begin{aligned} x_{\mathbb{c}} = y_{\mathbb{c}} + a_{\mathbb{c}} &= \sum_i \lambda_i q_{\mathbb{c}} + a_{\mathbb{c}} && \text{(where } \textstyle\sum_i \lambda_i = 1, \text{ and } \lambda_i \text{ ¿ } 0) \\ &= \sum_i \lambda_i q_{\mathbb{c}} + \sum_i \lambda_i a_{\mathbb{c}} && \text{(Since } \textstyle\sum_i \lambda_i = 1) \\ &= \sum_i \lambda_i (q_{\mathbb{c}} + a_{\mathbb{c}}) = \sum_i \lambda_i q'_{\mathbb{c}} \end{aligned}$$

Hence, $x_{\mathbb{c}}$ is a base vector of f'. Therefore, minimizing the minimum norm over a block, the way SoSMNP does it, can be seen as minimizing the norm of $x_{\mathbb{c}}$: the restriction of x over the elements of clique $\mathbb{c}$ (and not $y_{\mathbb{c}}$). Let us suppose, we have reached a situation where the SoSMNP performs minimization over all blocks (cliques), and no change was observed in any of the blocks. The following lemma establishes the relationship between the extreme base of $f_{\mathbb{c}}$, and the one corresponding to f.

Lemma 13. *Let* $q_{\mathbb{c}} = \arg\min_{q \in B(f_{\mathbb{c}})} x_{\mathbb{c}}^T q, \ \forall \mathbb{c} \in \mathcal{C}$. *Then* $b = \sum_{\mathbb{c} \in \mathcal{C}} q_{\mathbb{c}}$ *also satisfies* $b = \arg\min_{b \in B(f)} x^T b$.

Proof. In the SoSMNP algorithm, the extreme base $q_{\mathbb{c}}$ is generated using Edmond's Greedy Algorithm [43] on the order $\prec_{\mathbb{c}}$ of the indices obtained by sorting the elements of $x_{\mathbb{c}}$ in the increasing order. We represent the extreme base so obtained by $q_{\mathbb{c}}^{\prec_{\mathbb{c}}}$. The SoSMNP algorithm for a block terminates when $x_{\mathbb{c}}^T x_{\mathbb{c}} = x_{\mathbb{c}}^T (q_{\mathbb{c}}^{\prec_{\mathbb{c}}} + a_{\mathbb{c}})$

Consider the termination situation of SoSMNP for the overall problem (comprising of all the cliques). In such a case the algorithms tries to minimize for all the blocks/cliques and no change is found on any of the cliques. Therefore, termination condition of each block is met, and $q_{\mathbb{c}}^{\prec_{\mathbb{c}}} = \arg\min_{q \in B(f_{\mathbb{c}})} x_{\mathbb{c}}^T q$.

Let $\prec_f$ be the ordering of elements of x in the increasing order. It is easy to see that the ordering over x and $x_{\mathbb{c}}$ will be consistent with each other, in the sense that $x(e_1) \prec_f x(e_2) \Rightarrow x_{\mathbb{c}}(e_1) \prec_{\mathbb{c}} x_{\mathbb{c}}(e_2)$.

Let us create an extreme base of f, corresponding to the ordering $\prec_f$, and denote as $b^{\prec_f}$. Since $\prec_f$ denotes the ordering over elements of x, therefore, from

Edmond's algorithm, we have: $b^{\prec_f} = \arg\min_{b \in B(f)} x^T b$. Further, we also have:

$$b^{\prec_f}(e) = f(S_e \cup e) - f(S_e),$$

(As per Edmond's algorithm. S_e is the set of elements before e in $\prec_f$)

$$= \sum_{\mathbb{c} \in \mathcal{C}} f_{\mathbb{c}}(S_e \cup e \cap \mathbb{c}) - f(S_e \cap \mathbb{c}), \qquad \text{(Since } f(S) = \textstyle\sum_{\mathbb{c} \in \mathcal{C}} f_{\mathbb{c}}(S \cap \mathbb{c})\text{)}$$

$$= \sum_{\mathbb{c} \in \mathcal{C}} q_{\mathbb{c}}^{\prec_{\mathbb{c}}}(e \cap \mathbb{c}). \qquad \text{(Since } \prec_c \text{ is the restriction of } \prec\text{)}$$

Since above holds for all the elements $e \in \mathcal{V}$, therefore:

$$b^{\prec_f} = \sum_{\mathbb{c} \in \mathcal{C}} q_{\mathbb{c}}^{\prec_{\mathbb{c}}}.$$

Hence, we have proved both the properties of $b^{\prec_f}$ □

We can now give the convergence proof of the SoSMNP with the following lemma:

Lemma 14. *If in a complete cycle of SoSMNP over all the cliques, we can not improve the norm $x_{\mathbb{c}}$ for any $\mathbb{c}$, then we have $x \in B(f)$ such that $\|x\|^2 = x^T x = x^T b$, where $b = \arg\min_{b \in B(f)} x^T b$.*

Proof. Recall that for a clique $\mathbb{c}$, SoSMNP can be seen as minimizing the norm of $x_{\mathbb{c}}$ which is a base vector of $f'_{\mathbb{c}} = f_{\mathbb{c}} + a_{\mathbb{c}}$. Further $q'_{\mathbb{c}} = q_{\mathbb{c}} + a_{\mathbb{c}}$ is an extreme base of f'. Therefore, from the termination of basic MNP algorithm, the following must hold:

$$x_{\mathbb{c}}^T x_{\mathbb{c}} = x_{\mathbb{c}}^T (q_{\mathbb{c}} + a_{\mathbb{c}}). \qquad (q_{\mathbb{c}} = \arg\min_{q \in B(f_{\mathbb{c}})} x_{\mathbb{c}}^T q)$$

Summing over all the cliques we get

$$\begin{aligned}
\sum_{\mathbb{c}\in\mathcal{C}} x_{\mathbb{c}}^T x_{\mathbb{c}} &= \sum_{\mathbb{c}\in\mathcal{C}} x_{\mathbb{c}}^T (q_{\mathbb{c}} + a_{\mathbb{c}}), \\
\sum_{\mathbb{c}\in\mathcal{C}} x_{\mathbb{c}}^T (y_{\mathbb{c}} + a_{\mathbb{c}}) &= \sum_{\mathbb{c}\in\mathcal{C}} x_{\mathbb{c}}^T (q_{\mathbb{c}} + a_{\mathbb{c}}), \\
\sum_{\mathbb{c}\in\mathcal{C}} x_{\mathbb{c}}^T y_{\mathbb{c}} &= \sum_{\mathbb{c}\in\mathcal{C}} x_{\mathbb{c}}^T q_{\mathbb{c}}. \qquad (\textstyle\sum_{\mathbb{c}\in\mathcal{C}} x_{\mathbb{c}}^T a_{\mathbb{c}} \text{ cancels out})
\end{aligned}$$

Since vector $y_{\mathbb{c}}$ and $q_{\mathbb{c}}$ have non-zero values only for elements in $\mathbb{c}$. Therefore we can write $x_{\mathbb{c}}^T y_{\mathbb{c}} = x^T y_{\mathbb{c}}$ and $x_{\mathbb{c}}^T q_{\mathbb{c}} = x^T q_{\mathbb{c}}$. Substituting the values, we get:

$$\begin{aligned}
\sum_{\mathbb{c}\in\mathcal{C}} x^T y_{\mathbb{c}} &= \sum_{\mathbb{c}\in\mathcal{C}} x^T q_{\mathbb{c}}, \\
x^T \sum_{\mathbb{c}\in\mathcal{C}} y_{\mathbb{c}} &= x^T \sum_{\mathbb{c}\in\mathcal{C}} q_{\mathbb{c}} \\
x^T x &= x^T b \qquad (\text{where } b = \arg\min_{b\in B(f)} x^T b, \text{ by Lemma 13})
\end{aligned}$$

The equation above is the termination condition of basic MNP when run over the overall function f [5]. Therefore, the lemma essentially proves that the basic MNP terminating with optimal solutions for all cliques/blocks implies that the the base vector obtained by summing up the base vectors of all the cliques/blocks is the optimal solution for the overall objective function. □

When MNP algorithm is run in the block co-ordinate descent mode it is easy to show that any decrease in the $\|x_c\|^2$ of a clique decreases the over all $\|x\|^2$ by the same amount because $\|x_{\not c}\|$ is untouched when optimizing for $\mathbb{c}$. Since at each cycle there is at least one clique for which $\|x_c\|^2$ decreases, we can say that $\|x\|^2$ decreases monotonically at each cycle. Note that Theorem 4 of [5] gives us a lower bound on the improvement in every MNP iteration. It follows that MNP algorithm running in block co-ordinate descent mode will have a provable rate of convergence.

For the sake of completeness we will also like to point out that the optimal solution obtained when MNP is run globally also corresponds to the individual blocks having reached their local optima.

D Convergence of ML-hybrid Algorithm

Note that in SoSMNP each block is optimized using the MNP algorithm. In MLhybrid, on the other hand, each block is further subdivided. One corresponds to the set of valid extreme bases (the valid block) and the other to the set of invalid extreme bases (the invalid block) whose convex combination defines the base vector $x_{\mathfrak{c}}$. MNP is run on the valid block. If at any iteration MNP [13] inserts an invalid extreme base, the flow based Algorithm 2 is run on the invalid block. We show below that when MNP is run on the valid block now (that is just after a run of the flow based algorithm on the invalid block) the extreme base generated will be valid.

Lemma 15. *Algorithm 2 returns a vector $x_{\mathfrak{c}}$ for clique $\mathfrak{c}$ such that extreme base q_c given as $q_c = \arg\min_{q \in B(f_e)} x_{\mathfrak{c}}^T q$ is valid.*

Proof. It is easy to show that when Algorithm 2 terminates, for any pair of indices i, j corresponding to any $p \in \mathcal{P}$ if $i > j$ then $e(p^i) \leq e(p^j)$. Note that by construction the excess vector e is the base vector $x_{\mathfrak{c}}$. This implies that the order $\prec_{\mathfrak{c}}$ of the indices obtained by sorting the elements of $x_{\mathfrak{c}}$ will satisfy $p^i \prec_{\mathfrak{c}} p^j, \forall i > j$, and $\forall p \in \mathcal{P}$. This is the condition that has to be satisfied for an ordering to be valid (Cf. Def. 3). Recall that in the MNP algorithm the extreme base is found by computing the ordering of sorted elements of x. Hence, the extreme base $q_c = \arg\min_{q \in B(f_e)} x_{\mathfrak{c}}^T q$ will be a valid one. □

Lemma 15 implies that an iteration on the invalid block will be followed by the MNP algorithm making progress in the form of generation of a valid extreme base. Also note that the ℓ_2 norm decreases when the flow based algorithm is run on the invalid block. Therefore, termination and convergence of the MLhybrid

algorithm running on a clique/block follows along the same lines as that for the standard MNP algorithm [5].

Now we show that termination over a clique/block results in $x_{\mathbb{c}}$ using which minimizer obtained comes on a valid state.

Note that generation of an invalid extreme base can always be followed by generation of a valid extreme base (by running the flow based algorithm on the invalid block). Therefore, at termination it is guaranteed that the order $\prec_{\mathbb{c}}$ of the indices obtained by sorting the elements of $x_{\mathbb{c}}$ is valid. That is the optimal solution corresponds to a valid primal state. Hence, it follows, using Lemma 14, that the MLHybrid algorithm run in the block coordinate descent manner converges to the optimal.

E Proposed Complete Algorithm

In this section we give the complete proposed method in Algorithm 3. The algorithm takes tranformed 2-label submodular clique potentials $f_{\mathbb{c}}$'s and computes minimum ℓ_2 norm of $x \in B(f)$ s.t. $f = \sum_{\mathbb{c} \in \mathcal{C}} f_{\mathbb{c}}$. The overall algorithm solves valid block with the SoS-MNP algorithm given in [46] and uses Algorithm 1 to solve invalid block. Let $x_{\mathbb{c}}$ be the restriction of x over clique $\mathbb{c}$, the norm $\|x\|^2$ is optimized by computing minimum norm $\|x_{\mathbb{c}}\|^2$ over each clique cyclically. Algorithm 4 minimizes $\|x_{\mathbb{c}}\|^2$ in a very similar way as MNP Algorithm [46] described in Background section. The only difference lies in handling the invalid extreme base at step 4 of Algorithm 4.

Algorithm 2 ComputeInvalidContribution

Input: Vector $x_{\mathbb{c}}$ the output of the max flow algorithm.
Output: The transformed vector $x_{\mathbb{c}}$ with minimum ℓ_2 norm .

for $\forall p \in \mathbb{c}$ **do**
  for $i = 2 : m$ **do**
    repeat
      find smallest k, $i \geq k \geq 1$, such that $x_{\mathbb{c}}(p^i) > x_{\mathbb{c}}(p^{i-1}) = x_{\mathbb{c}}(p^{i-2}) \cdots = x_{\mathbb{c}}(p^k)$ or $x_{\mathbb{c}}(p^i) = x_{\mathbb{c}}(p^{i-1}) = x_{\mathbb{c}}(p^{i-2}) \cdots = x_{\mathbb{c}}(p^{k+1}) > x_{\mathbb{c}}(p^k)$;
      let av_k be the average of $x_{\mathbb{c}}(p^i), x_{\mathbb{c}}(p^{i-1}), \ldots, x_{\mathbb{c}}(p^k)$;
      set $x_{\mathbb{c}}(p^i), x_{\mathbb{c}}(p^{i-1}), \ldots, x_{\mathbb{c}}(p^k)$ equal to av_k;
    until $x_{\mathbb{c}}(p^{k+1}) \leq x_{\mathbb{c}}(p^k)$
  end for
end for

Algorithm 3 HybridML: Algorithm for minimizing a sum of multilabel submodular functions

Input: $\{f_{\mathbb{c}}\}$ such that $f = \sum f_{\mathbb{c}}$.
Output: $x = \arg\min \|x\|^2$ subject to $x \in B(f)$.

# Initialize
for all $(\mathbb{c} \in \mathcal{C})$ **do**
  $q_{\mathbb{c}} \leftarrow$ Take any extreme base of $f_{\mathbb{c}}$;
  $S_{\mathbb{c}} := \{q_{\mathbb{c}}\}$;
  $y_{\mathbb{c}} := q_{\mathbb{c}}$;
end for
$x := \sum_{\mathbb{c}} y_{\mathbb{c}}$;
# Perform Block Coordinate Descent with blocks specified by Cliques
while ($\|x\|$ decreases by more than δ) **do**
  for all $(\mathbb{c} \in \mathcal{C})$ **do**
    MLHybridOverAClique($f_{\mathbb{c}}$,$S_{\mathbb{c}}$,$x_{\mathbb{c}}$,$y_{\mathbb{c}}$);
  end for
end while

Algorithm 4 MLHybridOverAClique

Input: Clique function: $f_{\mathbb{c}}$
Input: Set of valid extreme bases selected in last iteration: $S_{\mathbb{c}}$
Input: Restriction of current solution vector x on $\mathbb{c}$: $x_{\mathbb{c}}$
Input: Current clique vector: $y_{\mathbb{c}}$
Output: Clique vector $y^*_{\mathbb{c}} \in B(f_c)$ minimizing $\|x_{\mathbb{c}}\|^2$
Output: Updated set $S^*_{\mathbb{c}}$ of valid extreme bases

while (TRUE) **do**
  Find new translation $a_{\mathbb{c}} := x_{\mathbb{c}} - y_{\mathbb{c}}$;
  Find extreme base $\hat{q}_{\mathbb{c}} := \arg\min_{q_{\mathbb{c}} \in B_{f_{\mathbb{c}}}} \langle x_{\mathbb{c}}, q_{\mathbb{c}} \rangle$ using Edmond's algorithm.
  if Extreme base $\hat{q}_{\mathbb{c}}$ is invalid according to Definition 3 **then**
    ComputeInvalidContribution($x_{\mathbb{c}}$);
    continue;
  end if
  Find translated extreme base $\hat{p}_{\mathbb{c}} = \hat{q}_{\mathbb{c}} + a_{\mathbb{c}}$;
  if $(\|x_{\mathbb{c}}\|^2 \leq \langle x_{\mathbb{c}}, \hat{p} \rangle + \epsilon)$ **then**
    break;
  end if
  $S_{\mathbb{c}} := S_{\mathbb{c}} \cup \hat{q}_{\mathbb{c}}$;
  $P_{\mathbb{c}} = \{\hat{q_{\mathbb{c}}} + a_{\mathbb{c}} | q_{\mathbb{c}} \in S_{\mathbb{c}}\}$;
  Find $x_{\mathbb{c}}$ in affine hull of $P_{\mathbb{c}}$;
  If $x_{\mathbb{c}}$ is not in convex hull $P_{\mathbb{c}}$, translate to nearest point in convex hull and update $S_{\mathbb{c}}$;
end while