Block Stability for MAP Inference

Hunter Lang
MIT

David Sontag
MIT

Aravindan Vijayaraghavan
Northwestern University

Abstract

Recent work (Lang et al., 2018) has shown that some popular approximate MAP inference algorithms perform very well when the input instance is *stable*. The simplest stability condition assumes that the MAP solution does not change at all when some of the pairwise potentials are adversarially perturbed. Unfortunately, this strong condition does not seem to hold in practice. We introduce a significantly more relaxed condition that only requires portions of an input instance to be stable. Under this block stability condition, we prove that the pairwise LP relaxation is *persistent* on the stable blocks. We complement our theoretical results with an evaluation of real-world examples from computer vision, and we find that these instances have large stable regions.

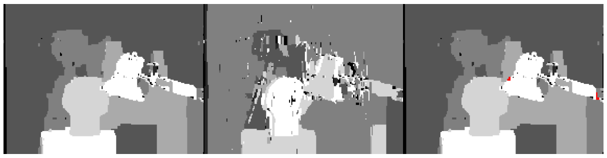
1 INTRODUCTION

As researchers and practitioners begin to apply machine learning algorithms to areas of society where human lives are at stake—such as bail decisions, autonomous vehicles, and healthcare—it becomes increasingly important to understand the empirical performance of these algorithms from a theoretical standpoint. Because many machine learning problems are NP-hard, the approaches deployed in practice are often heuristics or approximation algorithms. These sometimes come with performance guarantees, but the algorithms typically do *much better* in practice than their theoretical guarantees suggest. Heuristics are often chosen solely on the basis of their past empirical performance, and our theoretical understanding of the reasons for such performance is limited. To design better algorithms and to better understand the strengths of our existing approaches, we must bridge this gap between theory and practice.

To this end, many researchers have looked *beyond worst-case analysis*, developing approaches like smoothed analysis, average-case analysis, and stability. Broadly, these approaches all attempt to show that the worst-case behavior of an algorithm does not occur too often in the real world. Some methods are able to show that worst-case instances are "brittle," whereas others show that real-world instances have special structure that makes the problem significantly easier. In this work, we focus on stability, which takes the latter approach. Informally, an instance of an optimization problem is said to be stable if the (optimal) solution does not change when the instance is perturbed. This captures the intuition that solutions should "stand out" from other feasible points on real-world problem instances.

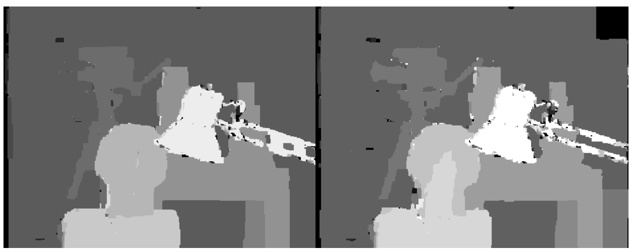
We focus on the MAP inference problem in Markov Random Fields. MAP inference is often used to solve *structured prediction* problems like stereo vision. The goal of stereo vision is to go from two images—one taken from slightly to the right of the other, like the images seen by your eyes—to an *assignment* of depths to pixels, which indicates how far each pixel is from the camera. Markov Random Fields give an elegant method for finding the best assignment of states (depths) to variables (pixels), taking into account the structure of the output space. Figure 1 illustrates the need for a better theoretical understanding of MAP inference algorithms. An exact solution to the MAP problem for a real-world stereo vision instance appears in Figure 1a. Figure 1b shows an assignment that, according to the current theory, might be returned by the best approximation algorithms. These two assignments agree on less than 1% of their labels. Finally, Figure 1c shows an assignment *actually* returned by an approximation algorithm—this assignment has over 99% of labels in common with the exact one. This surprising behavior is not limited to stereo vision. Many structured prediction problems have approximate MAP algorithms that perform extremely well in practice despite the exact MAP problems being

Proceedings of the 22[nd] International Conference on Artificial Intelligence and Statistics (AISTATS) 2019, Naha, Okinawa, Japan. PMLR: Volume 89. Copyright 2019 by the author(s).



(a) Exact soln. (b) Best theory (c) Actual appx. soln.

Figure 1: Example of an exact solution (left) to a stereo vision MAP problem compared to a 2-approximation (the best known theoretical performance bound, center), and a real approximate solution returned by the LP relaxation (right). Fractional portions of the LP solution are shown in red.



(a) Original solution (b) Solution to perturbation

Figure 2: Solutions to an original (left) and multiplicatively perturbed (right) stereo vision instance. The two solutions agree on over 95% of the vertices.

NP-hard (Koo et al., 2010; Savchynskyy et al., 2013; Kappes et al., 2015; Swoboda et al., 2014).

The huge difference between Figures 1b and 1c indicates that real-world instances must have some structure that makes the MAP problem easy. Indeed, these instances seem to have some stability to multiplicative perturbations. Figure 2 shows MAP solutions to a stereo vision instance and a small perturbation of that instance.[1] These solutions share many common labels, and many portions are exactly the same.

Put simply, in the remainder of this work we attempt to use the structure depicted in Figure 2 to explain why Figure 1c is so similar to Figure 1a.

The approximation algorithm used to produce Figure 1c is called the *pairwise LP relaxation* (Wainwright and Jordan, 2008; Chekuri et al., 2001). This algorithm formulates MAP inference as an integer linear program (ILP) with variables x that are constrained to be in $\{0,1\}$. It then relaxes that ILP to a linear program (LP) with constraints $x \in [0,1]$, which can be solved efficiently. Unfortunately, the LP solution may not be a valid MAP solution—it may have fractional values $x \in (0,1)$—so it might need to be *rounded* to a MAP solution. However, in practice, the LP solution frequently takes values in $\{0,1\}$, and these values "match" with the exact MAP solution, so very little rounding is needed. For example, the LP solution shown in Figure 1c takes binary values that agree with the exact solution on more than 99% of the instance. This property is known as *persistency*.

Much previous work has gone into understanding the persistency of the LP relaxation, typically stemming from a desire to give partial optimality guarantees for LP solutions (Kovtun, 2003; Savchynskyy et al., 2013; Shekhovtsov, 2013; Swoboda et al., 2016; Shekhovtsov et al., 2018; Haller et al., 2018). These results typically *use* the fact that the pairwise LP is often persistent on large portions of these instances to design fast algorithms for verifying partial optimality. Contrastingly, our work aims to understand *why* the LP is persistent so frequently on real-world instances.

Lang et al. (2018) first explored the stability framework of Makarychev et al. (2014) in the context of MAP inference. They showed that under a strong stability condition, the pairwise LP relaxation provably returns an exact MAP solution. Unfortunately, this condition (that the solution does not change *at all* under perturbations) is rarely, if ever, satisfied in practice. On the other hand, Figure 2 demonstrates that the original and perturbed solutions do have many labels in common, so there could be some stability present at the "sub-instance" level. In this work, we give an extended stability framework, generalizing the work of Lang et al. (2018) to the setting where only some parts of an instance have stable structure. This naturally connects to work on dual decomposition for MAP inference. We establish a theoretical connection between dual decomposition and stability, which allows us to use stability even when it is only present on parts of an instance, and allows us to combine stability with other reasons for persistency. In particular, we define a new notion called *block stability*, for which we show the following:

- We prove that approximate solutions returned by the pairwise LP relaxation agree with the exact solution on all the stable blocks of an instance.
- We design an algorithm to find stable blocks on real-world instances.
- We run this algorithm on several examples from low-level computer vision, including stereo vision, where we find that these instances contain large stable blocks.
- We demonstrate that the block framework can be used to incorporate stability with other structural reasons for persistency of the LP relaxation.

Taken together, these results suggest that block stabil-

[1] The instances in Figures 1 and 2 have the same input images, but Figure 2 uses higher resolution.

ity is a plausible explanation for the empirical success of LP-based algorithms for MAP inference.

2 BACKGROUND

2.1 MAP Inference and Metric Labeling

A Markov Random Field consists of a graph $G = (V, E)$, a discrete set of labels $L = \{1, \dots, k\}$, and *potential functions* θ that capture the cost of assignments $f : V \to L$. The MAP inference task in a Markov Random Field is to find the assignment (or *labeling*) $f : V \to L$ with the lowest cost:

$$\min_{f:V\to L} \sum_{u\in V} \theta_u(f(u)) + \sum_{uv\in E} \theta_{uv}(f(u), f(v)). \qquad (1)$$

Here we have decomposed the set of potential functions functions θ into θ_u and θ_{uv}, which correspond to nodes and edges in the graph G, respectively. A Markov Random Field that can be decomposed in this manner is known as a *pairwise* MRF; we focus exclusively on pairwise MRFs. In equation (1), the *single-node* potential functions $\theta_u(i)$ represent the cost of assigning label i to node u, and the *pairwise* potentials $\theta_{uv}(i, j)$ represent the cost of simultaneously assigning label i to node u and label j to node v.

The MAP inference problem has been extensively studied for special cases of the potential functions θ. When the pairwise potential functions θ_{uv} take the form

$$\theta_{uv}(i, j) = \begin{cases} 0 & i = j \\ w(u, v) & \text{otherwise,} \end{cases}$$

the model is called a *generalized Potts model*. When the *weights* $w(u, v)$ are nonnegative, as they are throughout this paper, the model is called *ferromagnetic* or *attractive*. This formulation has enjoyed a great deal of use in the computer vision community, where it has proven especially useful for modeling low-level problems like stereo vision, segmentation, and denoising (Boykov et al., 2001; Tappen and Freeman, 2003). With this special form of θ_{uv}, we can re-write the MAP inference objective as:

$$\min_{f:V\to L} Q(f) := \sum_{u\in V} \theta_u(f(u)) + \sum_{\substack{uv\in E\\ f(u)\neq f(v)}} w(u, v) \qquad (2)$$

Here we have defined Q as the objective of a feasible labeling f. We can then call (G, θ, w, L) an instance of MAP inference for a Potts model with node costs θ and weights w.

The minimization problem (2) is also known as Uniform Metric Labeling, and was first defined and studied under that name by Kleinberg and Tardos (2002). Exact minimization of the objective (2) is NP-hard (Kleinberg and Tardos, 2002), but many good approximation algorithms exist. Most notably for our work, Kleinberg and Tardos (2002) give a 2-approximation based on the pairwise LP relaxation (3).[2]

$$\begin{aligned}
\min_x \quad & \sum_{u\in V}\sum_{i\in L} \theta_u(i)x_u(i) + \sum_{uv\in E}\sum_{i,j} \theta_{uv}(i, j)x_{uv}(i, j) \\
\text{s.t.} \quad & \sum_i x_u(i) = 1, && \forall u \in V,\ \forall i \in L \\
& \textstyle\sum_j x_{uv}(i, j) = x_u(i) && \forall (u, v) \in E,\ \forall i \in L, \\
& \textstyle\sum_i x_{uv}(i, j) = x_v(j) && \forall (u, v) \in E,\ \forall j \in L, \\
& x_u(i) \geq 0, && \forall u \in V,\ i \in L. \\
& x_{uv}(i, j) \geq 0, && \forall (u, v) \in E,\ i,\ j \in L.
\end{aligned} \qquad (3)$$

Their algorithm rounds a solution x of (3) to a labeling f that is guaranteed to satisfy $Q(f) \leq 2Q(g)$. The $|V||L|$ decision variables $x_u(i)$ represent the (potentially fractional) assignment of label i at vertex u. While solutions x to (3) might, in general, take fractional values $x_u(i) \in (0, 1)$, solutions are often found to be *almost entirely* binary-valued in practice (Koo et al., 2010; Meshi et al., 2016; Swoboda et al., 2014; Savchynskyy et al., 2013; Kappes et al., 2015), and these values are typically the same ones taken by the exact solution to the original problem. Figure 1c demonstrates this phenomenon. In other words, it is often the case in practice that if $g(u) = i$, then $x_u(i) = 1$, where g and x are solutions to (2) and (3), respectively. This property is called *persistency* (Adams et al., 1998). We say a solution x is persistent at u if $g(u) = i$ and $x_u(i) = 1$ for some i.

This LP approach to MAP inference has proven popular in practice because it is frequently persistent on a large percentage of the vertices in an instance, and because researchers have developed several fast algorithms for solving (3). These algorithms typically work by solving the *dual*; Tree-reweighted Message Passing (TRW-S) (Kolmogorov, 2006), MPLP (Globerson and Jaakkola, 2008), and subgradient descent (Sontag et al., 2012) are three well-known dual approaches. Additionally, the introduction of fast general-purpose LP solvers like Gurobi (Gurobi Optimization, 2018) has made it possible to directly solve the primal (3) for medium-sized instances.

2.2 Stability

An instance of an optimization problem is stable if its solution doesn't change when the input is perturbed.

[2] Kleinberg and Tardos (2002) use the so-called "metric LP," but this is equivalent to (3) for Potts potentials (Archer et al., 2004; Lang et al., 2018), and their rounding algorithm also works for this formulation.

To discuss stability formally, one must specify the exact type of perturbations considered. As in Lang et al. (2018), we study multiplicative perturbations of the weights:

Definition 1 ((β, γ)-perturbation, Lang et al. (2018))**.** Given a weight function $w : E \to \mathbb{R}_{\geq 0}$, a weight function w' is called a (β, γ)-perturbation w' of w if for any $(u, v) \in E$,

$$\frac{1}{\beta} w(u, v) \leq w'(u, v) \leq \gamma w(u, v).$$

With the perturbations defined, we can formally specify the notion of stability:

Definition 2 ((β, γ)-stable, Lang et al. (2018))**.** A MAP inference instance (G, θ, w, L) with graph G, node costs θ, weights w, labels L, and integer solution g is called (β, γ)-stable if for any (β, γ)-perturbation w' of w, and any labeling $h \neq g$, $Q'(h) > Q'(g)$, where Q' is the objective with costs c and weights w'.

That is, g is the unique solution to the optimization (2) where w is replaced by any valid (β, γ)-perturbation of w. As β and γ increase, the stability condition becomes increasingly strict. One can show that the LP relaxation (3) is tight (returns an exact solution to (2)) on suitably stable instances:

Theorem 1 (Theorem 1, Lang et al. (2018))**.** *Let x be a solution to the LP relaxation* (3) *on a (2,1)-stable instance with integer solution g. Then $x = g$.*

Many researchers have used stability to understand the real-world performance of approximation algorithms. Bilu and Linial (2010) introduced perturbation stability for the Max Cut problem. Makarychev et al. (2014) improved their result for Max Cut and gave a general framework for applying stability to graph partitioning problems. Lang et al. (2018) extended their results to MAP inference in Potts models. Stability has also been applied to clustering problems in machine learning (Balcan et al., 2009, 2015; Balcan and Liang, 2016; Awasthi et al., 2012; Dutta et al., 2017).

3 BLOCK STABILITY

The current stability definition used in results for the LP relaxation (Definition 2) requires that the MAP solution does not change at all for any $(2, 1)$-perturbation of the weights w. This strong condition is rarely satisfied by practical instances such as those in Figure 1 and Figure 2. However, it may be the case that the instance is $(2, 1)$-stable when restricted to *large blocks* of the vertices. We show in Section 5 that this is indeed the case in practice, but for now we precisely define what it means to be *block stable*, where some parts of the instance may be stable, but others may not. We demonstrate how to connect the ideas of dual decomposition and stability, working up to our main theoretical result in Theorem 2. Appendix A.1 contains proofs of the statements in this section.

We begin our discussion with an informal version of our main theorem:

Informal Theorem (see Theorem 2)**.** *Assume an instance (G, θ, w, L) has a block S that is $(2, 1)$-stable* and *has some additional, additive stability with respect to the node costs θ for nodes* along the boundary *of S. Then the LP* (3) *is persistent on S.*

To reason about different blocks of an instance (and eventually prove persistency of the LP on them), we need a way to *decompose* the instance into subproblems so that we can examine each one more or less independently. The *dual decomposition* framework (Sontag et al., 2012; Komodakis et al., 2011) provides a formal method for doing so. The commonly studied Lagrangian dual of (3), which we call the *pairwise dual*, turns every node into its own subproblem:

$$\begin{aligned} \max_\eta P(\eta) = \max_\eta \sum_{u \in V} \min_i (\theta_u(i) + \sum_v \eta_{uv}(i)) \\ + \sum_{uv \in E} \min_{i,j} (\theta_{uv}(i, j) - \eta_{uv}(i) - \eta_{vu}(j)) \end{aligned} \tag{4}$$

This can be derived by introducing Lagrange multipliers η on the two consistency constraints for each edge $(u, v) \in E$ and each $i \in L$:

$$\sum_i x_{uv}(i, j) = x_v(j) \;\forall j$$
$$\sum_j x_{uv}(i, j) = x_u(i) \;\forall i$$

Many efficient solvers for (4) have been developed, such as MPLP (Globerson and Jaakkola, 2008). But the subproblems in (4) are too small for our purposes. We want to find *large* portions of an instance with stable structure. Given a set $S \subset V$, define $E_S = \{(u, v) \in E : u \in S,\ v \in S\}$ to be the set of edges with both endpoints in S, and let $T = V \setminus S$. We may consider relaxing fewer consistency constraints than (4) does, to form a *block dual* with blocks S and T.

$$\begin{aligned} \max_\delta \sum_{W \in \{S,T\}} \min_{x^W} \Bigg(\sum_{u \in W} \sum_{i \in L} (\theta_u(i) + \sum_{v:(u,v) \in E_\partial} \delta_{uv}(i)) x_u^W(i) \\ + \sum_{uv \in E_W} \sum_{i,j} \theta_{uv}(i, j) x_{uv}^W(i, j) \Bigg) \\ + \sum_{uv \in E_\partial} \min_{i,j} (\theta_{uv}(i, j) - \delta_{uv}(i) - \delta_{vu}(j)) \end{aligned} \tag{5}$$

subject to the following constraints for $W \in \{S, T\}$:

$$\begin{aligned}
&\sum_i x_u^W(i) = 1, && \forall u \in W,\ \forall i \in L\\
&\textstyle\sum_j x_{uv}^W(i,j) = x_u^W(i) && \forall (u,v) \in E_W,\ \forall i \in L,\\
&\textstyle\sum_i x_{uv}^W(i,j) = x_v^W(j) && \forall (u,v) \in E_W,\ \forall j \in L.\\
&x_u^W(i) \geq 0, && \forall u \in W\ \forall i \in L.\\
&x_{uv}^W(i,j) \geq 0, && \forall (u,v) \in E_W,\ \forall i,\ j \in L.
\end{aligned} \tag{6}$$

Here the consistency constraints of (3) are only relaxed for *boundary edges* that go between S and T, denoted by E_∂. Each subproblem (each minimization over x^W) is an LP of the same form as (3), but is defined only on the block W (either S or T, in this case). If $S = V$, the block dual is equivalent to the primal LP (3). We denote the constraint set (6) by $\mathcal{L}^W$. In these subproblems, the node costs $\theta_u(i)$ are modified by $\sum_{v:(u,v)\in E_\partial} \delta_{uv}(i)$, the sum of the block dual variables coming from the other block. We can thus rewrite each subproblem as an LP of the form:

$$\min_{x^W \in \mathcal{L}^W} \sum_{u\in W}\sum_{i\in L} \theta_u^\delta(i) x_u^W(i) + \sum_{uv\in E_W}\sum_{ij} \theta_{uv}(i,j) x_{uv}^W(i,j),$$

where

$$\theta_u^\delta(i) = \theta_u(i) + \sum_v \delta_{uv}(i). \tag{7}$$

By definition, θ^δ is equal to θ on the interior of each block. It only differs from θ on the *boundaries* of the blocks. We show in Appendix A.1 how to turn a solution η^* of (4) into a solution δ^* of (5); this block dual is efficiently solvable. The form of θ^δ suggests the following definition for a *restricted instance*:

Definition 3 (Restricted Instance)**.** Consider an instance (G, θ, w, L) of MAP inference, and let $S \subset V$. The instance *restricted to* S with modification δ is given by:

$$((S, E_S), \theta^\delta|_S, w|_{E_S}, L),$$

where θ^δ is as in (7) and is restricted to S, and the weights w are restricted to E_S.

Given a set S, let δ^* be a solution to the block dual (5). We essentially prove that if the instance restricted to S, with modification δ^*, is $(2,1)$-stable, the LP solution x to the original LP (3) (defined on the full, unmodified instance) takes binary values on S:

Lemma 1. *Consider the instance (G, θ, w, L). Let $S \subset V$ be any subset of vertices, and let δ^* be any solution to the block dual* (5)*. Let x be the solution to* (3) *on this instance. If the restricted instance*

$$((S, E_S), \theta^{\delta^*}|_S, w|_{E_S}, L)$$

is $(2,1)$*-stable with solution g_S, then $x|_S = g_S$.*

Here g_S is the exact solution to the restricted instance $((S, E_S), \theta^{\delta^*}|_S, w|_{E_S}, L)$ with node costs modified by δ^*. This may or may not be equal to $g|_S$, the overall exact solution restricted to the set S. If $g_S = g|_S$, Lemma 1 implies that the LP solution x is *persistent* on S:

Corollary 1. *For an instance (G, θ, w, L), let g and x be solutions to* (2) *and* (3)*, respectively. Let $S \subset V$ and δ^* a solution to the block dual for S. Assume the restricted instance $((S, E_S), \theta^{\delta^*}|_S, w|_{E_S}, L)$ is* $(2,1)$*-stable with solution $g|_S$. Then $x|_S = g|_S$; x is persistent on S.*

Appendix A.1 contains a proof of Lemma 1.

Finally, we can reinterpret this result from the lens of stability by defining additive perturbations of the node costs θ. Let $\bar{S}$ be the boundary of set S; i.e. the set of $s \in S$ such that s has a neighbor that is not in S.

Definition 4 (ϵ-bounded cost perturbation)**.** Given a subset $S \subset V$, node costs $\theta : V \times L \to \mathbb{R}$, and a function

$$\epsilon : \bar{S} \times L \to \mathbb{R},$$

a cost function $\theta' : V \times L \to \mathbb{R}$ is an ϵ-bounded perturbation of θ (with respect to S) if the following equation holds for some ψ with $|\psi_u(i)| \leq |\epsilon_u(i)|$ for all $(u, i) \in V \times L$:

$$\theta'_u(i) = \begin{cases} \theta_u(i) + \psi_u(i) & u \in \bar{S} \\ \theta_u(i) & \text{otherwise.} \end{cases}$$

In other words, a perturbation θ' is allowed to differ from θ by at most $|\epsilon_u(i)|$ for u in the boundary of S, and must be equal to θ everywhere else.

Definition 5 (Stable with cost perturbations)**.** A restricted instance $((S, E_S), \theta|_S, w|_{E_S}, L)$ with solution g_S is called $(\beta, \gamma, \epsilon)$-stable if for all ϵ-bounded cost perturbations θ' of θ, the instance $((S, E_S), \theta'|_S, w|_{E_S}, L)$ is (β, γ)-stable. That is, g_S is the unique solution to all the instances $((S, E_S), \theta'|_S, w'|_{E_S}, L)$ with θ' an ϵ-bounded perturbation of θ and w' a (β, γ)-perturbation of w.

Theorem 2. *Consider an instance (G, θ, w, L) with subset S, let g and x be solutions to* (2) *and* (3) *on this instance, respectively, and let δ^* be a solution to the block dual* (5) *with blocks $(S, V \setminus S)$. Define $\epsilon_u^*(i) = \sum_{v:(u,v)\in E_\partial} \delta_{uv}^*(i)$. If the restricted instance*

$$((S, E_S), \theta|_S, w|_{E_S}, L)$$

is $(2, 1, \epsilon^*)$*-stable with solution $g|_S$, then the LP x is persistent on S.*

Proof. This follows immediately from Definition 5, the definition of ϵ^*, and Corollary 1. $\square$

Definition 5 and Theorem 2 provide the connection between the dual decomposition framework and stability: by requiring stability to additive perturbations of the node costs along the boundary of a block S, where the size of the perturbation is determined by the block dual variables, we can effectively isolate S from the rest of the instance and apply stability to the modified subproblem.

In Appendix A.4, we show how to use the dual decomposition techniques from this section to combine stability with other structural reasons for persistency of the LP *on the same instance*.

4 FINDING STABLE BLOCKS

In this section, we present an algorithm for finding stable blocks in an instance. We begin with a procedure for testing (β, γ)-stability as defined in Definition 2. Lang et al. (2018) prove that it is sufficient to look for labelings that violate stability in the *adversarial perturbation*

$$w^*(u,v) = \begin{cases} \gamma w(u,v) & g(u) \neq g(v) \\ \frac{1}{\beta} w(u,v) & g(u) = g(v), \end{cases}$$

which tries to make the exact solution g as bad as possible. With that in mind, we can try to find a labeling f such that $f \neq g$, subject to the constraint that $Q^*(f) \leq Q^*(g)$ (here Q^* is the objective with costs θ and weights w^*). The instance is (β, γ)-stable if and only if no such f exists. We can write such a procedure as the following optimization problem:

$$\begin{aligned} \max_x \quad & \tfrac{1}{2n} \sum_{u \in V} \sum_{i \in L} |x_u(i) - x_u^g(i)| \\ \text{s.t.} \quad & \sum_i x_u(i) = 1 && \forall u \in V,\ \forall i \in L, \\ & \textstyle\sum_i x_{uv}(i,j) = x_v(j), && \forall (u,v) \in E,\ \forall j \in L \\ & \textstyle\sum_j x_{uv}(i,j) = x_u(i), && \forall (u,v) \in E,\ \forall i \in L \\ & x_u(i) \in \{0,1\} && \forall u \in V,\ i \in L \\ & x_{uv}(i,j) \in \{0,1\} && \forall (u,v) \in E,\ \forall i,j \in L \\ & Q^*(x) \leq Q^*(g) \end{aligned} \tag{8}$$

The first five sets of constraints ensure that x forms a feasible integer labeling f. The objective function captures the normalized Hamming distance between this labeling f and the solution g; it is linear in the decision variables x_u because g is fixed—$x_u^g(i) = 1$ if $g(u) = i$ and 0 otherwise. Of course, the "objective constraint" $Q^*(x) \leq Q^*(g)$ is also linear in x. We have only linear and integrality constraints on x, so we can solve (8) with a generic ILP solver such as Gurobi (Gurobi Optimization, 2018). This procedure is summarized in Algorithm 1. Put simply, the algorithm tries to find the labeling f that is most different from g (in Hamming distance) subject to the constraint that $Q^*(f) \leq Q^*(g)$. By construction, the instance is stable if and only if the optimal objective value of this ILP is 0. If there is a positive objective value, there is some f with $f \neq g$ but $Q^*(f) \leq Q^*(g)$; this violates stability. The program is always feasible because g satisfies all the constraints. Because it solves an ILP, `CheckStable` is not a polynomial time algorithm, but we were still able to use it on real-world instances of moderate size in Section 5.

Algorithm 1: `CheckStable`(g, β, γ)

Given g, compute the adversarial (β, γ)-perturbation w^* for g.
Construct ILP $\mathcal{I}$ according to (8) using g and w^*.
Set $x, d =$ `GenericILPSolver`$(\mathcal{I})$
if $d > 0$ **then**
  return x // *instance is not stable*
else
  return `None` // *instances is stable*
end

We now describe our heuristic algorithm for finding regions of an input instance that are $(2,1)$-stable after their boundary costs are perturbed. Corollary 1 implies that we do *not* need to test for $(2,1)$-stability for *all* ϵ^*-bounded perturbations of node costs—we can simply check with respect to the one given by (7) with $\delta = \delta^*$. That is, we need only check for $(2,1)$-stability in the instance with node costs θ^{δ^*}. This is enough to guarantee persistency.

In each iteration, the algorithm begins with a partition (henceforth "decomposition" or "block decomposition") of the nodes V into disjoint sets $(S_1, \ldots, S_B)$. It then finds a block dual solution for each S_b (see Appendix B.1 for details) and computes the restricted instances using the optimal block dual variables to modify the node costs. Next, it uses Algorithm 1 to check whether these modified instances are $(2,1)$-stable. Based on the results of `CheckStable`, we either update the previous decomposition or verify that a block is stable, then repeat.

All that remains are the procedures for initializing the algorithm and updating the decomposition in each iteration given the results of `CheckStable`. The initial decomposition consists of $|L| + 1$ blocks, with

$$S_b = \{u | g(u) = b \text{ and } \forall (u,v) \in E,\ g(v) = b\}. \tag{9}$$

So $|L|$ blocks consist of the interiors of the label sets of g—a vertex u belongs to S_b if u and all its neighbors have $g(\cdot) = b$. The boundary vertices—$u \in V$ such that there is some $(u,v) \in E$ with $g(u) \neq g(v)$—are

Algorithm 2: `BlockStable`(g, β, γ)

Given g, create blocks $(S_1^1, \dots, S_k^1, S_*^1)$ with (9).
Initialize $K^1 = |L|$.
for $t \in \{1, \dots, M\}$ **do**
  Initialize $S_*^{t+1} = \emptyset$.
  for $b \in \{1, \dots, K^t, *\}$ **do**
    Find block dual solution δ^* for $(S_b^t, V \setminus S_b^t)$.
    Form $\mathcal{I} = ((S_b^t, E_{S_b}), \theta'|_{S_b^t}, w|_{E_{S_b}}, L)$ using δ^* and (7).
    Set $f_b =$ `CheckStable`$(g|_{S_b^t}, \beta, \gamma)$ run on instance $\mathcal{I}$.
    Compute $V_\Delta = \{u \in S_b^t | f_b(u) \neq g(u)\}$.
    Set $S_b^{t+1} = S_b^t \setminus V_\Delta$
    Set $S_*^{t+1} = S_*^{t+1} \cup V_\Delta$.
    if $b = *$ **then**
      Set $R = S_*^t \setminus V_\Delta$.
      Let $(S_{K^t+1}^{t+1}, \dots, S_{K^t+p+1}^{t+1}) =$ `BFS`(R) be the p connected components in R that get the same label from g.
      Set $K^{t+1} = K^t + p$.
    end
  end
end

added to a special *boundary block* denoted by S_*. Some blocks may be empty if g is not surjective.

In an iteration of the algorithm, for every block, `CheckStable` returns a labeling f_b that satisfies $Q_{S_b}^{\theta', w^*}(f_b) \leq Q_{S_b}^{\theta', w^*}(g|_{S_b})$ and might also have $f_b \neq g|_{S_b}$. If $f_b = g|_{S_b}$, the block is stable and we do nothing. Otherwise, we *remove* the vertices $V_\Delta = \{u \in S_b : f_b(u) \neq g|_{S_b}(u)\}$ and add them to the boundary block S_*.

Finally, we try to *reclaim* vertices from the old boundary block. Like all the other blocks, the boundary block gets tested for stability in each iteration. Some of the vertices in this block may have $f_b(u) = g|_{S_b}(u)$. We call this the *remainder set* R. We run breadth-first-search in R to identify connected components of vertices that get the same label from g. Each of these components becomes its own new block, and is added to the block decomposition for the next step. This heuristic prevents the boundary block from growing too large and significantly improves our experimental results, since the boundary block is rarely stable. The entire procedure is summarized in Algorithm 2.

5 NUMERICAL EXAMPLES

We focus in this section on instances where the pairwise LP performs very well. The examples studied here are more extensively examined in Kappes et al. (2015), where they also compare the effectiveness of the LP to other MAP inference algorithms. Most importantly, though, they observe that the pairwise LP takes fractional values only at a very small percentage of the nodes on these instances. This makes them good candidates for a stability analysis.

5.1 Object Segmentation

For the object segmentation problem, the goal is to partition the pixels of the input image into a handful of different *objects* based on the semantic content of the image. The first two rows of figure 3 show some example object segmentation instances. We study a version of the segmentation problem where the number of desired objects is known. We use the model of Alahari et al. (2010); full details about the MRFs used in this experiment can be found in Appendix B. Each instance has 68,160 nodes and either five or eight labels, and we ran Algorithm 2 for $M = 50$ iterations to find $(2, 1)$-stable blocks. The LP (3) is persistent on 100% of the nodes for all three instances we study.

Row 3 of Figure 3 shows the output of Algorithm 2 on each segmentation instance. The red vertices are regions where the algorithm was unable to find a large stable block. The green pixels represent a boundary between blocks, demonstrating the block structure. The largest blocks seem to correspond to objects in the original image (and regions in the MAP solution).

One interesting aspect of these instances is the large number of stable blocks S with $|S| = 1$ for the Road instance (Column 2). If the LP is persistent at a node u, there is a trivial decomposition in which u belongs to its own stable block (see Appendix A.3 for discussion on block size). However, the existence of stable blocks with size $|S| > 1$ is not implied by persistency, so the presence of such blocks means the instances have special structure. The red regions in Figure 3, Row 3 could be replaced by stable blocks of size one. However, Algorithm 2 did not *find* the trivial decomposition for those regions, as it did for the center of the Road instance. We believe the large number of blocks with $|S| = 1$ for the Road instance could therefore be due to our "reclaiming" strategy in Algorithm 2, which does not try to merge together reclaimed blocks, rather than a lack of stability in that region.

5.2 Stereo Vision

As we discussed in Section 1, the stereo vision problem takes as input two images L and R of the same scene, where R is taken from slightly to the right of L. The goal is to output a depth label for each pixel in L that represents how far that pixel is from the camera. Depth is inversely proportional to the disparity (how

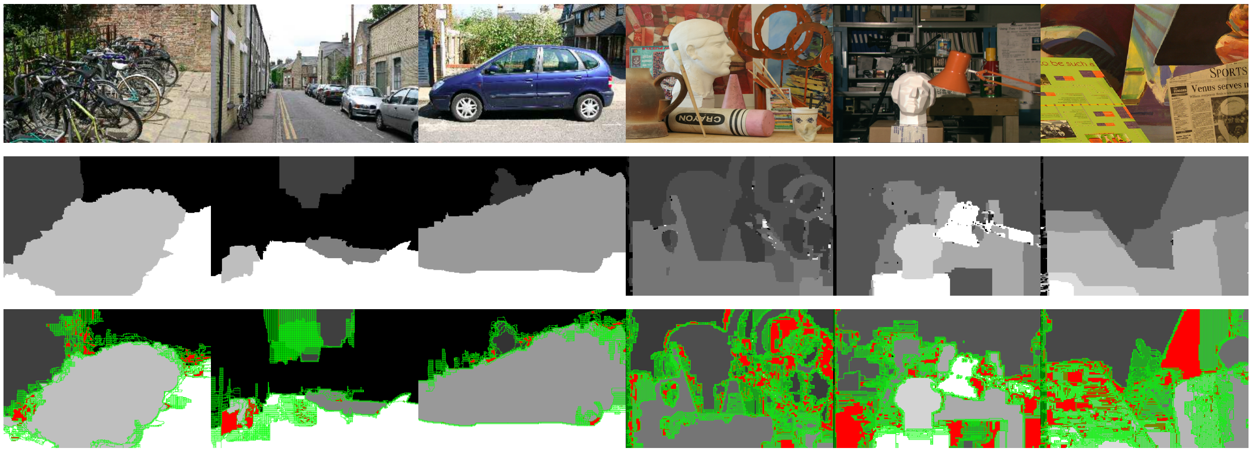


Figure 3: Columns 1-3: object segmentation instances; Bikes, Road, Car. Columns 4-6: stereo instances; Art, Tsukuba, Venus. Row 1: original image for the instance. Row 2: MAP solution for the model. Row 3: results of Algorithm 2. Regions where the algorithm failed to find a nontrivial stable decomposition are shown in red. Boundaries between blocks are shown in green.

much the pixel moves) of the pixel between the images L and R. So the goal is to estimate the (discretized) disparity of each pixel. The first two rows of Figure 3 show three example instances and their MAP solutions. We use the MRF formulation of Boykov et al. (2001) and Tappen and Freeman (2003). The exact details of these stereo MRFs can be found in Appendix B. These instances have between 23,472 and 27,684 nodes, and between 8 and 16 labels. The LP (3) is persistent on 98-99% of each instance.

Row 3 of Figure 3 shows the results of Algorithm 2 for the stereo instances. As with object segmentation, we observe that the largest stable blocks tend to coincide with the actual objects in the original image. Compared to segmentation, fewer vertices in these instances seem to belong to large stable blocks. We believe that decreased resolution may play a role in this difference. The computational challenge of scaling Algorithms 1 and 2 to the stereo model forced us to use downsampled (0.5x or smaller) images to form the stereo MRFs. Brief experiments with higher resolution suggest that improving the scalability of Algorithm 2 is an interesting avenue for improving these results.

The results in Figure 3 demonstrate that large stable regions exist in practical instances. Theorem 2 guarantees that solutions to (3) are persistent on these blocks, so stability provides a novel explanation for the persistency of the LP relaxation in practice.

6 DISCUSSION

The block stability framework we presented helps to understand the tightness and persistency of the pairwise LP relaxation for MAP inference. Our experiments demonstrate that large blocks of common computer vision instances are stable. While our experimental results are for the Potts model, our extension from (β, γ)-stability to block stability uses no special properties of the Potts model and is completely general. If a (β, γ)-stability result similar to Theorem 1 is given for other pairwise potentials, the techniques used here immediately give the analogous version of Theorem 2. Our results thus give a connection between dual decomposition and stability.

The method used to prove the results in Section 3 can even extend beyond stability. We only need stability to apply Theorem 1 to a modified block. Instead of stability, we could plug in any result that guarantees the pairwise LP on that block has a *unique* integer solution. Appendix A gives an example of incorporating stability with tree structure on the same instance. Combining different structures to fully explain persistency on *real-world* instances will require new algorithmic insight.

The stability of these instances suggests that designing new inference algorithms that directly take advantage of stable structure is an exciting direction for future research. The models examined in our experiments use mostly hand-set potentials. In settings where the potentials are learned from training data, is it possible to encourage stability of the learned models?

Acknowledgments

The authors thank Fredrik D. Johansson for many helpful discussions. This work was supported by NSF AitF awards CCF-1637585 and CCF-1723344. AV is also supported by NSF Grant No. CCF-1652491.

References

Warren P Adams, Julie Bowers Lassiter, and Hanif D Sherali. Persistency in 0-1 polynomial programming. *Mathematics of operations research*, 23(2):359–389, 1998.

Karteek Alahari, Pushmeet Kohli, and Philip HS Torr. Dynamic hybrid algorithms for map inference in discrete mrfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(10):1846–1857, 2010.

Aaron Archer, Jittat Fakcharoenphol, Chris Harrelson, Robert Krauthgamer, Kunal Talwar, and Éva Tardos. Approximate classification via earthmover metrics. In *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '04, pages 1079–1087, Philadelphia, PA, USA, 2004. Society for Industrial and Applied Mathematics. ISBN 0-89871-558-X.

Pranjal Awasthi, Avrim Blum, and Or Sheffet. Center-based clustering under perturbation stability. *Information Processing Letters*, 112(1):49–54, 2012.

Maria-Florina Balcan and Yingyu Liang. Clustering under perturbation resilience. 2016.

Maria-Florina Balcan, Avrim Blum, and Anupam Gupta. Approximate clustering without the approximation. In *Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '09, pages 1068–1077, 2009.

Maria-Florina Balcan, Nika Haghtalab, and Colin White. Symmetric and asymmetric k-center clustering under stability. *arXiv preprint arXiv:1505.03924*, 2015.

Yonatan Bilu and Nathan Linial. Are stable instances easy? In *Innovations in Computer Science*, pages 332–341, 2010.

Stan Birchfield and Carlo Tomasi. A pixel dissimilarity measure that is insensitive to image sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(4):401–406, 1998.

Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, Nov 2001. ISSN 0162-8828.

Chandra Chekuri, Sanjeev Khanna, Joseph (Seffi) Naor, and Leonid Zosin. Approximation algorithms for the metric labeling problem via a new linear programming formulation. In *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '01, pages 109–118, Philadelphia, PA, USA, 2001. Society for Industrial and Applied Mathematics. ISBN 0-89871-490-7.

Abhratanu Dutta, Aravindan Vijayaraghavan, and Alex Wang. Clustering stable instances of euclidean k-means. In *Advances in Neural Information Processing Systems (to appear)*, 2017.

Amir Globerson and Tommi S Jaakkola. Fixing max-product: Convergent message passing algorithms for map lp-relaxations. In *Advances in neural information processing systems*, pages 553–560, 2008.

LLC Gurobi Optimization. Gurobi optimizer reference manual, 2018. URL http://www.gurobi.com.

Stefan Haller, Paul Swoboda, and Bogdan Savchynskyy. Exact map-inference by confining combinatorial search with lp relaxation. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Joerg Kappes, Bjoern Andres, Fred Hamprecht, Christoph Schnorr, Sebastian Nowozin, Dhruv Batra, Sungwoong Kim, Bernhard Kausler, Jan Lellmann, Nikos Komodakis, et al. A comparative study of modern inference techniques for discrete energy minimization problems. In *International Journal of Computer Vision*, 2015.

Jon Kleinberg and Éva Tardos. Approximation algorithms for classification problems with pairwise relationships: Metric labeling and markov random fields. *J. ACM*, 49(5):616–639, September 2002. ISSN 0004-5411.

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

Nikos Komodakis, Nikos Paragios, and Georgios Tziritas. Mrf energy minimization and beyond via dual decomposition. *IEEE transactions on pattern analysis and machine intelligence*, 33(3):531–552, 2011.

Terry Koo, Alexander M. Rush, Michael Collins, Tommi Jaakkola, and David Sontag. Dual decomposition for parsing with non-projective head automata. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1288–1298, 2010.

Ivan Kovtun. Partial optimal labeling search for a np-hard subclass of (max,+) problems. In *Joint Pattern Recognition Symposium*, pages 402–409. Springer, 2003.

Hunter Lang, David Sontag, and Aravindan Vijayaraghavan. Optimality of approximate inference algorithms on stable instances. In *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*. PMLR, 2018.

Konstantin Makarychev, Yury Makarychev, and Aravindan Vijayaraghavan. Bilu-linial stable instances of max cut and minimum multiway cut. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 890–906. Society for Industrial and Applied Mathematics, 2014.

Ofer Meshi, Mehrdad Mahdavi, Adrian Weller, and David Sontag. Train and test tightness of lp relaxations in structured prediction. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1776–1785, New York, New York, USA, 20–22 Jun 2016. PMLR.

Bogdan Savchynskyy, Jörg Hendrik Kappes, Paul Swoboda, and Christoph Schnörr. Global map-optimality by shrinking the combinatorial search area with convex relaxation. In *Advances in Neural Information Processing Systems*, pages 1950–1958, 2013.

Alexander Shekhovtsov. Exact and partial energy minimization in computer vision. 2013.

Alexander Shekhovtsov, Paul Swoboda, and Bogdan Savchynskyy. Maximum persistency via iterative relaxed inference in graphical models. *IEEE transactions on pattern analysis and machine intelligence*, 40(7):1668–1682, 2018.

Jamie Shotton, John Winn, Carsten Rother, and Antonio Criminisi. Textonboost: Joint appearance, shape and context modeling for multi-class object recognition and segmentation. In *European conference on computer vision*, pages 1–15. Springer, 2006.

David Sontag, Amir Globerson, and Tommi Jaakkola. Introduction to dual decomposition for inference. In Suvrit Sra, Sebastian Nowozin, and Stephen J. Wright, editors, *Optimization for Machine Learning*, pages 219–254. MIT Press, 2012.

Paul Swoboda, Bogdan Savchynskyy, Jorg H Kappes, and Christoph Schnorr. Partial optimality by pruning for map-inference with general graphical models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1170–1177, 2014.

Paul Swoboda, Alexander Shekhovtsov, Jörg Hendrik Kappes, Christoph Schnörr, and Bogdan Savchynskyy. Partial optimality by pruning for map-inference with general graphical models. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 38(7), 2016.

Marshall F Tappen and William T Freeman. Comparison of graph cuts with belief propagation for stereo, using identical mrf parameters. In *null*, page 900. IEEE, 2003.

Martin J Wainwright and Michael I Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1–2):1–305, 2008.