Efficient Low Rank Convex Bounds for Pairwise Discrete Graphical Models

Valentin Durante [1] **George Katsirelos** [2] **Thomas Schiex** [1]

Abstract

In this paper, we extend a Burer-Monteiro style method to compute low rank Semi-Definite Programming (SDP) bounds for the MAP problem on discrete graphical models with an arbitrary number of states and arbitrary pairwise potentials. We consider both a penalized constraint approach and a dedicated Block Coordinate Descent (BCD) approach which avoids large penalty coefficients in the cost matrix. We show our algorithm is decreasing. Experiments show that the BCD approach compares favorably to the penalized approach and to usual linear bounds relying on convergent message passing approaches.

1. Introduction

Graphical models (GMs) (Koller & Friedman, 2009) are descriptions of decomposable multivariate functions. A variety of frameworks in Computer Science, Logic, Constraint Satisfaction/Programming, Machine Learning, Statistical Physics, Operations Research and Artificial Intelligence can be considered as specific sub-classes of graphical models. In this paper we consider discrete pairwise graphical models. The *maximum a posteriori* (MAP) assignment problem on probabilistic GMs consists in finding an assignment of all variables that maximizes the posterior probability. It is also known as the weighted constraint satisfaction problem (WCSP) for deterministic GMs (Cooper et al., 2020; Schiex et al., 1995). This problem has applications in image processing, computational biology (Simoncini et al., 2015), resource management (Bensana et al., 1999), configuration, etc.

A GM $M = \langle X, \Phi \rangle$ is specified by a set of n variables $X = \{x_1, \ldots, x_n\}$, where x_i can take d_i possible states and a set Φ of potential functions θ_S involving variables in $S \subset X$, the scope of θ_S. We note $d = \sum_{i=1}^{n} d_i$ the total number of states. A graphical model defines a joint function Θ_M over X as $\Theta_M = \sum_{\theta_S \in \Phi} \theta_S$. When all scopes have maximum cardinality 2, the model is pairwise and the graph with vertices in X and edges defined by scopes is the graph of the pairwise GM.

The MAP problem is decision NP-complete (Shimony, 1994). There exist some well studied polynomial classes, including structural restrictions, like bounded tree-width, and restrictions on the type of potential functions θ_S. The latter has been fully characterized by showing that, depending on the potential functions, the problem is either polynomial or NP-hard (Thapper & Živnỳ, 2016) and includes, for example, models defined by submodular functions (Cooper et al., 2020). Outside of polynomial cases, a vast array of polynomial time heuristics have been developed. One strand of research exploits the tight connection of GMs to Linear Programming (LP). Convergent belief propagation algorithms such as TRW-S (Kolmogorov, 2005) for MAP or the Virtual Arc Consistency (VAC) (Cooper et al., 2008; 2010) algorithm for WCSP give approximate LP-dual bounds through reparametrizations. Combined with primal heuristics, they offer post-hoc guarantees through an optimality gap, making such algorithms useful in time-constrained scenarios.

In many cases, these bounds correspond to high quality solutions and a small optimality gap. On the hardest instances however, the obtained bounds become increasingly loose. One natural direction for improvement is to climb up in the Sherali & Adams (1990) hierarchy by reasoning on more than two variables simultaneously (Yedidia et al., 2000; Sontag et al., 2012; Nguyen et al., 2017). The associated time and space costs quickly become extreme however, as moving up one step in the hierarchy increases time and space complexity by a factor equal to the average domain size. Another approach is to use SDP bounds, which provide superior guarantees (Goemans & Williamson, 1995) compared to approximate LP bounds.

However, SDP has found limited application in MAP/WCSP. The main reason lies in the fact that Goemans and Williamson's SDP relaxation of MAXCUT, a specialization of MAP with binary variables, requires $\theta(n^2)$ variables and memory resulting in an $O(n^6)$ complexity for usual interior point methods. The algorithm doesn't scale well

[1]Université Fédérale de Toulouse, ANITI, INRAE, UR 875, 31326 Toulouse, France [2]MIA-Paris-Mathématiques et Informatique Appliquées, INRAE, 75231 Paris, France. Correspondence to: Valentin Durante <valentin.durante@inrae.fr>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

and becomes impractical except on small very hard random problems. For this reason, approximate LP methods such as those we describe above are preferred because they achieve a better trade off between tightness and computational cost.

The suitability of SDP for MAP and other applications has improved with the non-convex low-rank Burer-Monteiro approach (Burer & Monteiro, 2003). Exploiting the result of Barvinok (1995) and Pataki (1998), who showed that the rank of solutions of SDP problems is bounded by $O(\sqrt{n})$, this approach uses a low-rank factorisation $\mathbf{X} = \mathbf{V}\mathbf{V}^T$ of the semi-definite matrix $\mathbf{X}$ in the SDP relaxation of these combinatorial problems. It has since been observed that, in practice, the rank of solutions is often lower, with some software using a constant $O(1)$ rank. Therefore, the number of variables and the memory requirements can be reduced to $O(nr)$ where r is the rank used in the decomposition. As a side benefit, the decomposition guarantees the semi-definiteness of the matrix $\mathbf{X}$, at the cost of addressing a non-convex optimization problem. Combined with row-by-row updates (Javanmard et al., 2016; Wang et al., 2017), Riemannian gradient descent (Javanmard et al., 2016; Mei et al., 2017) or Riemannian trust-region methods (Absil et al., 2007), the Burer-Monteiro approach provides empirically very efficient methods to solve SDP relaxations of various combinatorial optimization problems with binary variables such as MAXCUT, MAXSAT (Wang & Kolter, 2019) or MAP over Ising. The row-by-row update methods, with their efficient $O(nr)$ updates are especially attractive.

Multi-state Potts model MRFs, where pairwise functions are weighted identity matrices, can still be directly dealt by using simple row-by-row updates (Pabbaraju et al., 2020). In contrast, arbitrary pairwise GMs, with multi-state variables and arbitrary potential functions need to explicitly deal with an additional constraint for each original variable of the considered GM. In the usual one-hot encoding of finite domain variables, a GM variable x_k with d_k states is encoded as d_k binary variables, each representing whether a state is used or not. The new constraints specify that each variable is assigned exactly one state. In the associated SDP relaxation, one may naturally dualize such constraints with a suitable penalty (Lasserre, 2016). The ability to represent arbitrary pairwise potentials on an arbitrary number of states is not only useful for better mathematical modeling in, e.g., image segmentation and labelling (Krähenbühl & Koltun, 2011), but also for improving deep learning approaches trying to "learn how to reason". These architectures rely on efficient differentiable convex optimization layers, as approximations of discrete reasoning as in MAXSAT (Wang et al., 2019).

The main contributions of the paper are:

- the formal definition and characterization of two variants of Burer-Monteiro algorithms solving an SDP relaxation of the discrete MAP problem with arbitrary pairwise potentials.
- empirical results showing that the Block Coordinate Descent-based variant can outperform usual LP bounds (exact or message-passing based) in terms of tightness, and also outperforms exact LP and other SDP penalized variant in terms of efficiency on dense random problems as well as on a real computational biology problem.

2. SDP relaxation for pairwise GMs

Throughout the paper, matrices are denoted with a capital boldface font and all vectors are column vectors. For matrix $\mathbf{A}$, A_{ij} represents the entry at the i-th row and j-th column of $\mathbf{A}$. A_i represents its i-th row as a column vector. For a vector g, $||g||$ denotes its Euclidean norm. The vector defined by the diagonal of $\mathbf{A}$ is $\text{diag}(\mathbf{A})$, the Frobenius inner product of matrices $\mathbf{A}, \mathbf{B}$ is denoted as $\langle \mathbf{A}, \mathbf{B} \rangle$.

A pairwise GM $M = \langle X, \Phi \rangle$ with associated graph (X, E) defines the joint function $\Theta_M = \sum_{(x_i,x_j)\in E} \theta_{ij} + \sum_{x_i \in X} \theta_i + \theta_\varnothing$. We assume that the potential functions $\theta_S \in \Phi$ are described as tensors, which in our case means matrices, vectors and scalar, respectively. For optimization, the constant term $\theta_\varnothing$ can be ignored. Before building an SDP relaxation, the problem is reduced to a quadratic form by representing every finite domain variable $x_k \in X$ as a vector b_k of d_k Booleans. The j^{th} element of b_k will take value 1 when variable $x_k = j$ and value 0 otherwise. The value of $\theta_{ij}(x_i, x_j)$ is $b_i^T \mathbf{\Theta}_{ij} b_j$ and the value of $\theta_i(x_i)$ is then $b_i^T \theta_i$. If we denote by b the stacked vector $b^T = [b_1^T, \ldots, b_n^T]$, we then have $\Theta_M(x) = b^T \mathbf{\Theta}^{\mathbb{B}} b + b^T \theta^{\mathbb{B}}$ where $\theta^{\mathbb{B}}$ is the stacked vector of all θ_i and $2 \times \mathbf{\Theta}^{\mathbb{B}}$ is a symmetric block matrix having block $\mathbf{\Theta}_{ij}$ when $(x_i, x_j) \in E$ and the $d_i \times d_j$ zero matrix otherwise. Note that $\mathbf{\Theta}^{\mathbb{B}}$ is sparse by block, according to the connectivity of the graph of M. More importantly, $\mathbf{\Theta}^{\mathbb{B}}$ has zero blocks on its diagonal since no pairwise function in Φ connects a variable x_k to itself. This property will later be useful for BCD updates.

When optimizing over b instead of x, one must enforce the fact that only one element of every Boolean vector b_k is set to 1, *i.e.*, that $\forall x_k \in X, \sum_{j=1}^{d_k} (b_k)_j = 1$. This set of constraints can be gathered in a linear constraint $\mathbf{A}b = 1_n$ where $\mathbf{A}$ is a Boolean matrix in $\mathbb{B}^{n \times d}$ with $A_i^T = [0_{d_1}^T \ \cdots \ 0_{d_{i-1}}^T \ 1_{d_i}^T \ 0_{d_{i+1}}^T \ \cdots \ 0_{d_n}]$ and where 1_n is an n-vector of 1. Finally, minimizing Θ_M becomes equivalent to solving the following constrained quadratic program:

$$\min_{b \in \mathbb{B}^d} \ b^T \mathbf{\Theta}^{\mathbb{B}} b + b^T \boldsymbol{\theta}^{\mathbb{B}} \quad \text{s.t} \quad \mathbf{A}b = 1_n \tag{1}$$

To build an SDP relaxation of the quadratic program above, centered $\{-1, 1\}$ variables are used and relaxed to norm-1

vectors of dimension n. The problem above can be reduced to such formulation by using the $\{-1,1\}$ variables $c = 2b - 1_d$. After this variable change, denoting $\mathbf{\Theta} = \mathbf{\Theta}^{\mathbb{B}}/4$, $\theta = (\theta^{\mathbb{B}} + \mathbf{\Theta}^{\mathbb{B}} 1_d)/2$, $\mathbf{F} = \mathbf{A}/2$ and $u = 1_n - \mathbf{A}1_d/2$, the problem becomes:

$$\min_{c \in \{-1,1\}^d} \quad c^T \mathbf{\Theta} c + c^T \theta \quad \text{s.t} \quad \mathbf{F}c = u \tag{2}$$

Here, the initial exactly-one constraints $\sum b_i = 1$ become $2F_i c = 2 - d_i$, with $F_i \in \mathbb{R}^d$, $2F_i^T = [0_{d_1}^T \ \cdots \ 0_{d_{i-1}}^T \ 1_{d_i}^T \ 0_{d_{i+1}}^T \ \cdots \ 0_{d_n}]$. This change of variables creates a constant term in the objective value which can be ignored for optimization purposes. In this formulation, the matrix $\mathbf{\Theta}$ has the same exact sparsity as $\mathbf{\Theta}^{\mathbb{B}}$.

2.1. Lassere dualization of linear constraints

In order to get an unconstrained quadratic program, we must remove the linear constraints in (2). A usual approach to do this is to dualize the linear constraint with a penalty ρ that must be appropriately chosen (Lasserre, 2016). This reduces the problem to a pure MAXCUT problem on which the Burer-Monteiro approach can be directly applied and solved using efficient row-by-row updates (Wang et al., 2017). The relaxed problem becomes:

$$\min_{c \in \{-1,1\}^d} \quad c^T \mathbf{\Theta} c + c^T \theta + (2\rho + 1)||\mathbf{F}c - u||^2 \tag{3}$$

As soon as $\rho \geqslant \max\{|c^T \mathbf{\Theta} c + c^T \theta| \ : \ c \in \{-1,1\}^d\}$, the solution of (3) and (1) are the same (Lasserre, 2016). One can note at this point that if any potential function in the GM at hand contains potentials of large amplitude, ρ will need to take a large value to ensure the above property.

We next homogenize the problem by converting linear terms to quadratic terms. We introduce the extended vector $e^T = [c^T \ 1] \in \{-1,1\}^{d+1}$ and reformulate (3) in terms of e. Then, (3) equivalently asks to minimize $e^T \mathbf{Q} e$ where $e \in \{-1,1\}^{d+1}$ and $\mathbf{Q}$ is the symmetric matrix:

$$\begin{pmatrix} (2\rho+1)\mathbf{F}^T\mathbf{F} + \mathbf{\Theta} & \frac{1}{2}(\theta^T - 2(2\rho+1)u^T\mathbf{F})^T \\ \frac{1}{2}(\theta^T - 2(2\rho+1)u^T\mathbf{F}) & (2\rho+1)u^T u \end{pmatrix}$$

The usual SDP relaxation of the MAX-CUT problem can then be written using the new rank 1 matrix variable $\mathbf{X} = ee^T \in \mathbb{R}^{(d+1)\times(d+1)}$. Dropping the rank-1 constraint, the SDP relaxation is:

$$\min_{\mathbf{X}} \{\langle \mathbf{Q}, \mathbf{X} \rangle : \mathbf{X} \succeq 0; \ X_{ii} = 1, \ i = 1, \ldots, d+1\} \tag{4}$$

Burer & Monteiro (2003) introduced the idea of using a low-rank factorization of $\mathbf{X}$ to solve the SDP (4). This factorization was motivated by a proof by Barvinok (1995) and Pataki (1998) of the following theorem:

Theorem 2.1. *There exists an optimum $\mathbf{X}^*$ of (4) with rank r such that $\frac{r(r+1)}{2} \leqslant m$. With m the number of constraints.*

Every positive semi-definite matrix of rank r can be factorized as a product of two rank r matrices: $\mathbf{X} = \mathbf{V}\mathbf{V}^T$, $\mathbf{V} \in \mathbb{R}^{(d+1)\times r}$. Then, (4) becomes:

$$\min_{\mathbf{V} \in \mathbb{R}^{(d+1)\times r}} \ \langle \mathbf{Q}, \mathbf{V}\mathbf{V}^T \rangle \ \text{ s.c } ||V_i|| = 1 \ , \ i = 1, \ldots, d+1 \tag{5}$$

With $V_i \in \mathbb{R}^r$ the row vector i of $\mathbf{V}$, this reduces the number of variables from $(d+1)^2$ to $r(d+1)$ and the semi-definite constraint $\mathbf{X} \succeq 0$ now becomes implicit as $\mathbf{V}\mathbf{V}^T$ is always positive semi-definite. Several approaches exploit this idea. We use the mixing method (Wang et al., 2017), a row-by-row update method which uses efficient $O(rd)$ updates. It consists in cyclic updates of the row vectors V_i, all other row vectors being fixed:

$$V_i = -\frac{g_i}{||g_i||}, \qquad g_i = \sum_{j=1}^{d+1} Q_{i,j} V_j \quad \forall i = 1, \ldots, d+1$$

The mixing method is known to recover the optimum of the convex SDP (4) as long as $r > \sqrt{2(d+1)}$. One issue with using the mixing method in the presence of dualized constraints generated by Lassere's approach is that the row-by-row updates will be sensitive to the magnitude of the penalty ρ. As we observed previously, ρ may be large if the input GM has large terms. The large value of ρ will create large coefficients in $\mathbf{Q}$ and then the value of g_i will be dominated by few terms, possibly slowing down convergence.

A feasible solution of (1) always satisfies the so-called *gangster* constraint (Zhao et al., 1998). This constraint specifies that the off-diagonal entries of the matrix $b_k b_k^T$ must be all zeros. If we consider the Boolean vector b_k that corresponds to the variable $x_k \in X$, then $b_k^i b_k^j = 0 \ \forall \ i \neq j$, since at least one of the two terms is equal to zero. It is important to notice that if the solution to (4) nullifies the quadratic penalty term induced by the Lasserre dualization, it will implicitly satisfy this gangster constraint.

Proposition 2.2. *If $tr(\mathbf{P}\mathbf{V}\mathbf{V}^T) = 0$, where $\mathbf{P} = \begin{pmatrix} \mathbf{F}^T\mathbf{F} & -\mathbf{F}^T u \\ -u^T\mathbf{F} & u^T u \end{pmatrix}$, then $\mathbf{V}$ satisfies the gangster constraint*

The full proof is provided in Appendix A.6.

2.1.1. SDP BOUND USING BLOCK-COORDINATE DESCENT

We now consider an approach that exploits the specific structure of the matrix $\mathbf{Q}$ in the constrained quadratic formulation (2). As previously, using the extended vector e, the rank-1

matrix variable $\mathbf{X} = ee^T$, dropping the rank-1 constraint and using a low rank factorization $\mathbf{X} = \mathbf{V}\mathbf{V}^T$ as above, the quadratic optimization problem (2) can be relaxed into the SDP:

$$\min_{\mathbf{V}} \langle \mathbf{R}, \mathbf{V}\mathbf{V}^T \rangle \text{ s.t } \begin{cases} \langle \mathbf{U}_i, \mathbf{V}\mathbf{V}^T \rangle = 2 - d_i, i = 1, \dots, n \\ \text{diag}(\mathbf{V}\mathbf{V}^T) = 1_{d+1} \end{cases} \tag{6}$$

with $\mathbf{R} = \begin{pmatrix} \mathbf{\Theta} & \frac{1}{2}\theta \\ \frac{1}{2}\theta^T & 0 \end{pmatrix}$ and $\mathbf{U}_i = \begin{pmatrix} 0 & F_i \\ F_i^T & 0 \end{pmatrix}$. The diagonal constraint can be rewritten $||V_i|| = 1, \forall i$. We note $s_1 = 1, s_i = s_{i-1} + d_{i-1}, i = 2, \dots, n$. For a given GM variable x_k, s_k represents the position of the first row representing x_k in b, c and e. The exactly-one constraint $\langle \mathbf{U}_i, \mathbf{V}\mathbf{V}^T \rangle = 2 - d_i$ can be rewritten $V_{d+1}^T (\sum_{j=s_i}^{s_i+d_i-1} V_j) = 2 - d_i$.

Instead of doing row-by-row updates, we exploit the property that the $\mathbf{\Theta}$ matrix has zero blocks on the diagonal. For a given GM variable x_k, we simultaneously optimize all the rows of $\mathbf{V}$ that correspond to x_k while keeping all other rows fixed. To simultaneously update all these rows, noting $g_i = \sum_{j=1}^{d+1} R_{i,j} V_j$ as before, we have to solve:

$$\min_{V_i, s_k \leqslant i \leqslant s_{k+1}} \sum_{i=s_k}^{s_k+d_k-1} V_i^T g_i \text{ s.t. } \begin{cases} V_{d+1}^T (\sum_{j=s_k}^{s_k+d_k-1} V_j) = 2 - d_k \\ ||V_i|| = 1, \ s_k \leq i < s_{k+1} \end{cases} \tag{7}$$

Notice that since the corresponding $d_k \times d_k$ diagonal block matrix of $\mathbf{R}$ is all zero, the g_i do not depend on the optimized V_i and can be considered as fixed. In the following, we make the assumption that the vectors g_i and V_{d+1} are never colinear. The approaches suggested by Wang et al. (2017) also apply here.

By the second constraint, every solution row vector V_i must lie in the unit spherical manifold.

Proposition 2.3. *At the optimum of (7), every optimized vector V_i lies in the dimension 2 subspace generated by g_i and V_{d+1}.*

The full proof is provided in Appendix A.1. Given that V_i lies on the unit sphere and in this dimension 2 subspace, it is therefore entirely defined by the angle it makes with V_{d+1}. With vectors on the sphere, the problem above can be rewritten using trigonometric functions, omitting the d_i norm-1 constraints which are implicitly satisfied. To deal with the remaining exactly-one constraint, we write the Lagrangian dual using multiplier λ.

Theorem 2.4. *If we denote as ϕ_i the angle from g_i to V_{d+1}, the dual Lagrangian of problem (7) is:*

$$h(\lambda) = -\sum_{i=s_k}^{s_k+d_k-1} \sqrt{||g_i||^2 + 2\lambda||g_i|| \cos(\phi_i) + \lambda^2} + (d_k - 2)\lambda \tag{8}$$

The proof in Appendix A.2 is obtained using angle variables in the Lagrangian, differentiating, solving for equality to zero and injecting the solution back in the Lagrangian.

We could not exhibit a closed form of the optimum λ^*. However, the function $f = -h$ can be optimized using the Newton algorithm:

Proposition 2.5. *The second derivative of f is Lipschitz on $\mathbb{R}$ and is strictly positive whenever the g_i's and V_{d+1} are not colinear.*

The full proof is available in Appendix A.3. To reach quadratic convergence with the Newton method, we need to find a starting point which is close enough to λ^*. This starting point can be produced by solving the Lagrangian dual of a relaxation of the BCD problem (7):

$$\min_v v^T g \quad \text{s.t} \quad v'^T v = 2 - d_k, ||v||^2 = d_k \tag{9}$$

where $g^T = [g_1^T \ \dots \ g_{d_1}^T]$, $v'^T = [V_{d+1}^T \ \dots \ V_{d+1}^T]$ and $v^T = [V_1^T \ \dots \ V_{d_1}^T]$.

Thankfully, a solution of this relaxation can be identified analytically.

Theorem 2.6. *Let us define $p = 4 - \frac{4}{d_k}$ and $\gamma = ||g||^2 - \frac{1}{d_k}(v'^T g)^2$. The solution to (9) is given by $v^* = \alpha v' + \beta g$ with*

$$\alpha = -\tfrac{1}{d_k}(\beta v'^T g + d_k - 2) \quad \beta = -\sqrt{\tfrac{p}{\gamma}}$$

The full proof is provided in Appendix A.4.

The block-coordinate descent algorithm that we propose cycles over blocks and optimizes each, using the Newton method. It uses the solution identified in Theorem 2.6 to start the Newton iteration. We have not proved that this algorithm converges to the optimum solution of (6), but it does converge, as we show next.

Proposition 2.7. *The block-coordinate descent with fixed last column V_{d+1} is decreasing.*

Proof sketch. We can show that the difference in the objective value between successive iterations is positive, using the Cauchy-Schwartz inequality and trigonometric identities. The full proof is provided in Appendix A.5 □

This last theorem ensures that the block-coordinate descent will converge, since (6) is bounded. The convergence speed

using random row-by-row updates is linear in the neighborhood of a local optimum when the rank is sufficiently large (Wang et al., 2017; Erdogdu et al., 2018).

Computational Complexity. The update rule for the Newton method is $\lambda^{j+1} = \lambda^j - [\nabla^2 f(\lambda^j)]^{-1} \nabla f(\lambda^j)$ and the first and second derivative of f are:

$$f'(\lambda) = \sum_{i=s_k}^{s_k+d_k-1} \frac{g_i^T V_{d+1}^T + \lambda}{||g_i + \lambda V_{d+1}^T||} - (d_k - 2) \tag{10}$$

$$f''(\lambda) = \sum_{i=s_k}^{s_k+d_k-1} \frac{||g_i||^2 - (g_i^T V_{d+1}^T)^2}{||g_i + \lambda V_{d+1}^T||^3} \tag{11}$$

Given that all the vectors we are dealing with have size r, by pre-computing the dot products with g_i, computing the first derivative requires $O(d_k r)$ operations. For the second derivative, we can pre-compute the numerators $||g_i||^2 - (g_i^T V_{d+1}^T)^2$ and the evaluation of the second derivative is again $O(d_k r)$. Overall, one update step of the Newton method will require only $O(d_k r)$ operations, which is also what a single round of row-by-row updates over the d_k rows associated with x_k would require. The BCD updates however benefit from the efficient Newton updates. In practice, we observe that only few iterations of the Newton method are needed in our experiments.

2.2. Producing an integer primal solution

To produce a primal integer solution from the optimal solution of the convex relaxation, we use random rounding. Let V_r be a normalized vector sampled from a Gaussian distribution. We assign each variable to the value $x_k = \left(\max_{s_k \leqslant i \leqslant s_{k+1}} V_i^T V_r\right) - s_k$. The integer solution obtained is then submitted to a simple greedy search where, for every variable, the best improving change of state (if any) is applied. This is done on every variable repeatedly until a local minimum is reached. The same process is used for every convex relaxation considered in the paper.

3. Experiments

3.1. Description of solvers

We implemented the row-by-row updates with the dualized exactly-one constraints (LR-LAS) as well as the BCD update method (LR-BCD) in C++ (code available at github.com/ValDurante/LR-BCD) with the Eigen3 (Guennebaud et al., 2010) sparse matrix library (MPL2 licence). We use rank $r = \lceil \sqrt{2(d+1)} \rceil$ by default for LR-LAS and $r = \lceil \sqrt{2(n+d+1)} \rceil$ for LR-BCD but also tried variants with fixed lower ranks. The penalty coefficient ρ is set as the sum of the maximum of all the binary and unary functions. Thus, it enforces the theoretical assumption of the Lasserre dualization result.

They are compared with exact LP (local polytope) bounds and message passing MAP/MRF algorithms Min-Sum and TRW-S. The LP bounds are computed using CPLEX 20.1.0.0. We used Open-GM2 (Kappes et al., 2015), an efficient C++ MIT-licensed library, in release 3.3.7, available at `https://github.com/opengm` for Min-Sum (maximum number of iterations of 100, minimal message distance of 0.01 and damping of 0.8) and TRW-S (maximum number of iterations $100,000$, tolerance of 10^{-5}, TABLE mode). Min-Sum provides only upper bounds.

Overall, we therefore have five solvers: LR-LAS and LR-BCD (parameterized by their rank), LP, Min-Sum and TRW-S. We also considered using the ECOS interior point method available in CVXPY (Agrawal et al., 2018), but preliminary tests showed that it ran several orders of magnitude slower than the Mixing Method on small problems with 120 binary variables from the BiqMac library at `http://biqmac.uni-klu.ac.at/biqmaclib.html`.

All experiments were run on a single thread on a server equipped with a Xeon®Gold 6248R CPU@3.00GHz and 1TB of RAM running Debian Linux 4.19.98-1.

3.2. Experiments on random instances

Following (Park et al., 2019), we first used random pairwise MAP problems. The pairwise potentials are sampled uniformly from $[0, s]$ with $s = 5$. The unary potentials are sampled uniformly from $[0, 1]$ using fixed precision numbers with 9 digits precision. The magnitude of s controls the importance of pairwise couplings which are the source of NP-hardness. Empirically, we found that solution time and quality was insensitive to the value of s except for very low s (where coupling effects become negligible). We generated random problems varying the number of variables (100-1000) and number of states (3-10). All the random problems have a complete graph. For each point the results are averaged over 10 instances. We observed that the standard deviations of the measures were very low on the 10 generated samples and we therefore do not report it.

3.2.1. CPU-TIME.

We present in Figure 1 a summary of the cpu-time performance of all solvers on problems of increasing size (100 to 1,000 MRF variables) and increasing number of states (and therefore increasing size d). TRW-S is clearly the most efficient algorithm here, often one order of magnitude faster than LR-BCD. LR-LAS, instead, is considerably slower than LR-BCD, often by more than two orders of magnitude. On instances of 300 MRF variables and more than 3 states ($d > 900$), it times-out. The CPLEX-based exact local polytope solver is even slower. It times-out on 100 MRF variables instances with 7 states.

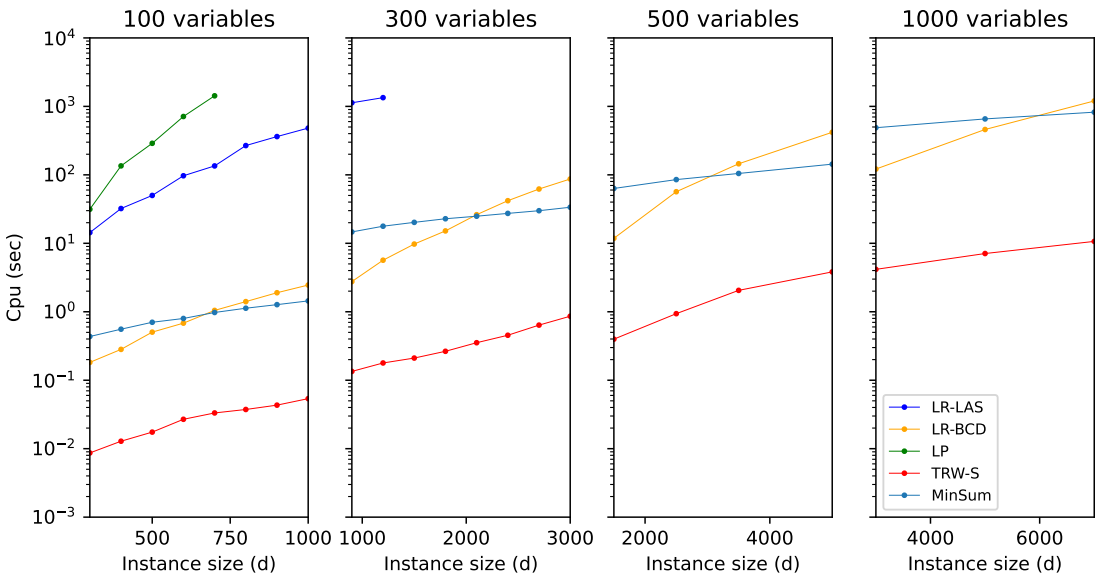

Figure 1. Cpu-time for all solvers on problems of increasing size (100, 300, 500 and 1000 variables) as a function of the problem size (number of states $\times$ number of variables).

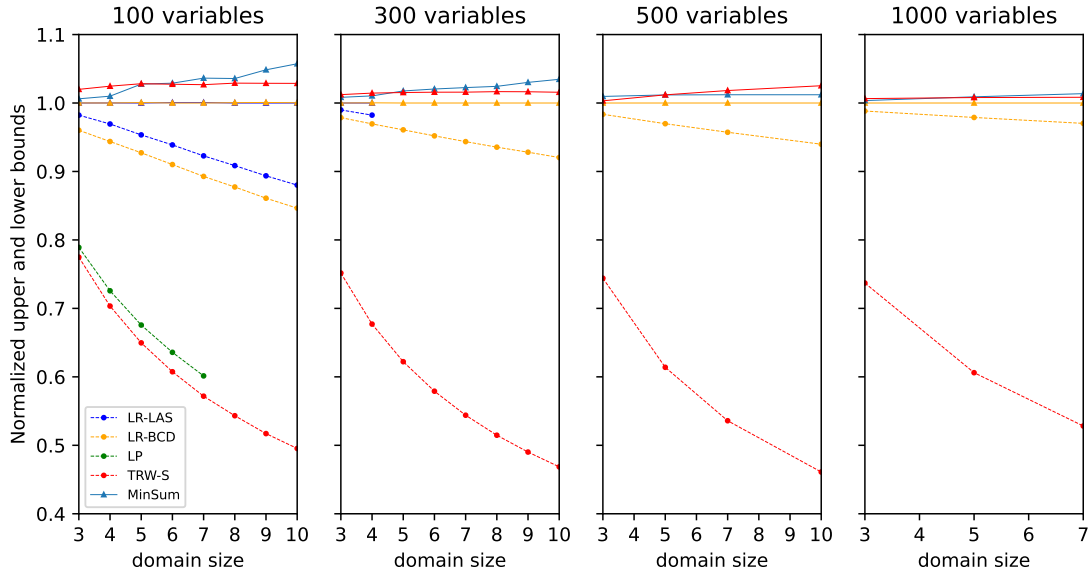

Figure 2. Normalized upper and lower bounds for all solvers on problems of increasing size (100, 300, 500 and 1000 variables) as a function of the number of states (domain size). The best integer primal solution value is taken as reference.

3.2.2. BOUNDS.

Figure 2 shows normalized upper and lower bounds on the same families of instances. Considering upper bounds, message passing algorithms show slightly inferior performance compared to other solvers. The most important differences appear in the lower bounds, where LP only slightly improves over TRW-S and where LR-LAS and LR-BCD really stand out. Thanks to its implicit enforcing of the gangster constraints, LR-LAS offers slightly imporved lower bounbds, but this extra tightness comes at an extreme computational cost. It is interesting to notice that for LR-BCD, the optimality gap decreases as the ratio of (number of variables)/(number of states) increases. This means that for a fixed number of states, increasing the number of variables will make the LR-BCD bounds more precise.

3.2.3. VERY LOW RANK RELAXATIONS.

One interesting feature of Burer-Monteiro style methods is the ability to control their efficiency through the rank r. As far as integer solutions are concerned, it is perfectly fine to use low or very low ranks, below the theoretical limits of $\lceil\sqrt{2(d+1)}\rceil$ and $\lceil\sqrt{2(n+d+1)}\rceil$: rounding will still provide a feasible integer solution. To compare the ability of TRW-S, LR-LAS and LR-BCD to quickly provide an integer solution, we ran all three algorithms with increasing time-outs and measured the cost of the integer solution produced as a function of time. Because LR-LAS times-out on all but the smallest instances, we tested this on a 100 variables, 5 states instance (we observed very consistent behavior among all samples from a given family of instances). This is illustrated in Figure 3. TRW-S is extremely fast and immediately produces its best possible integer solution. Surprisingly, LR-LAS and LR-BCD are able to produce a better integer solution, even from the first iteration. This integer solution also improves in quality as the number of iterations grows.

The effect of strong rank reduction is different between LR-LAS and LR-BCD. For best performance, LR-BCD is best used with a large rank: it reaches its best solution after less than 25 iterations (0.2 seconds). Instead LR-LAS may benefit from an intermediate rank, where it produces a comparable solution well before convergence. With the highest rank, the quality of the integer solutions produced by LR-LAS improves very slowly. This could be explained by the fact that the primal relaxed solutions that LR-LAS computes do not exactly satisfy the exactly-one constraints of each MRF variable until the very end.

3.2.4. EMPIRICAL CONVERGENCE RESULTS.

In this section, we empirically study the convergence of LR-BCD and the effect of the rank r on the value of the optimized criteria with increasing cpu-time and number of iterations. For an instance with 500 variables and three states, we computed a tight solution of (6) using Mosek (ApS, 2022). Figure 4 shows the relative error of the solutions produced by LR-BCD compared to the Mosek solution as a function of cpu-time. We observe that, even when the rank is high, LR-BCD converges very quickly to solutions with a very low tolerance, a tolerance of 10^{-3} being reached in less than 0.5 seconds. As the rank decreases, the convergence to a certain precision accelerates without any loss in precision, until a very low rank is reached where the quality of the solution cannot be pushed to tight tolerance. On the right, we also plot the relative error against the number of BCD iterations. Each iteration corresponds to an entire update of the low rank matrix $\mathbf{V}$. As the rank reduces, the number of iterations grows to the point where faster iterations are almost compensated by the increased number of iterations. In comparison, the industrial solver Mosek, with default settings, took 111 seconds to solve this relatively small instance.

3.3. Experiments on a real instance

We applied the same algorithms on a genomic assembly related problem which reduces naturally to a pairwise MAP/MRF problem with $8,574$ variables having 4 states ($34,296$ Boolean variables) and $80,763$ pairwise factors. This instance has been made available in the Cost Function Library, in the `real/fish` category. On this instance,TRW-S provides a lower bound of 0 and an integer solution with cost $127,878$ in 0.069 seconds. The LP bound is also 0 with a running time of 80.04 seconds. Min-Sum provides an integer solution of cost $479,924$ in 0.270 seconds (and seems stuck there, with no improvement with more iterations). LR-BCD produces a bound of $102,900$ and an integer solution of cost $122,694$, reducing the optimality gap from 100% (TRW-S) to 16.1% (LR-BCD). LR-BCD took 68.6 minutes on this instance while LR-LAS did not finish after several hours.

4. Conclusion

An increasing number of algorithms for solving convex relaxations of discrete optimization problems rely on the Burer-Monteiro approach (Burer & Monteiro, 2003). These approaches are fast and offer a lower bound on the discrete instance as well as an upper bound through rounding. This makes them highly desirable when efficiency is required. They can also offer tighter gaps than the more usual LP-related convergent message passing, especially on the hardest dense problems. However, to the best of our knowledge, all existing Burer-Monteiro algorithms deal only with the simple case where only diagonal-one constraints exist. This is enough for MAXCUT, MAX2SAT, and MAP/Ising and even MAP/Potts (where potential functions are weighted

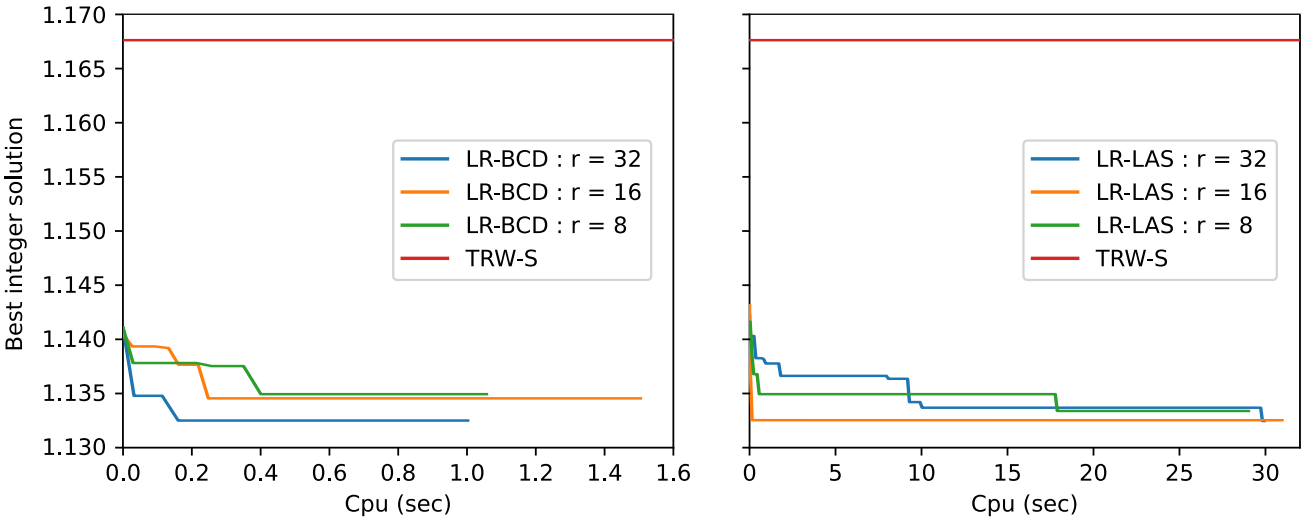

Figure 3. Comparison of the best integer solution as a function of time for TRW-S and LR-BCD (left) and TRW-S and LR-LAS (right) on a 100 variables instance with five states. Note the different time scales.

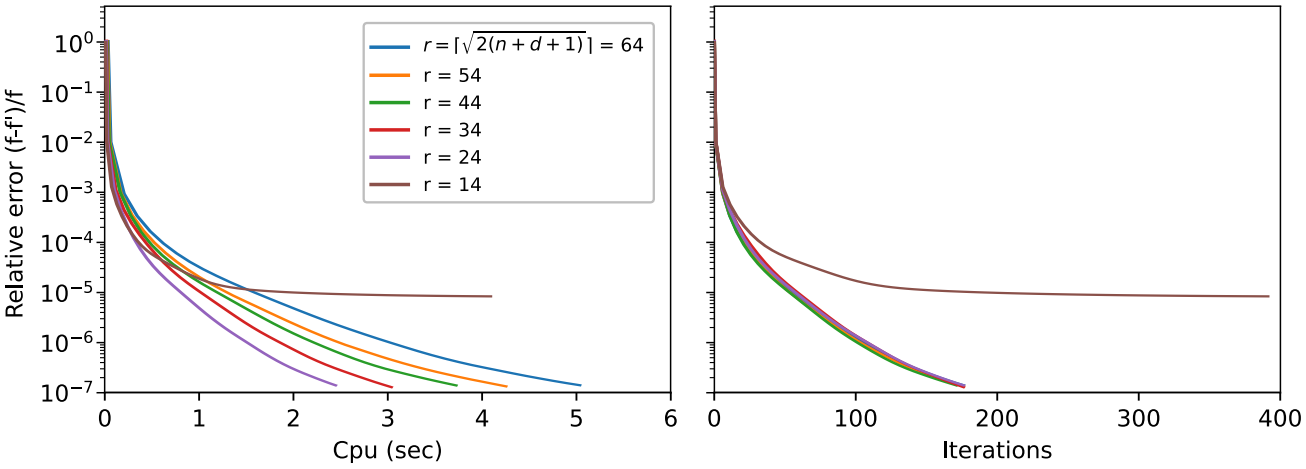

Figure 4. Relative error with a high quality solution vs. cpu-time (left) and number of iterations (whole matrix update, right) using different ranks for the relaxation. The experiment was made on a 500 variables, three states instance ($d = 1500$) and is representative of the behavior on all instances.

identity matrices) or MAXSAT (at the cost, however, of a significantly weakened relaxation for k-clauses, $k > 2$).

But a variety of real world problems expressed as optimization over graphical models, in resource allocation, computational biology or image processing, naturally involve variables with many states and arbitrary pairwise functions. Convex relaxations of discrete optimization problems such as MAXSAT have also been used as differentiable layers inside deep neural nets targeted at learning "how to reason" (Wang et al., 2019), a situation where categorical data with several states abound.

This work shows that the extension of the Burer-Monteiro approach to arbitrary domain sizes and functions is not straightforward: even if it improves over usual interior points methods, as implemented in CVXPY, the direct dualization of the required "exactly-one" constraints with adequate penalization leads to slow convergence, removing most of the practical interest of the Burer-Monteiro approach. Dedicated ways of dealing with these exactly-one constraints are needed. The BCD approach introduced here is a clear step forward in this direction. With its fast Newton updates, it offers important speed-ups compared to the dualized variant, at the sole cost of relaxing the gangster constraint. On the hardest instances, where LP-based methods can only provide very large gaps, the BCD approach we described is several orders of magnitude faster than interior points methods and orders of magnitude faster than the Lassere-dualized approach, while offering much tighter optimality gaps than LP. It becomes the method of choice for hard problems and can be used in parallel with the approximate linear bounds provided by convergent message passing to get the best of both worlds, at modest additional runtime.

The possibility of relying on extremely low rank is another attractive capacity of the Burer-Monteiro method. In our empirical experiments, we observe that the CPU-time that is required to produce an upper bound can be significantly reduced with essentially no loss in quality. A dual bound would then be able to produce an optimality gap that could be exploited in the context of Branch-and-Bound methods (Allouche et al., 2015), providing a desirable tight bounding method with an adjustable cpu/tightness compromise.

Acknowledgments

This work has been supported by the Artificial and Natural Intelligence Institute (ANITI) of Toulouse, France through ANR/PIA grant ANR-19-PI3A-0004 (V. Durante PhD thesis funding).

References

Absil, P.-A., Baker, C. G., and Gallivan, K. A. Trust-region methods on riemannian manifolds. *Foundations of Computational Mathematics*, 7(3):303–330, 2007.

Agrawal, A., Verschueren, R., Diamond, S., and Boyd, S. A rewriting system for convex optimization problems. *Journal of Control and Decision*, 5(1):42–60, 2018.

Allouche, D., Givry, S. d., Katsirelos, G., Schiex, T., and Zytnicki, M. Anytime hybrid best-first search with tree decomposition for weighted csp. In *International Conference on Principles and Practice of Constraint Programming*, pp. 12–29. Springer, 2015.

ApS, M. *MOSEK Fusion API for Python 9.3.12*, 2022. URL https://docs.mosek.com/latest/pythonfusion/index.html.

Barvinok, A. I. Problems of distance geometry and convex properties of quadratic maps. *Discrete & Computational Geometry*, 13(2):189–202, 1995.

Bensana, E., Lemaitre, M., and Verfaillie, G. Earth observation satellite management. *Constraints*, 4(3):293–299, 1999.

Burer, S. and Monteiro, R. D. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming*, 95(2):329–357, 2003.

Cooper, M., de Givry, S., and Schiex, T. Graphical models: queries, complexity, algorithms. *Leibniz International Proceedings in Informatics*, 154:4–1, 2020.

Cooper, M. C., De Givry, S., Sánchez-Fibla, M., Schiex, T., and Zytnicki, M. Virtual arc consistency for weighted CSP. In *AAAI*, pp. 253–258, 2008.

Cooper, M. C., De Givry, S., Sánchez, M., Schiex, T., Zytnicki, M., and Werner, T. Soft arc consistency revisited. *Artificial Intelligence*, 174(7-8):449–478, 2010.

Erdogdu, M. A., Ozdaglar, A., Parrilo, P. A., and Vanli, N. D. Convergence rate of block-coordinate maximization burer-monteiro method for solving large SDPs. *arXiv preprint arXiv:1807.04428*, 2018.

Goemans, M. X. and Williamson, D. P. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, 42(6):1115–1145, 1995.

Guennebaud, G., Jacob, B., et al. Eigen v3. http://eigen.tuxfamily.org, 2010.

Javanmard, A., Montanari, A., and Ricci-Tersenghi, F. Phase transitions in semidefinite relaxations. *Proceedings of the National Academy of Sciences*, 113(16):E2218–E2223, 2016.

Kappes, J. H., Andres, B., Hamprecht, F. A., Schnörr, C., Nowozin, S., Batra, D., Kim, S., Kausler, B. X., Kröger, T., Lellmann, J., et al. A comparative study of modern inference techniques for structured discrete energy minimization problems. *International Journal of Computer Vision*, 115(2):155–184, 2015.

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

Kolmogorov, V. Convergent tree-reweighted message passing for energy minimization. In *International Workshop on Artificial Intelligence and Statistics*, pp. 182–189. PMLR, 2005.

Krähenbühl, P. and Koltun, V. Efficient inference in fully connected CRFs with Gaussian edge potentials. *Advances in neural information processing systems*, 24:109–117, 2011.

Lasserre, J. B. A max-cut formulation of 0/1 programs. *Operations Research Letters*, 44(2):158–164, 2016.

Mei, S., Misiakiewicz, T., Montanari, A., and Oliveira, R. I. Solving SDPs for synchronization and maxcut problems via the grothendieck inequality. In *Conference on Learning Theory*, pp. 1476–1515. PMLR, 2017.

Nguyen, H., Bessiere, C., De Givry, S., and Schiex, T. Triangle-based consistencies for cost function networks. *Constraints*, 22(2):230–264, 2017.

Pabbaraju, C., Wang, P.-W., and Kolter, J. Z. Efficient semidefinite-programming-based inference for binary and multi-class mrfs. *Advances in Neural Information Processing Systems*, 33, 2020.

Park, S., Yang, E., Yun, S.-Y., and Shin, J. Spectral approximate inference. In *International Conference on Machine Learning*, pp. 5052–5061. PMLR, 2019.

Pataki, G. On the rank of extreme matrices in semidefinite programs and the multiplicity of optimal eigenvalues. *Mathematics of operations research*, 23(2):339–358, 1998.

Schiex, T., Fargier, H., Verfaillie, G., et al. Valued constraint satisfaction problems: Hard and easy problems. *IJCAI (1)*, 95:631–639, 1995.

Sherali, H. D. and Adams, W. P. A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. *SIAM Journal on Discrete Mathematics*, 3(3):411–430, 1990.

Shimony, S. E. Finding MAPs for belief networks is NP-hard. *Artificial intelligence*, 68(2):399–410, 1994.

Simoncini, D., Allouche, D., de Givry, S., Delmas, C., Barbe, S., and Schiex, T. Guaranteed discrete energy optimization on large protein design problems. *Journal of chemical theory and computation*, 11(12):5980–5989, 2015.

Sontag, D., Meltzer, T., Globerson, A., Jaakkola, T. S., and Weiss, Y. Tightening LP relaxations for MAP using message passing. *arXiv preprint arXiv:1206.3288*, 2012.

Thapper, J. and Živnỳ, S. The complexity of finite-valued csps. *Journal of the ACM (JACM)*, 63(4):1–33, 2016.

Wang, P.-W. and Kolter, J. Z. Low-rank semidefinite programming for the MAX2SAT problem. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI*, pp. 1641–1649, 2019.

Wang, P.-W., Chang, W.-C., and Kolter, J. Z. The mixing method: low-rank coordinate descent for semidefinite programming with diagonal constraints. *arXiv preprint arXiv:1706.00476*, 2017.

Wang, P.-W., Donti, P., Wilder, B., and Kolter, Z. Satnet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver. In *International Conference on Machine Learning*, pp. 6545–6554. PMLR, 2019.

Yedidia, J. S., Freeman, W. T., Weiss, Y., et al. Generalized belief propagation. In *NIPS*, volume 13, pp. 689–695, 2000.

Zhao, Q., Karisch, S. E., Rendl, F., and Wolkowicz, H. Semidefinite programming relaxations for the quadratic assignment problem. *Journal of Combinatorial Optimization*, 2(1):71–109, 1998.

A. Full proofs

A.1. Proof of Theorem 2.3

Proof. Let V_i^* be the value of V_i in an optimum of the problem above and assume that V_i^* does not lie in the dimension 2 subspace generated by g_i and V_{d+1}. Let $u = V_{d+1}, v$ be the orthonormal basis of this subspace such that g_i has a positive coordinate over v. Then, V_i can be written as $V_i = \alpha u + \beta v + \gamma w$ where w lies in the $r-2$ dimensional subspace orthogonal to the subspace generated by g_i and V_{d+1} and $\gamma \neq 0$. Since, when $\beta > 0$, changing the sign of β improves the criteria without changing the status of the "exactly-one" and norm-1 constraints, we must have $\beta < 0$. Let $\beta' = -\sqrt{\beta^2 + \gamma^2}$ and $V_i^{*\prime} = \alpha u + \beta' v$. If we now consider a solution where V_i^* has been replaced by $V_i^{*\prime}$, the "exactly one" constraint is still satisfied because the scalar product of $V_i^{*\prime}$ with V_{d+1} is unchanged. The norm 1 constraint is still satisfied as $||V_i^{*\prime}||^2 = \alpha^2 + \beta'^2 = \alpha^2 + \beta^2 + \gamma^2 = 1$. Then, the new solution improves the criteria given that $\beta' < \beta < 0$ and that $g_i^T v > 0$. □

A.2. Proof of Theorem 2.4

Proof. With V_i being norm one and lying in the dimension 2 subspace generated by g_i and V_{d+1}, let ψ_i be the angle from V_{d+1} to V_i in this subspace. We can rewrite the problem (7) as:

$$\min_{\psi_i \in [0,\pi]} \sum_{i=s_k}^{s_k+d_k-1} ||g_i|| \cos(\phi_i + \psi_i) \quad \text{s.t} \quad \sum_{i=s_k}^{s_k+d_k-1} \cos(\psi_i) = 2 - d_k \tag{12}$$

The Lagrangian is:

$$L(\lambda, \Psi) = \sum_{i=1}^{d_1} ||g_i|| \cos(\phi_i + \psi_i) + \lambda(\sum_{i=s_k}^{s_k+d_k-1} \cos(\psi_i) + d_k - 2)$$

The partial derivatives of L are $\frac{\partial L}{\partial \psi_i} = -||g_i|| \sin(\phi_i + \psi_i) - \lambda \sin(\phi_i)$ which must all be equal to zero at the optimum. We use the fact that the sum of sinusoids with the same period and different phases is also a sinusoid:

$$A \sin(\omega t + \phi) + B \sin(\omega t) = \sqrt{A^2 + B^2 + 2AB\cos(\phi)} \sin\left(\omega t + \arctan\left(\frac{A\sin(\phi)}{A\cos(\phi) + B}\right)\right)$$

We have:

$$-||g_i|| \sin(\phi_i + \psi_i) - \lambda \sin(\phi_i) = \sqrt{||g_i||^2 + \lambda^2 + 2\lambda||g_i|| \cos(\phi_i)} \sin\left(\psi_i + \arctan\left(\frac{||g_i|| \sin(\phi_i)}{||g_i|| \cos(\phi_i) + \lambda}\right)\right)$$

which implies that

$$\psi_i + \arctan\left(\frac{||g_i|| \sin(\phi_i)}{||g_i|| \cos(\phi_i) + \lambda}\right) = \pm\pi$$

Since $-\frac{3}{2}\pi < -\pi - \arctan\left(\frac{||g_i|| \sin(\phi_i)}{||g_i|| \cos(\phi_i)+\lambda}\right) < -\frac{\pi}{2} < 0$, we have:

$$\psi_i = \pi - \arctan\left(\frac{||g_i|| \sin(\phi_i)}{||g_i|| \cos(\phi_i) + \lambda}\right)$$

By trigonometric identities, noting $\gamma_i = \frac{||g_i|| \sin(\phi_i)}{||g_i|| \cos(\phi_i)+\lambda}$, we have:

$$\cos(\psi_i) = -\cos(\arctan(\gamma_i)) = -\frac{1}{\sqrt{1+\gamma_i^2}}$$

Developing and simplifying $(1 + \gamma_i^2)$ and plugging the result back above, we get:

$$\cos(\psi_i) = -\frac{||g_i|| \cos(\phi_i) + \lambda}{||g_i + \lambda V_{d+1}||}$$

Similarly, one can derive:

$$\cos(\phi_i + \psi_i) = -\frac{||g_i|| + \lambda\cos(\phi_i)}{||g_i + \lambda V_{d+1}||}$$

Plugging this back into the Lagrangian, we get:

$$\begin{aligned} L(\lambda, \Psi) &= \sum_{i=s_k}^{s_k+d_k-1}\left(-||g_i||\frac{||g_i|| + \lambda\cos(\phi_i)}{||g_i + \lambda V_{d+1}||}\right) + \lambda\left(\sum_{i=s_k}^{s_k+d_k-1}\left(-\frac{||g_i||\cos(\phi_i) + \lambda}{||g_i + \lambda V_{d+1}||}\right) + d_k - 2\right) \\ &= -\sum_{i=s_k}^{s_k+d_k-1}(||g_i + \lambda V_{d+1}||) + \lambda(d_k - 2) \end{aligned}$$

So the dual problem is to find λ that maximizes $h(\lambda) = -\sum_{i=s_k}^{s_k+d_k-1}(||g_i + \lambda V_{d+1}||) + \lambda(d_k - 2)$. The first derivative of h is:

$$h'(\lambda) = -\sum_{i=s_k}^{s_k+d_k-1}\left(\frac{g_i^T V_{d+1} + \lambda)}{||g_i + \lambda V_{d+1}||}\right) + (d_k - 2)$$

and the second derivative:

$$h''(\lambda) = -\sum_{i=s_k}^{s_k+d_k-1}\left(\frac{||g_i||^2 - (g_i^T V_{d+1})^2}{||g_i + \lambda V_{d+1}||^3}\right)$$

One can see that $h''(\lambda) < 0$ whenever g_i and V_{d+1} are not colinear which proves h is strictly concave and guarantees the unicity of the optimum. Being strictly concave, we just have to find λ^* such that $h'(\lambda^*) = 0$. □

A.3. Proof of Proposition 2.5

Proof. We need to show that the second derivative of f is Lipschitz on $\mathbb{R}$. In order to prove this, we will show that the derivative of f'' is bounded on $\mathbb{R}$. Let $u = V_{d+1}$, we will show that the derivative of each term $f_i(\lambda) = \frac{||g_i||^2 - (g_i^T u)^2}{||g_i + \lambda u||^3}$ in the sum is bounded.

$$\begin{aligned} f_i'(\lambda) &= -\frac{3(||g_i||^2 - (g_i^T u)^2)(g_i^T u + \lambda)}{||g_i + \lambda u||^5} \\ &= -\frac{3(||g_i||^2 - (g_i^T u)^2)(g_i^T u)}{||g_i + \lambda u||^5} - \frac{3(||g_i||^2 - (g_i^T u)^2)\lambda}{||g_i + \lambda u||^5} \end{aligned}$$

Using the assumption that g_i and u are not colinear there exists $k_i > 0$ such that $||g_i + \lambda u||^5 \geqslant k_i > 0$. Thus, we have :

$$\left|\frac{3(||g_i||^2 - (g_i^T u)^2)(g_i^T u)}{||g_i + \lambda u||^5}\right| \leqslant \frac{|3(||g_i||^2 - (g_i^T u)^2)(g_i^T u)|}{k_i}$$

Now we need to show that the second term is also bounded. Let $a_i = 3(||g_i||^2 - (g_i^T u)^2)$, we can rewrite the absolute value of the second term as :

$$\begin{aligned} \left|\frac{a_i\lambda}{||g_i + \lambda u||^5}\right| &= |a_i|\frac{(\lambda^2)^{\frac{1}{2}}}{(||g_i||^2 + 2g_i^T u\lambda + \lambda^2)^{\frac{5}{2}}} \\ &= |a_i|\left(\frac{\lambda^2}{(||g_i||^2 + 2g_i^T u\lambda + \lambda^2)^5}\right)^{\frac{1}{2}} \end{aligned}$$

If we consider the function :

$$l(\lambda) = \frac{\lambda^2}{(||g_i||^2 + 2g_i^T u\lambda + \lambda^2)^5}$$

We know that l is a continuous function on $\mathbb{R}$. Moreover, we have that :

$$l(\lambda) \underset{\lambda\to\pm\infty}{\sim} \frac{\lambda^2}{\lambda^{10}} = \frac{1}{\lambda^8} \underset{\lambda\to\pm\infty}{\longrightarrow} 0$$

We can conclude that l is a continuous function with finite limits at $\pm\infty$ thus it is bounded. So, we have the result that there exists $M \in \mathbb{R}_+^*$ such that the second term is bounded by $|a_i|\sqrt{M}$. □

A.4. Proof of Theorem 2.6

Let us first introduce a lemma that will be useful for the proof of this theorem.

Lemma A.1. *The solution v^* to (9) must lie in the subspace $vec(v', g)$.*

Let us consider $P_1 : \left\{v \in \mathbb{R}^n \mid v'^T v = 2 - d_k\right\}$ and $D_1 : \left\{v \in \mathbb{R}^n \mid ||v||^2 = d_k\right\}$. For $y \in P_1$, we have $y = y_0 + z$ with y_0 a particular solution to the linear equation $v'^T y_0 = 2 - d_k$ and z a vector such that $v'^T z = 0$. For y_0, we take $y_0 = \frac{2-d_i}{||v'||^2} v'$ which gives us the desired solution $v'^T y_0 = \frac{2-d_k}{||u||^2} v'^T v' = 2 - d_k$. Since $y \in D_1$, the squared norm of z must also satisfies $||z||^2 = d_k - (2 - d_k)^2$.
Then, we compute the dot product $g^T y = \frac{(2-d_k)}{||v'||^2} v'^T g + z^T g$. The first term is constant for all y that lies in the feasible set $P_1 \cap D_1$. We just have to check the value of the second term $z^T g$. Let $\Pi_{P_1}(g)$ be the projection of g into the hyperspace orthogonal to v'. We can write the dot product : $z^T g = z^T(g - \Pi_{P_1}(g) + \Pi_{P_1}(g)) = z^T(g - \Pi_{P_1}(g)) + z^T \Pi_{P_1}(g) = z^T \Pi_{P_1}(g)$ since $z^T(g - \Pi_{P_1}(g)) = 0$. Thus we only worry about the value of the dot product $z^T \Pi_{P_1}(g)$ which is minimal in the direction $-\Pi_{P_1}(g)$. Since $\Pi_{P_1}(g)$ lies in the space generated by v' and g we proved that the solution to (9) must lie in the space generated by v' and g.

Proof. Let us consider $v \in vec(v', g)$ e.g. $v = \alpha v' + \beta g$ for some $\alpha, \beta \in \mathbb{R}$. The first constraint gives us:

$$\alpha = -\frac{1}{d_k}(\beta v'^T g + d_k - 2)$$

By developing the squared norm we have:

$$\beta^2(\frac{1}{d_k}(v'^T g)^2 - \frac{2}{d_k}(v'^T g)^2 + ||g||^2) + \frac{(d_k-2)^2}{d_k} = d_k$$
$$\beta^2(||g||^2 - \frac{1}{d_k}(u^T g)^2) = d_k - \frac{(d_k-2)^2}{d_k}$$
$$\beta^2(||g||^2 - \frac{1}{d_k}(v'^T g)^2) = 4 - \frac{4}{d_k}$$

Using the Cauchy-Schwartz inequality we know that $||g||^2 - \frac{1}{d_k}(v'^T g)^2 > 0$ whenever v' and g are not colinear. Thus using the notation $p = 4 - \frac{4}{d_k}$:

$$\beta = \mp\sqrt{\frac{p}{\gamma}} \text{ with } \gamma = ||g||^2 - \frac{1}{d_k}(v'^T g)^2$$

One can see that the objective value is smaller for $\beta = -\sqrt{\frac{p}{\gamma}}$ which gives us the result. □

A.5. Proof of Proposition 2.7

Proof. Let us show that the objective difference before and after a BCD update of V is always positive. For a given column V_i the objective difference before and after the update is:

$$f(V_i) - f(\hat{V}_i) = g_i^T(V_i - \hat{V}_i)$$
$$\text{with } \hat{V}_i = \alpha_i V_{d+1} + \beta_i g_i$$
$$\Rightarrow g_i = \frac{1}{\beta_i}(\hat{V}_i - \alpha_i V_{d+1})$$
$$g_i^T(V_i - \hat{V}_i) = \frac{1}{\beta_i}(\hat{V}_i - \alpha_i V_{d+1})^T(V_i - \hat{V}_i)$$

$$= \frac{1}{\beta_i}\big((\hat{V_i}^T V_i - 1) - \alpha_i V_{d+1}^T (V_i - \hat{V_i})\big)$$

Let us first check the first term. We know that:

$$\frac{1}{\beta_i}(\hat{V_i} V_i - 1) = -\frac{1}{2\beta_i}||V_i - \hat{V_i}||^2$$

Moreover:

$$\beta_i = \frac{||g_i||cos(\phi_i+\psi_i) - cos(\psi_i)V_{d+1}^T g_i}{||g_i||^2 - (V_{d+1}^T g_i)^2}$$

Using the Cauchy-Schwartz inequality, we already know that $||g_i||^2 - (V_{d+1}^T g_i)^2 \geqslant 0$. Next we will rewrite the numerator:

$$\begin{aligned} ||g_i||cos(\phi_i + \psi_i) - cos(\psi_i)V_{d+1}^T g_i &= ||g_i||(cos(\phi_i + \psi_i) - cos(\psi_i)cos(\phi_i)) \\ &= -||g_i||sin(\phi_i)sin(\psi_i) \end{aligned}$$

Since $\phi_i \in [0, \pi]$ and $\psi_i \in [0, \pi]$ the sinus are positive so we can conclude that $\beta_i \leqslant 0$. We proved that the first term is positive, let us now check the second term.

$$\begin{aligned} -\alpha_i V_{d+1}^T (V_i - \hat{V_i}) &= -\alpha_i V_{d+1}^T (V_i - (\alpha_i V_{d+1} + \beta_i g_i)) \\ &= -\alpha_i V_{d+1}^T V_i + \alpha_i^2 + \alpha_i \beta_i V_{d+1}^T g_i. \end{aligned}$$

Using $\alpha_i = V_{d+1}^T V_i - \beta_i V_{d+1}^T g_i$ we have:

$$\begin{aligned} -\alpha_i V_{d+1}^T (V_i - \hat{V_i}) &= -(V_{d+1}^T V_i - \beta_i V_{d+1}^T g_i) V_{d+1}^T V_i + \alpha_i^2 + (V_{d+1}^T V_i - \beta_i V_{d+1}^T g_i)\beta_i V_{d+1}^T g_i \\ &= -(V_{d+1}^T V_i)^2 + \beta_i (V_{d+1}^T g_i)(V_{d+1}^T V_i) + \alpha_i^2 + \beta_i (V_{d+1}^T V_i)(V_{d+1}^T g_i) - \beta_i^2 (V_{d+1}^T g_i)^2 \\ &= 2\beta_i (V_{d+1}^T V_i)(V_{d+1}^T g_i) - \beta_i^2 (V_{d+1}^T g_i)^2 - (V_{d+1}^T V_i)^2 + \alpha_i^2 \\ &= -(\beta_i (V_{d+1}^T g_i) - V_{d+1}^T V_i)^2 + (V_{d+1}^T V_i - \beta_i (V_{d+1}^T g_i))^2 = 0 \end{aligned}$$

Thus, the second term is equal to 0 so we proved that $f(V_i) - f(\hat{V_i}) \geqslant 0$ □

A.6. Proof of Theorem 2.2

Before going through the proof, we describe how the gangster constraint can be formulated in our SDP. First, if we consider the Boolean vector b_k that corresponds to the variable $x_k \in X$, the goal of the gangster constraint is to ensure that the of-diagonal entries of the matrix $b_k b_k^T$ are all zeros. To ensure that b_k will satisfy the gangster constraint, we can use the following matrix $\mathbf{H}_k \in \mathbb{R}^{d_k \times d_k}$

$$\mathbf{H}_k = \begin{pmatrix} 0 & 1 & \cdots & 1 \\ 1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 1 \\ 1 & \cdots & 1 & 0 \end{pmatrix}$$

Thus, for each $k = 1, \ldots, n$, we must have, $b_k^T \mathbf{H}_k b_k = 0$. If we think in terms of the concatenated vector $b \in \mathbb{R}^d$ we must have $b^T \mathbf{H} b = 0$ with

$$\mathbf{H} = \begin{pmatrix} \mathbf{H}_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \mathbf{H}_n \end{pmatrix}, \mathbf{H} \in \mathbb{R}^{d \times d}$$

Now, going from $\{0,1\}$ to $\{-1,1\}$, $b^T\mathbf{H}b = 0 \iff c^T\mathbf{G}c + c^Tq + r = 0$ with $\mathbf{G} = \frac{1}{4}\mathbf{H},\ q = \frac{1}{2}\mathbf{H}1_d,\ r = \frac{1}{4}1_d^T\mathbf{H}1_d$. Using the enhanced vector $e = [c\ 1] \in \{-1,1\}^{d+1}$:

$$\text{tr}(\mathbf{R}ee^T) = 0 \text{ with } \mathbf{R} = \begin{pmatrix} \mathbf{G} & \frac{1}{2}q \\ \frac{1}{2}q^T & r \end{pmatrix} \in \mathbb{R}^{(d+1)\times(d+1)}$$

Then we can use the usual relaxation and the usual low-rank factorisation :

$$\mathbf{X} = ee^T \in \mathbb{R}^{(d+1)\times(d+1)},\ \text{tr}(\mathbf{R}\mathbf{X}) = 0$$
$$\mathbf{X} = \mathbf{V}\mathbf{V}^T,\ \text{tr}(\mathbf{R}\mathbf{V}\mathbf{V}^T) = 0$$

Proof.

$$tr(\mathbf{P}\mathbf{V}\mathbf{V}^T) = 0 \ \Rightarrow \mathbf{V} \text{ satisfies the exactly one constraint. With } \mathbf{P} = \begin{pmatrix} \mathbf{F}^T\mathbf{F} & -\mathbf{F}^Tu \\ -u^T\mathbf{F} & u^Tu \end{pmatrix} \succeq 0$$

We can factorize the matrix $\mathbf{P}$, $\mathbf{P} = \mathbf{S}^T\mathbf{S}$ with $\mathbf{S} = \begin{pmatrix} \mathbf{F} & -u \end{pmatrix} \in \mathbb{R}^{N\times(d+1)}$. Thus, $tr(\mathbf{P}\mathbf{V}\mathbf{V}^T) = 0$ becomes:

$$tr(\mathbf{S}^T\mathbf{S}\mathbf{V}\mathbf{V}^T) = 0 \iff tr(\mathbf{V}^T\mathbf{S}^T\mathbf{S}\mathbf{V}) = 0$$
$$tr((\mathbf{S}\mathbf{V})^T\mathbf{S}\mathbf{V}) = 0$$
$$||\mathbf{S}\mathbf{V}||^2 = 0$$
$$\mathbf{S}\mathbf{V} = 0$$

One can see that this last equality is equivalent to the exactly one constraint as in the BCD formulation (6). Now we will prove that

$$tr(\mathbf{P}\mathbf{V}\mathbf{V}^T) = 0 \Rightarrow tr(\mathbf{R}\mathbf{V}\mathbf{V}^T) = 0$$

with $\mathbf{R} = \begin{pmatrix} \mathbf{G} & \frac{1}{2}q \\ \frac{1}{2}q^T & r \end{pmatrix} \in \mathbb{R}^{(d+1)\times(d+1)}$ the gangster constraint matrix. We will rewrite $\mathbf{P} = \mathbf{M} + \mathbf{R}$ with $\mathbf{M}$ a matrix we will describe later. The objective is to show that:

$$tr(\mathbf{P}\mathbf{V}\mathbf{V}^T) = 0 \Rightarrow tr(\mathbf{M}\mathbf{V}\mathbf{V}^T) = 0$$

$$\text{Thus, } tr(\mathbf{P}\mathbf{V}\mathbf{V}^T) = 0 \Rightarrow tr((\mathbf{M}+\mathbf{R})\mathbf{V}\mathbf{V}^T) = 0$$

$$tr(\mathbf{P}\mathbf{V}\mathbf{V}^T) + tr(\mathbf{R}\mathbf{V}\mathbf{V}^T) = 0$$

$$tr(\mathbf{R}\mathbf{V}\mathbf{V}^T) = 0$$

First,

$$\mathbf{M} = \mathbf{P} - \mathbf{R} = \begin{pmatrix} \mathbf{D} & -\mathbf{F}^Tu - \frac{1}{2}q \\ -u^T\mathbf{F} - \frac{1}{2}q^T & u^Tu - r \end{pmatrix}$$

With $\mathbf{D} = \text{diag}(\frac{1}{4}) \in \mathbb{R}^{d\times d}$. For $i = 1, ..., d$

$$(-\mathbf{F}^Tu - \frac{1}{2}q)_i = \frac{d_i}{4} - \frac{1}{2} - \frac{1}{4}(d_i - 1) = -\frac{1}{4}$$

and

$$u^Tu - r = \sum_{i=1}^{n}(1 - \frac{d_i}{2})^2 - \frac{1}{4}\sum_{i=1}^{n} d_i(d_i - 1)$$

$$= n - d + \frac{d}{4} = n - \frac{3}{4}d$$

Next, $\text{tr}(\mathbf{MVV}^T) = \sum_i \sum_j M_{i,j} V_i^T V_j$ with $\begin{cases} M_{i,j} = \frac{1}{4},\ i = j,\ i,j = 1,...,d \\ M_{i,j} = -\frac{1}{4},\ i = d+1,\ j = 1,...,d \\ M_{i,j} = -\frac{1}{4},\ j = d+1,\ i = 1,...,d \\ M_{i,j} = n - \frac{3}{4}d,\ i = j = d+1 \end{cases}$

Thus,

$$tr(\mathbf{MVV}^T) = \sum_{i=1}^{d} M_{i,i}||V_i||^2 + 2\sum_{j=1}^{d} M_{d+1,j} V_j^T V_{d+1} + n - \frac{3}{4}d$$

$$= \frac{1}{4}d + n - \frac{3}{4}d - \frac{1}{2}\sum_{j=1}^{d} V_{d+1}^T V_j$$

We showed that $tr(\mathbf{PVV}^T) = 0 \Rightarrow \mathbf{V}$ satisfies the exactly-one constraint. We can simplify the last term:

$$\sum_{j=1}^{d} V_{d+1}^T V_j = \sum_{j=1}^{n} (2 - d_j)$$

Finally,

$$tr(\mathbf{MVV}^T) = \frac{1}{4}d + n - \frac{3}{4}d - \frac{1}{2}\sum_{j=1}^{n}(2 - d_j)$$

$$= \frac{1}{4}d + n - \frac{3}{4}d - n + \frac{1}{2}d = 0$$

We can now conclude that $tr(\mathbf{PVV}^T) = 0 \Rightarrow tr(\mathbf{MVV}^T) + tr(\mathbf{RVV}^T) = tr(\mathbf{RVV}^T) = 0$, so $\mathbf{V}$ satisfies the gangster constraint. □