# FAST OPTIMIZATION OF BOOLEAN QUADRATIC FUNCTIONS VIA ITERATIVE SUBMODULAR APPROXIMATION AND MAX-FLOW

*Aritra Konar and Nicholas D. Sidiropoulos*

Dept. of Electrical and Computer Engineering, University of Virginia, Charlottesville, VA

## ABSTRACT

We consider the NP–hard combinatorial optimization problem of minimizing arbitrary quadratic forms over the $\{0, 1\}$ (Boolean) lattice. While polynomial-time approximation algorithms do exist for such problems, they suffer from the practical drawback of being computationally involved – often a side effect of being agnostic to the combinatorial structure inherent in the problem. In this paper, we propose a computationally lightweight approximation alternative which specifically exploits the combinatorial structure of the problem. The key result underlying our approach is that any Boolean quadratic function can be expressed as a difference of quadratic submodular functions, which enables us to construct and iteratively minimize a sequence of global submodular upper bounds on the cost function. This entails solving a quadratic submodular function minimization problem at each step, which can be efficiently accomplished via the seminal Max-Flow algorithm. Overall, our algorithm performs iterative approximation by solving a sequence of maximum-flow problems. The merits of using this approach are illustrated via simulations which indicate the very favorable performance of our algorithm.

## 1. INTRODUCTION

Quadratically constrained quadratic programming (QCQP) constitutes an important class of optimization problems with pervasive applications in various engineering disciplines (see [1, 2] and references therein). This paper considers an important sub-class of QCQP where the goal is to minimize an arbitrary quadratic form over the $\{0, 1\}$ (Boolean) lattice. Such problems arise in performing maximum-likelihood $\{\pm 1\}$ multi-user detection [3], designing binary spreading codes in CDMA systems [4], and computing most probable configurations in undirected, discrete graphical models with $\{0, 1\}$ random variables [5]. The problem is non-convex and well known to be NP–hard in the worst-case. Consequently, the design of approximation algorithms which compute high-quality, albeit suboptimal solutions in polynomial-time, is well motivated for such problems.

The prevailing approximation approach is a technique known as *Semidefinite Relaxation* (SDR) [6], which uses matrix-lifting together with rank relaxation to obtain a relaxed, convex semidefinite-programming (SDP) problem from the original QCQP formulation. In general, the solution of the SDP problem is not guaranteed to be rank-1, and a randomized rounding step is performed on the SDP solution to obtain a feasible solution for the QCQP problem. In special cases [7–9], it is possible to derive theoretical approximation guarantees on the (expected) quality of the randomized solution obtained. More recently, the work of [10] introduced a SDR technique for obtaining non-trivial upper and lower bounds on the optimal value of QCQP problems involving minimization of *convex* quadratic forms over the general integer lattice. To the best of our knowledge, no such approximation bounds are currently known for SDR applied to the more general case of minimizing arbitrary quadratic forms (not necessarily convex) over the $\{0, 1\}$ lattice. Additionally, a practical limitation of employing SDR for approximating such problems is its potentially high computational complexity, which stems from the matrix-lifting step that effectively increases the number of problem variables by an order of magnitude.

Another pertinent approximation strategy for such a class of QCQP problems is the *Successive Convex Approximation* (SCA) [11–14] framework. In this approach, the quadratic cost function is expressed as a difference of convex quadratic forms, and each combinatorial $\{0, 1\}$ constraint is reformulated as a pair of quadratic inequalities. Thereafter, the non-convex functions appearing in both the cost and constraint set are linearized about the current solution to obtain a convex QCQP problem, which can be solved optimally using off-the-shelf convex programming solvers. The solution obtained is then set to be the linearization point in the subsequent iteration. Such an approach results in an *inner* approximation of the given QCQP problem via a sequence of convex QCQP problems. However, the cost of solving a generic convex QCQP problem at every step can be computationally demanding when the number of variables is large. Moreover, while SCA is a flexible framework for approximating general non-convex problems, it does not explicitly take into account the combinatorial properties any given problem may possess.

Is it possible to devise an efficient polynomial-time approximation scheme which exploits the combinatorial structure inherent in the problem at hand? In this paper, we provide an affirmative answer to the above question. Our key enabling result is the following fact which we establish in this paper: *every* Boolean quadratic function can be expressed as a difference of quadratic *submodular* functions. Submodular functions constitute a special class of discrete functions which are particularly notable for featuring a diminishing returns property, while also exhibiting several other interesting properties analogous to both convex and concave functions [15–17]. Leveraging the particular "convex" property that every submodular function possesses a (discrete) subgradient, we are able to construct a global submodular upper bound of the cost function at any given point of the $\{0, 1\}$ lattice. We exploit this attribute by using a discrete optimization variant of the majorization-minimization (MM) algorithm proposed in [18, 19], which iteratively minimizes a sequence of global submodular upper bounds on the quadratic cost function. At every step of the algorithm, we are required to solve a quadratic submodular function minimization problem, which is well known [5] to be equivalent to computing the maximum-flow in a certain weighted, directed graph, and can be efficiently solved in polynomial-time [20–22].

To summarize, the MM algorithm exploits the difference of sub-

Contact: (aritra,nikos)@virginia.edu.

modular functions structure of arbitrary quadratic forms over the $\{0,1\}$ lattice to perform iterative minimization by solving a sequence of Max-Flow problems. By design, the algorithm features monotonically non-increasing cost and is computationally lightweight compared to pre-existing alternatives. We provide experimental results carried out on synthetic data which indicate the promising performance of the approach.

## 2. PROBLEM STATEMENT AND BACKGROUND

Consider the problem of minimizing an arbitrary quadratic function over the $\{0,1\}^n$ hypercube

$$\min_{\mathbf{x}\in\{0,1\}^n} \left\{ f(\mathbf{x}) := \mathbf{x}^T\mathbf{A}\mathbf{x} + \mathbf{b}^T\mathbf{x} \right\} \tag{1}$$

where the matrix $\mathbf{A} \in \mathbb{S}^n$ is non-zero, symmetric and the vector $\mathbf{b} \in \mathbb{R}^n$. Note that (1) contains the following $\{0,1\}^n$ least-squares problem

$$\min_{\mathbf{x}\in\{0,1\}^n} \|\mathbf{P}\mathbf{x} - \mathbf{q}\|_2^2 \tag{2}$$

as a special case, where $\mathbf{P} \in \mathbb{R}^{m\times n}$ and $\mathbf{q} \in \mathbb{R}^n$ constitute the data variables. Not only are such problems known to be NP–hard in their general form, but it is also NP–hard to obtain constant-factor approximation guarantees [23]. Consequently, we will settle for approximate minimization in polynomial-time.

For our purpose, it will be convenient to reformulate (1) in terms of set-notation as follows: first, we define a ground set of finite elements $\mathcal{V} = [n] := \{1, \cdots, n\}$. Then, every Boolean vector $\mathbf{x} \in \{0,1\}^n$ can be represented as the indicator vector of a subset $\mathcal{S} \subseteq \mathcal{V}$, i.e., i.e., $\mathbf{x} = \mathbb{1}_\mathcal{S}$. Hence, problem (1) can be equivalently expressed as

$$\min_{\mathcal{S}\subseteq\mathcal{V}} \left\{ f(\mathcal{S}) := \mathbb{1}_\mathcal{S}^T\mathbf{A}\mathbb{1}_\mathcal{S} + \mathbf{b}^T\mathbb{1}_\mathcal{S} \right\} \tag{3}$$

where $f : 2^\mathcal{V} \to \mathbb{R}$ is a set function that assigns a real value to any subset $\mathcal{S} \subseteq \mathcal{V}$. Our polynomial-time approximation approach for (3) is centered around exploiting a diminishing returns property associated with certain set functions known as *submodularity*.

**Definition 1. [Submodular function]** A set function $f$ is said to be *submodular* if $f(\mathcal{A}\cup\{v\}) - f(\mathcal{A}) \geq f(\mathcal{B}\cup\{v\}) - f(\mathcal{B})$ for all $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{V}\setminus\{v\}$.

That is, given subsets $\mathcal{A} \subseteq \mathcal{B} \subseteq \mathcal{V}\setminus\{v\}$, the marginal gain obtained by adding an element $v$ to $\mathcal{A}$ does not increase when we add $v$ to the superset $\mathcal{B}$, which can be viewed as a natural diminishing returns property.

**Definition 2. [Modular function]** A set function $b$ is said to be modular if and only if there exists a vector $\mathbf{b} \in \mathbb{R}^n$ for all subsets $\mathcal{S} \subseteq \mathcal{V}$ such that $b(\mathcal{S}) = \mathbf{b}^T\mathbb{1}_\mathcal{S} = \sum_{e\in\mathcal{S}} \mathbf{b}(e)$.

Note that such a function satisfies the inequality in Definition 1 with equality. The link between submodular functions and quadratic functions has been formally established via the following proposition.

**Proposition 1.** [17, Proposition 6.3] A quadratic function $f : \mathcal{S} \to \mathbb{1}_\mathcal{S}^T\mathbf{A}\mathbb{1}_\mathcal{S} + \mathbf{b}^T\mathbb{1}_\mathcal{S}$ is submodular if and only if all off-diagonal elements of $\mathbf{A}$ are non-positive.

We remark that the special case of (3) where $f$ is submodular, corresponds to a (unconstrained) submodular function minimization (SFM) problem. In a seminal paper [24], it was established that such a class of combinatorial optimization problems can be solved to global optimality in polynomial-time via the ellipsoid algorithm. Follow-up work has produced several other optimal polynomial-time algorithms for SFM, ranging from combinatorial methods [25–27], to ones based on minimizing continuous, convex extensions of $f$ over the relaxed domain $[0,1]^n$ (see [15] for more details) via convex optimization techniques (e.g., subgradient methods [17], the Fujishige-Wolfe algorithm [28], and cutting-plane methods [29]).

Additionally, the quadratic form of $f$ in SFM can be further exploited to yield simpler and more efficient optimal algorithms for (3). The premise behind this line of work is the following observation of Kolmogorov *et al.*

**Proposition 2.** [5, Theorem 4.1] Every quadratic submodular function can be equivalently represented as a graph-cut function.

The above result implies that solving any instance of quadratic SFM (3) is equivalent to computing the cut-set in a certain weighted, undirected graph (explicitly constructed using $\mathbf{A}$ and $\mathbf{b}$) which minimizes the sum of weighted edges cut. Simply stated, in this case, (3) is equivalent to solving a Min-Cut problem. By utilizing the celebrated Max-Flow Min-Cut theorem of Ford and Fulkerson [20], this further implies that (3) can be optimally solved in polynomial-time by any algorithm capable of computing the maximum flow between a designated source and sink terminal in a weighted, directed graph obtained via an appropriate reduction from the Min-Cut problem. Owing to space limitations, we do not provide the explicit graph construction here; instead we refer the interested reader to [5, Section 4.2] for the details.

## 3. PROPOSED APPROACH

In this paper, we focus on the more general case of (3) where the quadratic function $f$ is not guaranteed to be submodular. Consequently, the aforementioned results regarding quadratic SFM do not apply directly. Since (3) is NP–hard in its general form, we aim to approximately minimize (3) via polynomial-time algorithms. The starting point of our approach is the following result regarding general set functions.

**Proposition 3.** [18, Lemma 4], [19, Lemma 3.1] Every set function $f$ can be expressed as a difference of submodular functions (DSF) $f(\mathcal{S}) = g(\mathcal{S}) - h(\mathcal{S}), \forall\, \mathcal{S} \subseteq \mathcal{V}$, for some submodular functions $g$ and $h$.

While determining such a characterization may not be easy for general $f$ [19], in the special case where $f$ is quadratic, such a characterization can always be easily found, as we now show.

**Proposition 4. [DSF representation of Boolean quadratic functions]** Note that any matrix $\mathbf{A}$ admits a representation of the form $\mathbf{A} := \mathbf{A}_1 - \mathbf{A}_2$, where $\mathbf{A}_1 = \min\{\mathbf{A}, \mathbf{0}\}$ and $\mathbf{A}_2 = \min\{-\mathbf{A}, \mathbf{0}\}$. It is obvious that both $\mathbf{A}_1$ and $\mathbf{A}_2$ have non-positive off-diagonal elements, which makes their associated quadratic forms submodular via Proposition 1. Defining $g(\mathcal{S}) := \mathbb{1}_\mathcal{S}^T\mathbf{A}_1\mathbb{1}_\mathcal{S} + \mathbf{b}^T\mathbb{1}_\mathcal{S}$ [1] and $h(\mathcal{S}) := \mathbb{1}_\mathcal{S}^T\mathbf{A}_2\mathbb{1}_\mathcal{S}$, it follows that (3) can be expressed as

$$\min_{\mathcal{S}\subseteq\mathcal{V}} \left\{ f(\mathcal{S}) := g(\mathcal{S}) - h(\mathcal{S}) \right\} \tag{4}$$

Hence, any quadratic set function can be represented as a difference of quadratic submodular functions by simply evaluating the sign of the entries in the matrix $\mathbf{A}$ and subsequently constructing $\mathbf{A}_1$ and $\mathbf{A}_2$.

[1] Note that the sum of a submodular and modular function is submodular.

In order to exploit the DSF structure inherent in our problem, we utilize a discrete optimization analogue of the majorization-minimization (MM) algorithm proposed in [18, 19]. The algorithm is iterative in nature and requires initialization from a starting set $\mathcal{S}_0 \subseteq \mathcal{V}$. At every subsequent iteration $k \in \mathbb{N}$, a tight modular lower bound $m_k(\mathcal{S}) := \mathbf{m}_k^T \mathbb{1}_{\mathcal{S}}$ of $h(\mathcal{S})$ is constructed about the current solution set $\mathcal{S}_k$ such that

$$h(\mathcal{S}) \geq m_k(\mathcal{S}), \forall\, \mathcal{S} \subseteq \mathcal{V}, \text{ and } h(\mathcal{S}_k) = m_k(\mathcal{S}_k). \tag{5}$$

From (5), we conclude that

$$f(\mathcal{S}) = g(\mathcal{S}) - h(\mathcal{S}) \leq g(\mathcal{S}) - m_k(\mathcal{S}), \forall\, \mathcal{S} \subseteq \mathcal{V} \tag{6a}$$

$$\text{and } \; f(\mathcal{S}_k) = g(\mathcal{S}_k) - h(\mathcal{S}_k) = g(\mathcal{S}_k) - m_k(\mathcal{S}_k) \tag{6b}$$

Hence, at each iteration $k$, we obtain a tight submodular upper bound $\phi_k(\mathcal{S}) = g(\mathcal{S}) - m_k(\mathcal{S})$ of $f(\mathcal{S})$. On replacing $f(\mathcal{S})$ by $\phi_k(\mathcal{S})$ in (3) at each iteration, we minimize an upper bound on $f(\mathcal{S})$. Note that the subproblem

$$\min_{\mathcal{S} \subseteq \mathcal{V}} \left\{ \phi_k(\mathcal{S}) = \mathbb{1}_{\mathcal{S}}^T \mathbf{A}_1 \mathbb{1}_{\mathcal{S}} + (\mathbf{b} - \mathbf{m}_k)^T \mathbb{1}_{\mathcal{S}} \right\} \tag{7}$$

that we are required to solve at each iteration corresponds to an instance of quadratic SFM. From the discussion earlier on in this section regarding solving quadratic SFM problems, we conclude that any Max-Flow algorithm can be utilized to solve (7).

From the preceding description of the algorithm, it is clear that the key ingredient required is the construction of the modular lower bound $m_k(\mathcal{S})$ of $h(\mathcal{S})$ at each iteration which satisfies the desired properties (5). Such a modular lower bound can be computed by utilizing the notion of submodular subdifferentials, which is formally defined as follows.

**Definition 3. [Subdifferential Sets of Submodular functions]** [16, Section 6.2] The subdifferential set of a submodular set function $h$ at a given set $\mathcal{Y} \subseteq \mathcal{V}$ is defined as

$$\partial h(\mathcal{Y}) = \{\mathbf{y} \in \mathbb{R}^n : h(\mathcal{X}) - h(\mathcal{Y}) \geq y(\mathcal{X}) - y(\mathcal{Y}), \forall\, \mathcal{X} \subseteq \mathcal{V}\} \tag{8}$$

where every vector $\mathbf{y} \in \partial h(\mathcal{Y})$ defines a modular function $y(\mathcal{X}) = \mathbf{y}^T \mathbb{1}_{\mathcal{X}}, \forall\, \mathcal{X} \subseteq \mathcal{V}$. Observe that (8) is reminiscent of the definition of subdifferential sets of convex functions. For the purpose of constructing our desired modular lower bound, we will be required to compute a subgradient $\mathbf{v}_{\mathcal{Y}}^h \in \partial h(\mathcal{Y})$ of $h$ at a given set $\mathcal{Y}$. In order to do so, it suffices to compute an extreme point of the subdifferential set $\partial h(\mathcal{Y})$. Towards this end, we use the fact that the set of extreme points which describe $\partial h(\mathcal{Y})$ admit the following characterization.

**Proposition 5. [Extreme points of submodular subdifferentials]** [16, Theorem 6.11] For each $\mathcal{Y} \subseteq \mathcal{V}$, a vector $\mathbf{v}_{\mathcal{Y}}^h$ is an extreme point of $\partial h(\mathcal{Y})$ if and only if there exists a maximal chain $\mathcal{C} : \emptyset = \mathcal{S}^{(0)} \subset \mathcal{S}^{(1)} \subset \cdots \subset \mathcal{S}^{(n)} = \mathcal{V}$ which includes $\mathcal{Y}$ (i.e., $\mathcal{Y} = \mathcal{S}^{(j)}$ for some $j \in [n]$) such that the modular function $v_{\mathcal{Y}}^h$ associated with $\mathbf{v}_{\mathcal{Y}}^h$ satisfies

$$\begin{aligned} v_{\mathcal{Y}}^h(\mathcal{S}^{(i)} \setminus \mathcal{S}^{(i-1)}) &= v_{\mathcal{Y}}^h(\mathcal{S}^{(i)}) - v_{\mathcal{Y}}^h(\mathcal{S}^{(i-1)}) \\ &= h(\mathcal{S}^{(i)}) - h(\mathcal{S}^{(i-1)}), \forall\, i \in [n] \end{aligned} \tag{9}$$

Using this description, Edmonds [30] prescribed the following approach for computing an extreme point of $\partial h(\mathcal{Y})$. Given a set $\mathcal{Y}$, let $\boldsymbol{\pi} \in \mathbb{R}^n$ represent a permutation of the ground set $\mathcal{V} = [n]$ which contains the elements of $\mathcal{Y}$ in its first $|\mathcal{Y}|$ positions, i.e., we have $\boldsymbol{\pi}(i) \in \mathcal{Y}, \forall\; i \leq |\mathcal{Y}|$. The remaining $n - |\mathcal{Y}|$ positions of $\boldsymbol{\pi}$ can be assigned randomly, provided that $\boldsymbol{\pi}$, in its entirety, is a valid permutation of $[n]$. Every such permutation vector can be used to construct a maximal chain $\mathcal{S}_{\boldsymbol{\pi}}^{(0)} \subset \mathcal{S}_{\boldsymbol{\pi}}^{(1)} \subset \cdots \subset \mathcal{S}_{\boldsymbol{\pi}}^{(n)}$ with elements $\mathcal{S}_{\boldsymbol{\pi}}^{(0)} = \emptyset$, and $\mathcal{S}_{\boldsymbol{\pi}}^{(i)} = \{\boldsymbol{\pi}(1), \boldsymbol{\pi}(2), \cdots, \boldsymbol{\pi}(i)\}, \forall\, i \in [n]$. Note that we have $\mathcal{S}_{\boldsymbol{\pi}}^{|\mathcal{Y}|} = \mathcal{Y}$. Using this chain, we define a vector $\mathbf{v}_{\mathcal{Y},\boldsymbol{\pi}}^h \in \mathbb{R}^n$ with entries

$$\mathbf{v}_{\mathcal{Y},\boldsymbol{\pi}}^h(\boldsymbol{\pi}(i)) = \begin{cases} h(\mathcal{S}_{\boldsymbol{\pi}}^{(1)}), & \text{if } i = 1 \\ h(\mathcal{S}_{\boldsymbol{\pi}}^{(i)}) - h(\mathcal{S}_{\boldsymbol{\pi}}^{(i-1)}), & \text{otherwise} \end{cases} \tag{10}$$

From the above construction, it can be verified that $\mathbf{v}_{\mathcal{Y},\boldsymbol{\pi}}^h$ satisfies the description of an extreme point of the subdifferential set of $\partial h(\mathcal{Y})$ as listed in Proposition 5. Using $\mathbf{v}_{\mathcal{Y},\boldsymbol{\pi}}^h$, we next define the following modular function

$$v_{\mathcal{Y},\boldsymbol{\pi}}^h(\mathcal{S}) := \sum_{e \in \mathcal{S}} \mathbf{v}_{\mathcal{Y},\boldsymbol{\pi}}^h(e) \tag{11}$$

for all subsets $\mathcal{S} \subseteq \mathcal{V}$. It can be shown that [24] for every set $\mathcal{Y} \subseteq \mathcal{V}$, the modular function $v_{\mathcal{Y},\boldsymbol{\pi}}^h(\mathcal{S})$ satisfies the following properties:

(A1) $v_{\mathcal{Y},\boldsymbol{\pi}}^h(\mathcal{S}) \leq h(\mathcal{S}), \forall\, \mathcal{S} \subseteq \mathcal{V}$

(A2) $v_{\mathcal{Y},\boldsymbol{\pi}}^h(\mathcal{S}_{\boldsymbol{\pi}}^{(i)}) = h(\mathcal{S}_{\boldsymbol{\pi}}^{(i)}), \forall\, i \in [n]$

Furthermore, from the last property, we obtain the condition

$$v_{\mathcal{Y},\boldsymbol{\pi}}^h(\mathcal{S}_{\boldsymbol{\pi}}^{|\mathcal{Y}|}) = v_{\mathcal{Y},\boldsymbol{\pi}}^h(\mathcal{Y}) = h(\mathcal{Y}) \tag{12}$$

Taken together, these conditions imply that the modular function $v_{\mathcal{Y},\boldsymbol{\pi}}^h(\mathcal{S})$ obtained by approximating the submodular function $h(\mathcal{S})$ about the given set $\mathcal{Y}$ satisfies the requisite properties (5) needed for use in the MM algorithm.

We now complete our description of the MM algorithm. Given the current solution set $\mathcal{S}_k$, we first compute a subgradient $\mathbf{v}_{\mathcal{S}_k,\boldsymbol{\pi}_k}^h \in \partial h(\mathcal{S}_k)$ via Edmond's procedure, which incurs complexity of the order $O(nM)$, where $M$ is an upper-bound on the complexity of evaluating $h$ at a set $\mathcal{S}$. The subgradient $\mathbf{v}_{\mathcal{S}_k,\boldsymbol{\pi}_k}^h$ defines a tight modular lower bound $v_{\mathcal{S}_k,\boldsymbol{\pi}_k}^h(\mathcal{S})$ of $h(\mathcal{S})$ about $\mathcal{S}_k$. In turn, this yields the quadratic submodular upper bound

$$\phi_k(\mathcal{S}) = g(\mathcal{S}) - v_{\mathcal{S}_k,\boldsymbol{\pi}_k}^h(\mathcal{S}) = \mathbb{1}_{\mathcal{S}}^T \mathbf{A}_1 \mathbb{1}_{\mathcal{S}} + (\mathbf{b} - \mathbf{v}_{\mathcal{S}_k,\boldsymbol{\pi}_k}^h)^T \mathbb{1}_{\mathcal{S}} \tag{13}$$

of $f(\mathcal{S})$ which is tight at $\mathcal{S} = \mathcal{S}_k$. Hence, at each iteration $k \in \mathbb{N}$, we are required to solve a quadratic SFM problem of the form (7), which can be accomplished by using a Max-Flow algorithm. The solution obtained is then updated to be the solution set $\mathcal{S}_{k+1}$ for the next iteration of the MM algorithm.

Overall, the algorithm exploits the DSF structure of the quadratic cost function $f(\mathcal{S})$ to successively minimize a sequence of global upper bounds of $f(\mathcal{S})$, while respecting the discrete nature of the constraints at every step.

**Algorithm 1** : **I**terative **M**ax **F**low (IMF)

**Initialization:** Set $k := 0, \mathcal{S}_0 \subseteq \mathcal{V}$ and generate a permutation vector $\boldsymbol{\pi}_0$ corresponding to $\mathcal{S}_0$.

**Repeat**

- Construct modular lower bound $v_{\mathcal{S}_k,\boldsymbol{\pi}_k}^h(\mathcal{S})$ of $h(\mathcal{S})$ about $\mathcal{S}_k$ via Edmond's procedure.
- $\mathcal{S}_{k+1} = \arg\min_{\mathcal{S} \subseteq \mathcal{V}} \{\phi_k(\mathcal{S}) := \mathbb{1}_{\mathcal{S}}^T \mathbf{A}_1 \mathbb{1}_{\mathcal{S}} + (\mathbf{b} - \mathbf{v}_{\mathcal{S}_k,\boldsymbol{\pi}_k}^h)^T \mathbb{1}_{\mathcal{S}}\}$ via a Max-Flow algorithm.
- Form random permutation $\boldsymbol{\pi}_{k+1}$ corresponding to $\mathcal{S}_{k+1}$.
- Set $k := k + 1$.

**Until** termination criterion is met

From the description of the algorithm, we obtain the following chain

of inequalities

$$f(\mathcal{S}_{k+1}) = g(\mathcal{S}_{k+1}) - h(\mathcal{S}_{k+1}) \tag{14a}$$
$$\leq g(\mathcal{S}_{k+1}) - v^h_{\mathcal{S}_k,\boldsymbol{\pi}_k}(\mathcal{S}_{k+1}) \tag{14b}$$
$$\leq g(\mathcal{S}_k) - v^h_{\mathcal{S}_k,\boldsymbol{\pi}_k}(\mathcal{S}_k) \tag{14c}$$
$$= g(\mathcal{S}_k) - h(\mathcal{S}_k) = f(\mathcal{S}_k) \tag{14d}$$

where the first inequality follows from the fact that $h(\mathcal{S}_{k+1}) \geq v^h_{\mathcal{S}_k,\boldsymbol{\pi}_k}(\mathcal{S}_{k+1})$, the second inequality is due to the optimality of $\mathcal{S}_{k+1}$ and the last inequality stems from the tightness of the modular approximation at $\mathcal{S} = \mathcal{S}_k$. Hence, the algorithm generates a sequence of solution sets $\{\mathcal{S}_k\}_{k\geq 0}$ with monotonically non-increasing cost.

**Remark 1:** While the approach outlined in this section concerns (approximate) minimization of quadratic functions subject to Boolean $\{0,1\}$ constraints, it can also be suitably modified to approximate Boolean quadratic *maximization* problems, i.e., problem (1) with minimization replaced by maximization. To see this, we again exploit the DSF structure of $f(\mathcal{S})$, albeit in a different manner. At each iteration $k \in \mathbb{N}$, we now construct a tight modular lower bound $m_k(\mathcal{S})$ for the submodular function $g(\mathcal{S})$ about the current solution set $\mathcal{S}_k$ while leaving $h(\mathcal{S})$ unchanged. On replacing $g(\mathcal{S})$ by its surrogate $m_k(\mathcal{S})$, we obtain the following maximization subproblem

$$\max_{\mathcal{S}\subseteq\mathcal{V}} \left\{ \psi_k(\mathcal{S}) = -\mathbb{1}_{\mathcal{S}}^T \mathbf{A}_2 \mathbb{1}_{\mathcal{S}} + (\mathbf{b} + \mathbf{m}_k)^T \mathbb{1}_{\mathcal{S}} \right\} \tag{15}$$

which now corresponds to maximizing a global *lower* bound on $f(\mathcal{S})$. Note that (15) is equivalent to $\min_{\mathcal{S}\subseteq\mathcal{V}}\{-\psi_k(\mathcal{S})\}$, which is again an instance of quadratic SFM, and thus, can be solved optimally in polynomial-time via a Max-Flow algorithm. Repeatedly applying this MM principle results in maximizing a sequence of global lower bounds on $f(\mathcal{S})$, and the algorithm now generates a sequence of solution sets $\{\mathcal{S}_k\}_{k\geq 0}$ with monotonically *non-decreasing* reward.

## 4. SIMULATION RESULTS

We carried out our experiments in MATLAB on a Windows desktop with 4 Intel i7 cores and 16GB of RAM. We also used MATLAB's in-built Max-Flow solver (which uses the algorithm of [22] by default) to solve the subproblems in the IMF algorithm.

An application in performing maximum-aposteriori (MAP) inference in undirected graphical models with Boolean $\{0,1\}$ random variables is considered to illustrate the performance of the algorithm. In this case, the joint probability distribution of $n$ random variables $\{X_i\}_{i=1}^n$ taking values $X_i = \{0,1\}, \forall\ i \in [n]$ is represented by a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with vertex set $\mathcal{V} = [n]$ and edge set $\mathcal{E} \subseteq [n]\times[n]$ that admits the following decomposition over pairwise cliques

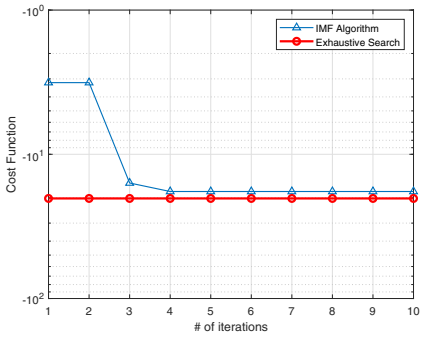
$$P(x_1, x_2, \cdots, x_n) = \frac{1}{Z} \exp(-H(x_1, x_2, \cdots, x_n)) \tag{16}$$

where $Z$ is a normalizing constant (i.e., the partition function) and

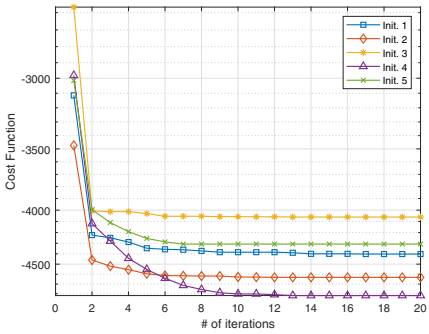
$$H(x_1, x_2, \cdots, x_n) := \sum_{i\in[n]} b_i x_i + \sum_{(i,j)\in\mathcal{E}} a_{ij} x_i x_j \tag{17}$$

denotes the energy function. Assuming that the model parameters $\{b_i\}_{i\in[n]}$ and $\{a_{ij}\}_{(i,j)\in\mathcal{E}}$ are known apriori, the problem of determining the most probable configuration seeks to find an assignment of random variables which maximizes (16), or equivalently, minimizes the energy function (17). Clearly, the problem is equivalent to minimizing an arbitrary quadratic form over the $\{0,1\}$ lattice.

Figure (1a) demonstrates the performance of the algorithm for a synthetically generated instance involving $n = 15$ variables. Here, the model parameters where generated from a standard normal distribution. The graph $\mathcal{G}$ was initially chosen to be fully-connected,




(a) Illustrative example on single instance with $n = 15$.




(b) Illustrative example on single instance with $n = 1000$.

and then a random subset of the edges was removed. We initialized our IMF algorithm from a randomly generated $\{0,1\}^n$ vector and performed 10 iterations of Max-Flow. Since the problem is small in size, we computed the optimal solution of the inference problem via exhaustive search to serve as a benchmark against IMF. It can be observed that the algorithm quickly converges to a solution with near optimal cost; the overall running time was $< 10$ milli-seconds. We noted that the choice of initialization has an impact on the quality of the attained solution. However, by using at most 4 re-initializations, we observed that IMF always attained a near optimal solution.

To demonstrate the scalability of the algorithm, we performed synthetic experiments with $n = 1000$ variables; a representative example showing the evolution of the cost function from different random initialization points is provided in Figure (1b). The average running time was $\sim 3.5$ seconds overall, which is very satisfactory. It can be observed that the algorithm is indeed affected by the choice of initialization. Unfortunately, we were unable to benchmark the performance of IMF in this regime since obtaining a lower bound from SDR proved to be too expensive in this case. In the future, we will explore the design of judicious initialization strategies to take advantage of IMF's inherent speed.

## 5. CONCLUSIONS

In this paper, we considered the NP–hard problem of minimizing an arbitrary quadratic form over the $\{0,1\}$ lattice and proposed an iterative approximation algorithm for the task. The algorithm exploits a difference of quadratic submodular functions representation of the cost function to successively construct and minimize a sequence of global submodular upper bounds, which entails solving a Max-Flow problem at each step. Experiments on synthetic data showcase the promising performance of the algorithm in terms of attained solution quality and running time.

## 6. REFERENCES


[1] Z.-Q. Luo, and T.-H. Chang, "SDP relaxation of homogeneous quadratic optimization: approximation bounds and applications", in *Convex Optimization in Signal Processing and Communications*, (D. Palomar and Y. Eldar, eds.), pp. 117–165, Cambridge University Press, 2010.

[2] A. d'Aspremont, and S. Boyd, "Relaxations and Randomized Methods for Nonconvex QCQPs," *EE392 Lecture Notes, Stanford University*, 2003.

[3] W.-K. Ma, T. N. Davidson, K. M. Wong, Z.-Q. Luo, and P.-C. Ching, "Quasi-maximum-likelihood multiuser detection using semidefinite relaxation with applications to synchronous CDMA", *IEEE Trans. Signal Process.*, vol. 50, no. 4, pp. 912–922, Aug. 2002.

[4] G. N. Karystinos, and A. P. Liavas, "Efficient computation of the binary vector that maximizes a rank-deficient quadratic form", *IEEE Trans. Inf. Theory*, vol. 56, no. 7, pp. 3581–3593, July 2010.

[5] V. Kolmogorov, and R. Zabih, "What energy functions can be minimized via graph cuts?", *IEEE Trans. Patt. Analys. and Mach. Intell.* vol. 1, no. 2, pp. 147–159, Jan. 2004.

[6] Z.-Q. Luo, W.-k. Ma, A.-C. So, Y. Ye and S. Zhang, "Semidefinite relaxation of quadratic optimization problems," *IEEE Signal Process. Mag.*, vol. 27, no. 3, pp. 20–34, May 2010.

[7] M. X. Goemans, and D. P. Williamson, "Improved approximation algorithms for maximum cut and satisfiability problems using semi-definite programming," *J. ACM*, vol. 42, no. 6, pp. 1115–1145, Nov. 1995.

[8] Y. Nesterov, "Semidefinite relaxation and nonconvex quadratic optimization," *Optim. Meth. Softw.*, vol. 9, pp. 140–160, 1998.

[9] S. Zhang, "Quadratic maximization and semidefinite relaxation," *Math. Program.*, vol. 87, pp. 453–465, May 2000.

[10] J. Park, and S. Boyd, "A semidefinite programming method for integer convex quadratic minimization," *Optim. Lett.*, vol. 12, no. 3, pp. 499–518, May 2018.

[11] B. Marks and G. Wright, "A general inner approximation algorithm for nonconvex mathematical programs," *Oper. Res.*, vol. 26, no. 4, pp. 681–683, 1978.

[12] A. Beck, A. Ben-Tal, and L. Tetruashvili, "A sequential parametric convex approximation method with applications to nonconvex truss topology design problems," *J. Global Optim.*, vol. 47, no. 1, pp. 29—51, 2010.

[13] G. Scutari, F. Facchinei, L. Lampariello, "Parallel and distributed methods for constrained nonconvex optimization-part I: Theory," *IEEE Trans. Signal Process.*, vol. 65, no. 8, pp. 1929–1944, April 2017.

[14] G. Scutari, F. Facchinei, L. Lampariello, S. Sardellitti, and P. Song, "Parallel and distributed methods for constrained nonconvex optimization-part II: Applications in communications and machine learning," *IEEE Trans. Signal Process.*, vol. 65, no. 8, pp. 1945–1960, April 2017.

[15] L. Lovasz, "Submodular functions and convexity", in *Mathematical Programming – The State of the Art*, pp. 235–257, Springer Berlin Heidelberg, 1983.

[16] S. Fujishige, "*Submodular functions and optimization*", 2nd edition, Annals of Disc. Math., vol. 58, 2005.

[17] F. Bach, "Learning with submodular functions: A convex optimization perspective," *Found. Trends in Mach. Learn.*, vol. 6, no. 2-3, pp. 145–373, Dec. 2013.

[18] M. Narasimhan, and J. Bilmes, "A submodular-supermodular procedure with applications to discriminative structure learning," *arXiv preprint arXiv:1207.1404*, 2012.

[19] R. Iyer, and J. Bilmes, "Algorithms for approximate minimization of the difference between submodular functions, with applications", *arXiv preprint arXiv:1207.0560*, 2013.

[20] L. Ford, and D. Fulkerson. "*Flows in Networks*", Princeton University Press, 1962.

[21] A. V. Goldberg, and R. E. Tarjan, "A new approach to the Maximum-Flow problem," *J. ACM*, vol. 35, no. 4, pp. 921–940, Oct. 1988.

[22] Y. Boykov, and V. Kolmogorov, "An experimental comparison of Min-Cut/Max-Flow algorithms for energy minimization in vision", *IEEE Trans. Patt. Analys. and Mach. Intell.* vol. 26, no. 9, pp. 1124–1137, Sept. 2004.

[23] S. Arora, L. Babai, J. Stern, and Z. Sweedyk, "The hardness of approximate optima in lattices, codes, and systems of linear equations", *J. Comp. Syst. Sciences*, vol. 54 , no. 2, pp. 317–331, April 1997.

[24] M. Grotschel, L. Lovasz, and A. Schrijver, "The ellipsoid algorithm and its consequences in combinatorial optimization", *Combinatorica*, vol. 1, no. 2, pp. 169—197, June 1981.

[25] A. Schrijver, "A combinatorial algorithm minimizing submodular functions in strongly polynomial time", *J. Combin. Theory* Ser. B, vol. 80, no. 2, pp. 346—355, Nov. 2000.

[26] S. Iwata, L. Fleischer, and S. Fujishige, "A combinatorial strongly polynomial algorithm for minimizing submodular functions," *J. ACM*, vol. 48, no. 4, pp. 761—777, July 2001.

[27] J. B. Orlin, "A faster strongly polynomial time algorithm for submodular function minimization," *Math. Program.*, vol. 118, no. 2, pp. 240—251, June 2009.

[28] S. Fujishige, and S. Isotani, "A submodular function minimization algorithm based on the minimum norm base", *Pacific J. of Optim.*, vol. 7, no. 1, pp. 3—17, Jan. 2011.

[29] Y. T. Lee, A. Sidford, and S. C. Wong, "A faster cutting plane method and its implications for combinatorial and convex optimization", in *Proc. IEEE FOCS*, pp. 1049–1065, Oct. 17-20, 2015, Berkley, CA.

[30] J. Edmonds, "Submodular functions, matroids and certain polyhedra", in *Combinatorial structures and their applications*, (G. Goos, J. Hartmanis, and J. van Leeuwen, eds.), vol. 11, Springer, 1970.