Approximation algorithms for semidefinite packing problems with applications to MAXCUT and graph coloring

G. Iyengar, D. J. Phillips, and C. Stein *

November 15, 2004

Abstract

We describe the semidefinite analog of the vector packing problem, and show that the semidefinite programming relaxations for MAXCUT [10] and graph coloring [16] are in this class of problems. We extend a method of Bienstock and Iyengar [4] which was based on ideas from Nesterov [24] to design an algorithm for computing ϵ-approximate solutions for this class of semidefinite programs. Our algorithm is in the spirit of Klein and Lu [17], and decreases the dependence of the run-time on ϵ from ϵ^{-2} to ϵ^{-1}. For sparse graphs, our method is faster than the best specialized interior point methods. A significant feature of our method is that it treats both the MAXCUT and the graph coloring problem in a unified manner.

1 Introduction

Semidefinite programming (SDP) has become a powerful tool for solving optimization problems. Lovász [21] applied semidefinite programming to model the Shannon-capacity of a graph, which, with the work of Grötschel, Lovász, and Schrijver [14], led to the first polynomial-time algorithm for finding the largest stable set in a perfect graph. Beginning with the work of Goemans and Williamson [10], semidefinite programming has been used as a tool for approximating NP-hard optimization problems. In this case, a semidefinite relaxation of the original problem is formulated and solved and then a rounding step is used to output a feasible and approximately optimal solution to the original problem. Since Goemans and Williamson used this technique to design approximation algorithms for MAXCUT, MAXDICUT and MAX2SAT, it has been used successfully by several other researchers. Karger, Motwani, and Sudan [16] use an SDP relaxation and rounding strategy to develop an approximation algorithm for the graph coloring problem. Skutella [29] used SDP to solve a scheduling problem, and recently, Arora, Rao and Vazirani [2] have used semidefinite programming to approximate graph partitioning problems. In all these problems, using the SDP relaxation yields better approximation bounds than using a linear programming relaxation or combinatorial techniques. In this paper we show that several of these and other SDPs appearing in the context of relaxations of combinatorial optimization problems can be viewed as *packing* problems over semidefinite matrices.

We will call an SDP

$$\begin{array}{rl} \max & \mathbf{C} \bullet \mathbf{X}, \\ \text{s.t.} & \mathbf{A}_i \bullet \mathbf{X} \leq 1, \quad i = 1, \ldots, m, \\ & \mathbf{X} \in \mathcal{X} \subseteq \mathbf{R}^{n \times n}, \end{array} \tag{1}$$

an SDP with *packing* constraints if it satisfies the following two conditions:

(i) Each $\mathbf{A}_i \succeq \mathbf{0}$ ($\mathbf{A} \succeq \mathbf{0}$ denotes that $\mathbf{A}$ is symmetric positive definite);

(ii) Optimizing a linear function over the set $\mathcal{X} \subseteq \{\mathbf{X} : \mathbf{X} \succeq \mathbf{0}\}$ is "easy." For example, let $\mathcal{X} = \{\mathbf{X} : \mathbf{X} \succeq \mathbf{0}, \mathbf{Tr}(\mathbf{X}) = a > 0\}$. Then $\max\{\mathbf{C} \bullet \mathbf{X} : \mathbf{X} \in \mathcal{X}\} = a \max\{\lambda_{\max}(\mathbf{C}), 0\}$, where $\lambda_{\max}(\mathbf{C})$ denotes the maximum eigenvalue of $\mathbf{C}$.

*First author partially supported by NSF grants CCR-00-09972, DMS-01-04282 and ONR grant N000140310514. Second author partially supported by NSF Grants DGE-0086390 and DMI-9970063. Third author partially supported by NSF Grant DMI-9970063. Department of IEOR, Columbia University, New York, NY. garud@ieor.columbia.edu, djp80@columbia.edu, cliff@ieor.columbia.edu.

The SDP with packing constraints is the natural extension of a vector problem with packing constraints [26] defined as follows.

$$\begin{array}{rll} \max & \mathbf{c}^T\mathbf{x}, & \\ \text{s.t.} & \mathbf{a}_i^T\mathbf{x} \leq 1, & i = 1, \ldots, m, \\ & \mathbf{x} \in \mathcal{X}, & \end{array} \tag{2}$$

where $\mathcal{X}$ is a polytope over which linear optimization is easy. The vector packing (and corresponding covering) problem has received a lot of attention in the combinatorial optimization community. Since (2) is a linear program (LP), an optimal solution can be found in polynomial time. However, in practice, for common applications such as multicommodity flow, it takes an extremely long time and large amount of memory to solve these problems to optimality [5]. If, on the other, one solves the problems to within a $1+\epsilon$ factor of optimality (called ϵ-optimality), the situation is much more encouraging. Leveraging the fact that LPs over $\mathcal{X}$ are "easy," one designs Lagrangian relaxation algorithms which "dualize" the constraints $\mathbf{a}_i^T\mathbf{x}$ with appropriate multipliers and reduce the packing problem to a series of LPs over $\mathcal{X}$. These ϵ-optimal approximation algorithms tend to be faster, both in theory and in practice. There are many factors involved in the analysis of such algorithms, here we will focus mainly on the dependence of the number of iterations on ϵ. Shahrokhi and Matula [28] developed the first approximation algorithm that computes an ϵ-optimal solution for an interesting special case of (2) (concurrent flow). The running time's dependence on ϵ is $O(\epsilon^{-7})$, and each iteration is linear optimization over $\mathcal{X}$. Subsequent research quickly reduced the dependence on ϵ to $O(\epsilon^{-2})$, and many other improvements reduced the number of iterations, the time per iteration, and expanded the techniques to broader packing and covering problems. See, e.g., [19, 20, 12, 13, 26, 27, 9, 8] for details. A recent breakthrough by Bienstock and Iyengar [4] employed a result of Nesterov [24] to reduce the dependence on ϵ to $O(\epsilon^{-1})$. Since the basic operation in all of these algorithms is linear optimization ([4] considers a regularized version), all of these methods are useful only when this step is cheap.

In this paper, we continue research into algorithms which have $O(\epsilon^{-1})$ dependence on ϵ, and extend the theory to SDPs with packing constrains. The main contributions in this paper are as follows.

(a) We show that several interesting SDPs, such as the Maxcut SDP, the graph coloring SDP, and the Lovász Shannon-capacity problem can all be cast as SDPs with packing constraints. This allows us to design algorithms for all these problems in a unified manner, leveraging the knowledge gained from designing algorithms for the vector packing problem.

(b) We extend the technique proposed by Nesterov [24] to design an algorithm for the SDP with packing constraints that computes an ϵ-optimal solution in $O(\frac{n \log n}{\epsilon})$ iterations, where each iteration solves a regularized linear optimization problem over the set $\mathcal{X} = \{\mathbf{X} : \mathbf{X} \succeq \mathbf{0}, \mathbf{Tr}(\mathbf{X}) \leq 1\}$, i.e. we reduce the SDP packing problem to a series of simple optimization problems over the set $\mathcal{X}$. As in the case of vector packing, such an algorithm will be attractive if linear optimization over $\mathcal{X}$ is cheap.

(c) Algorithms for vector packing problems, including the one in [4], do *not* yield a feasible solution (one notable exception is in [7]); instead, they compute an ϵ-feasible solution that is close to optimal. In contrast, the algorithm proposed in this paper computes ϵ-optimal *strictly* feasible solution for two special instances, namely the Maxcut SDP and the graph coloring SDP.

(d) We show that a regularized version of the linear optimization problem $\max\{\mathbf{W} \bullet \mathbf{X} : \mathbf{X} \in \mathcal{X}\}$, with $\mathbf{W} = \mathbf{W}^T$, can be solved in $O\left(n(n+m)\log^3(1/\epsilon)\right)$ time, where m denotes the maximum number of non-zero terms in $\mathbf{W}$.

Klein and Lu [17] (see also [18]) describe ϵ-approximation algorithms to solve the Maxcut SDP and the graph coloring SDP that build on an algorithm described in [26]. The number of iterations required to compute ϵ-optimal solution to the Maxcut SDP (resp. graph coloring SDP) grows in ϵ as $O(\epsilon^{-2})$ (resp. $O(\epsilon^{-4})$), where each iteration computes the maximum eigenvalue of a matrix. Since iterative methods allow efficient computation of the maximum eigenvalue when the underlying graph is sparse, algorithms in [17] are especially efficient for such graphs. This work, however, does not utilize the fact that both the Maxcut and the graph coloring SDPs are, in fact, particular instances of a more general problem class that can be efficiently solved. (Note the large difference in the run times for MAXCUT and graph coloring.)

Interior point algorithms can solve SDPs with packing constraints in time polynomial in the input size and logarithmic in the error ϵ [1, 25]. Specialized interior point methods [3, 6] for solving the MAXCUT SDP have a worst-case complexity of $O(n^{3.5}\log(\frac{1}{\epsilon}))$; in practice, however, the specialized methods perform faster than this worst-case bound. The theoretical complexity of general interior point methods for solving the graph coloring problem is $O(n^{6.5}\log(\frac{1}{\epsilon}))$. The significant difference in worst-case complexity is a consequence of the fact that the number of constraints increases from $O(n)$ to $O(n^2)$. No specialized interior point methods have been developed for graph coloring.

In contrast to interior point methods, the iteration count of the algorithm proposed in this paper is $O(\frac{n\log n}{\epsilon})$ independent of the number of constraints in the packing problem. Since any linear optimization problem over $\mathcal{X}$ can be solved by computing a spectral decomposition, each iteration in the packing algorithm is $O(n^3)$, yielding a worst case bound of $O(\frac{n^4\log n}{\epsilon})$. If the matrix is sparse, i.e., $m = O(n)$, we obtain a bound of $O(\epsilon^{-1}n^3\log(n)\log^3(\frac{1}{\epsilon}))$. Clearly, interior point methods will be superior to our algorithm for very small ϵ; thus, these methods will be competitive only for moderately small ϵ or in the presence of sparsity. In addition, as is the case with vector packing problem, the algorithms proposed here are interesting only for large problems with special structure that still allow cheap linear optimization over $\mathcal{X}$ but the interior point methods are not able to leverage the structure to reduce memory requirements.

Our presentation focuses on the MAXCUT and coloring problems, and briefly describes the Lovász Shannon-capacity problem. Our results apply to a broader class of SDPs but we do not pursue the details in this extended abstract. In Section 2, we give some notation and definitions. In Section 3, we describe the special case of the MAXCUT SDP and how solutions to this SDP can be computed from ϵ-optimal solutions to a related saddle-point problem. In Section 4, we describe the main algorithm that approximates the saddle-point problem. In Section 5, we describe how the graph coloring SDP can be approximated in an analogous fashion. These two instances will clearly imply the algorithm for general SDP packing problems. We are currently simplifying some of the details of the general algorithm. In Section 6, we describe the Lovász Shannon-capacity problem as an SDP with packing constraints. In Section 7, we describe how to exploit sparsity to improve the run-time of a bottleneck subroutine used to optimize over $\mathcal{X}$. The technical details for the algorithms approximating the MAXCUT SDP and graph coloring SDP can be found in Appendices A and B.

2 Notation and definitions

All vectors will be denoted by lowercase boldfaced letters, and matrices by capital boldfaced letters. Unless explicitly indicated, we use n dimensional column vectors and $n\times n$ matrices. We use $\mathbf{I}$ to denote the identity matrix and $\mathbf{1}$ for the vector of all ones. For a square matrix, $\mathbf{A} = [a_{ij}]$, define $\mathbf{diag}(\mathbf{A}) = [a_{11}\ a_{22}\ \ldots\ a_{nn}]^T$. For a vector $\mathbf{a}$, let $\mathbf{diag}(\mathbf{a}) = [d_{ij}]$ where $d_{ij} = a_i$ when $i = j$ and zero when $i \neq j$, i.e., $\mathbf{diag}(\mathbf{a})$ is the diagonal matrix with the vector $\mathbf{a}$ as the main diagonal. For matrices $\mathbf{A}$ and $\mathbf{B}$, we define $\mathbf{A}\bullet\mathbf{B} = \mathbf{Tr}(\mathbf{AB})$, and use $\mathbf{A}\succeq\mathbf{0}$ to indicate that $\mathbf{A}$ is a positive semidefinite matrix.

For a function $\Phi : \Theta\times\Upsilon \to \mathbf{R}$ consider the *saddle-point* problem

$$\max_{\mathbf{z}\in\Theta}\min_{\mathbf{p}\in\Upsilon}\Phi(\mathbf{z},\mathbf{p}). \tag{3}$$

For a given $\epsilon > 0$, we say that the pair $(\bar{\mathbf{z}},\bar{\mathbf{p}}) \in \Theta\times\Upsilon$ is an ϵ-*saddle-point* if,

$$0 \leq \min_{\mathbf{p}\in\Upsilon}\Phi(\bar{\mathbf{z}},\mathbf{p}) - \max_{\mathbf{z}\in\Theta}\Phi(\mathbf{z},\bar{\mathbf{p}}) \leq \epsilon. \tag{4}$$

Given a function $h : \Theta \to \mathbf{R}$ that we wish to minimize, let $\mathbf{z}^*$ be the minimum-valued solution. We say that $\bar{\mathbf{z}}$ is ϵ-optimal in the absolute sense if $h(\bar{\mathbf{z}}) \leq h(\mathbf{z}^*) + \epsilon$, i.e. $h(\bar{\mathbf{z}})$ is within an *additive* error ϵ to the optimal value. Suppose $h(\mathbf{z}^*) \geq C$. Then $h(\bar{\mathbf{z}}) \leq h(\mathbf{z}^*) + \epsilon = h(\mathbf{z}^*) + C(\epsilon/C) \leq (1+\epsilon/C)h(\mathbf{z}^*)$, thus, an ϵ-optimal solution in the absolute sense has a *relative* error at most ϵ/C. If C is a constant, then an ϵ-optimal solution in the absolute sense is an ϵ-optimal solution in the more traditional multiplicative sense. (We will make analogous definitions for maximization problems.) This relation between the absolute and relative error in each of these problems was described by Klein and Lu [17]. Our algorithms will actually return an ϵ-optimal primal-dual pair, i.e. a pair of primal and dual solutions whose objective values differ by no more than ϵ. By standard strong duality arguments, this immediately implies that both the primal and the dual are ϵ-optimal.

Let $\boldsymbol{\gamma} \in \mathbf{R}^n$. Then the following linear program has a simple solution.

$$\max_{\mathbf{y}} \Big\{ \sum_{i=1}^{n} \gamma_i y_i : \mathbf{y} \geq 0, \sum_{i=1}^{n} y_i \leq 1 \Big\} = \max \big\{ 0, \max_{1 \leq i \leq n} \{\gamma_i\} \big\}. \tag{5}$$

For a square matrix $\mathbf{A}$, $\lambda_1(\mathbf{A}) \leq \lambda_2(\mathbf{A}) \leq \ldots \leq \lambda_n(\mathbf{A})$ will denote the ordered set eigenvalues of $\mathbf{A}$, $\lambda_{\max}(\mathbf{A}) := \lambda_n(\mathbf{A})$ and $\lambda_{\min} := \lambda_1(\mathbf{A})$. Note that for all square matrices $\mathbf{A}$, the following optimization problem reduces to a linear program identical to (5) in the space of eigenvalues:

$$\begin{aligned} \max_{\mathbf{X}} \big\{ \mathbf{A} \bullet \mathbf{X} : \mathbf{Tr}(\mathbf{X}) \leq 1, \mathbf{X} \succeq 0 \big\} &= \max_{\mathbf{y}} \Big\{ \sum_{i=1}^{n} \lambda_i(\mathbf{A}) y_i : \mathbf{y} \geq 0, \sum_{i} y_i \leq 1 \Big\} \\ &= \max \big\{ 0, \lambda_{\max}(\mathbf{A}) \big\}. \end{aligned} \tag{6}$$

3 The SDP relaxation for Maxcut

In this section we review the well-known SDP relaxation of the exact vector formulation for the Maxcut problem. We then show the equivalence of this relaxation to a maximin problem. The exact vector formulation for Maxcut is as follows.

$$\begin{array}{rl} \max & \frac{1}{4}\mathbf{W} \bullet (\mathbf{1}\mathbf{1}^T - \mathbf{X}) \\ \text{subject to} & \mathbf{X} = \mathbf{x}\mathbf{x}^T, \\ & \mathbf{x} \in \{-1, 1\}^n, \end{array} \tag{7}$$

where $\mathbf{W}$ is the weight matrix. The *Laplacian* $\mathbf{L} = [\ell_{ij}]$ of a weighted graph with weights $\mathbf{W}$ is given by

$$\ell_{ij} = \begin{cases} -w_{ij}, & i \neq j, \\ \sum_{k=1}^{n} w_{ik}, & i = j. \end{cases} \tag{8}$$

The Laplacian $\mathbf{L}$ of a graph is a positive semidefinite matrix. From (8) it follows that (7) is equivalent to

$$\begin{array}{rl} \max & \frac{1}{4}\mathbf{L} \bullet \mathbf{X} \\ \text{subject to} & \mathbf{X} = \mathbf{x}\mathbf{x}^T \\ & \mathbf{x} \in \{-1, 1\}^n, \end{array} \tag{9}$$

The SDP relaxation of (9) employed by the Goemans-Williamson [10] approximation algorithm is equivalent to the following

$$\begin{array}{rl} \max & \mathbf{L} \bullet \mathbf{X} \\ \text{subject to} & \mathbf{diag}(\mathbf{X}) \leq \mathbf{1}, \\ & \mathbf{X} \in \bar{\mathcal{X}} \equiv \{\mathbf{X} : \mathbf{X} \succeq \mathbf{0}, \mathbf{Tr}(\mathbf{X}) \leq n\}. \end{array} \tag{10}$$

Note that relaxing $\mathbf{diag}(\mathbf{X}) = \mathbf{1}$ to $\mathbf{diag}(\mathbf{X}) \leq \mathbf{1}$ does not change the formulation, because increasing the diagonal of a positive semidefinite matrix keeps it positive semidefinite, and only increases the objective. The extra constraint $\mathbf{Tr}(\mathbf{X}) \leq n$ in the definition of $\mathcal{X}$ is implied by $\mathbf{diag}(\mathbf{X}) \leq \mathbf{1}$. Since linear optimization over $\bar{\mathcal{X}}$ reduces to a problem of the form (6), it follows that (10) is an SDP with packing constraints. We assume that the edge weights $\{w_{ij}\}$ sum to 1, i.e. $\mathbf{L} \bullet \mathbf{I} = 2$. Since $\mathbf{I}$ is feasible for (10), it follows that the optimal value of (10) is at least 2.

On dualizing the constraints $\mathbf{diag}(\mathbf{X}) - \mathbf{1}$ we get the saddle-point problem

$$\max_{\{\mathbf{X}:\mathbf{X}\succeq\mathbf{0},\mathbf{Tr}(\mathbf{X})\leq n\}} \max_{\{\mathbf{u}:\mathbf{u}\geq\mathbf{0}\}} \Big\{\mathbf{L}\bullet\mathbf{X} - \sum_{i=1}^{n} u_i(x_{ii}-1)\Big\}. \tag{11}$$

We want to compute a good solution $\widehat{\mathbf{X}}$ for (10) by starting from an initial $\mathbf{X}^{(0)} \succeq \mathbf{0}$ with $\mathbf{Tr}(\mathbf{X}^{(0)}) \leq n$, and then iterating by choosing the next dual iterate $\mathbf{u}^{(k+1)}$ (resp. primal iterate $\mathbf{X}^{(k+1)}$) to be the "best response" to current primal iterate $\mathbf{X}^{(k)}$ (resp. dual iterate $\mathbf{u}^{(k)}$). However, this is impossible unless we are able to bound the "width" of the set of dual variables. In Theorem 1, we show that it is sufficient to restrict to dual variables $\mathbf{u}$ such that $\sum_{i=1}^{n} u_i \leq 5n$. We will find it more convenient to work with the following scaled version of the saddle-point problem (11):

$$\max_{\mathbf{X}\in\mathcal{X}} \min_{\mathbf{u}\in\mathcal{U}} \phi(\mathbf{X},\mathbf{u}) = \min_{\mathbf{u}\in\mathcal{U}} \max_{\mathbf{X}\in\mathcal{X}} \phi(\mathbf{X},\mathbf{u}),$$

where

$$\begin{aligned} \phi(\mathbf{X},\mathbf{u}) &= n\mathbf{X}\bullet\mathbf{L} - \sum_{i=1}^{n} 5nu_i(nx_{ii}-1), && (12)\\ \mathcal{X} &= \Big\{\mathbf{X}\in\mathbf{R}^{n\times n} : \mathbf{X}\succeq 0, \mathbf{Tr}(\mathbf{X})\leq 1\Big\}, && (13)\\ \mathcal{U} &= \Big\{\mathbf{u}\geq 0 : \sum_{i=1}^{n} u_i \leq 1\Big\}. && (14) \end{aligned}$$

In the proof of the following result we use the fact that the dual of (10) is given by

$$\begin{array}{rl} \min & \sum_{i=1}^{n} u_i \\ \text{subject to} & \mathbf{diag}(\mathbf{u}) - \mathbf{L} \succeq \mathbf{0}, \\ & \mathbf{u} \geq 0. \end{array} \tag{15}$$

Theorem 1 *Fix $\epsilon > 0$. Suppose $(\bar{\mathbf{X}}, \bar{\mathbf{u}}) \in \mathcal{X}\times\mathcal{U}$ is an ϵ-saddle-point with respect to ϕ satisfying (4). Let $\bar{d} = n\max_{1\leq i\leq n}\{\bar{x}_{ii}\}$, $\bar{\lambda} = \lambda_{\max}(\mathbf{L} - 5n\,\mathbf{diag}(\bar{\mathbf{u}}))$, and*

$$\begin{aligned} \widehat{\mathbf{X}} &= \begin{cases} n\bar{\mathbf{X}} & \bar{d}\leq 1, \\ n\bar{\mathbf{X}}/\bar{d}, & \textit{otherwise;} \end{cases} \\ \widehat{\mathbf{u}} &= \begin{cases} 5n\bar{\mathbf{u}}, & \bar{\lambda}\leq 0, \\ 5n\bar{\mathbf{u}} + \bar{\lambda}\mathbf{1}, & \textit{otherwise.} \end{cases} \end{aligned} \tag{16}$$

Then $(\widehat{\mathbf{X}}, \widehat{\mathbf{u}})$ are an ϵ-optimal primal-dual pair for (10) and (15).

Proof: From the definition of $\bar{d}$, it follows that $\widehat{\mathbf{X}}$ is feasible for (10). Since

$$\mathbf{diag}(\widehat{\mathbf{u}}) - \mathbf{L} = 5n\,\mathbf{diag}(\bar{\mathbf{u}}) + \max\big\{0, \lambda_{\max}(\mathbf{L} - \mathbf{diag}(5n\bar{\mathbf{u}}))\big\}\mathbf{I} - \mathbf{L} \succeq \mathbf{0},$$

it follows that $\widehat{\mathbf{u}}$ is feasible for (15). From the definition of $\mathcal{X}$ and (6) it follows that

$$\begin{aligned} \max_{\mathbf{X}\in\mathcal{X}} \phi(\mathbf{X},\bar{\mathbf{u}}) &= \max_{\mathbf{X}\in\mathcal{X}} \Big\{n\mathbf{X}\bullet(\mathbf{L} - 5n\,\mathbf{diag}(\bar{\mathbf{u}})) + 5n\sum_{i=1}^{n}\bar{u}_i\Big\} \\ &= 5n\sum_{i=1}^{n}\bar{u}_i + n\max\big\{0, \lambda_{\max}\big(\mathbf{L} - 5n\,\mathbf{diag}(\bar{\mathbf{u}})\big)\big\} = \sum_{i=1}^{n}\widehat{u}_i. \end{aligned} \tag{17}$$

From the definition of $\mathcal{U}$ and (5), we have that $\min_{\mathbf{u}\in\mathcal{U}} \phi(\bar{\mathbf{X}},\mathbf{u}) = n\bar{\mathbf{X}}\bullet\mathbf{L} - \max_{\mathbf{u}\in\mathcal{U}}\Big\{5n\sum_{i=1}^{n} u_i(n\bar{x}_{ii}-1)\Big\} = n\bar{\mathbf{X}}\bullet\mathbf{L} - 5n\max\big\{0, \bar{d}-1\big\}$. We show below that $\mathbf{L}\bullet\widehat{\mathbf{X}}$ is at least $\min_{\mathbf{u}\in\mathcal{U}} \phi(\bar{\mathbf{X}},\mathbf{u})$.

(i) $\bar{d} \leq 1$: Then $\mathbf{L} \bullet \widehat{\mathbf{X}} = n\mathbf{L} \bullet \bar{\mathbf{X}} = n\mathbf{L} \bullet \bar{\mathbf{X}} - 5n \max\{0, \bar{d} - 1\} = \min_{\mathbf{u} \in \mathcal{U}} \phi(\bar{\mathbf{X}}, \mathbf{u})$.

(ii) $\bar{d} > 1$: Since $\mathbf{Tr}(\bar{\mathbf{X}}) \leq 1$, we have $|\widehat{x}_{ij}| \leq \sqrt{\bar{x}_{ii}\bar{x}_{jj}} \leq 1$, and $0 \leq \mathbf{L} \bullet \bar{\mathbf{X}} \leq \sum_{i,j=1}^{n} |\ell_{ij}||\bar{x}_{ij}| \leq \sum_{i,j=1}^{n} |\ell_{ij}| \leq 5$. For all $d > 0$, we have $1/d \geq 1 - (d-1)$, therefore,

$$\mathbf{L} \bullet \widehat{\mathbf{X}} = \frac{n\mathbf{L} \bullet \bar{\mathbf{X}}}{\bar{d}} \geq n\mathbf{L} \bullet \bar{\mathbf{X}} - n(\bar{d} - 1)(\mathbf{L} \bullet \bar{\mathbf{X}}) \geq n\mathbf{L} \bullet \bar{\mathbf{X}} - 5n(\bar{d} - 1) = \min_{\mathbf{u} \in \mathcal{U}} \phi(\bar{\mathbf{X}}, \mathbf{u}). \tag{18}$$

From (4), (17), and (18) we have that $\sum_{i=1}^{n} \widehat{u}_i - \mathbf{L} \bullet \widehat{\mathbf{X}} \leq \epsilon$, i.e. $(\widehat{\mathbf{X}}, \widehat{\mathbf{u}})$ is an ϵ-optimal primal-dual pair. ■

4 Computing the approximate saddle-point $(\bar{\mathbf{X}}, \bar{\mathbf{u}})$

In order to "implement" Theorem 1 we need to compute an ϵ-saddle-point, $(\bar{\mathbf{X}}, \bar{\mathbf{u}})$, for ϕ. In this section we describe how to extend a technique proposed in Nesterov [24], to compute such a point. The technical details of the method are provided in Appendix A.

We use the shorthand $\lambda_i = \lambda_i(\mathbf{L} - 5n\,\mathbf{diag}(\mathbf{u})), i = 1, \ldots, n$, and define $f : \mathcal{U} \to \mathbf{R}$ as follows.

$$f(\mathbf{u}) = \max_{\mathbf{X} \in \mathcal{X}} \phi(\mathbf{X}, \mathbf{u}) = 5n \sum_{i=1}^{n} u_i + n \max\{0, \lambda_{\max}\}, \tag{19}$$

where $\lambda_{\max} = \lambda_n = \max_{1 \leq i \leq n}\{\lambda_i\}$, and the second equality follows from (6). We compute a pair $(\bar{\mathbf{X}}, \bar{\mathbf{u}})$ such that $f(\bar{\mathbf{u}}) - \min_{\mathbf{u} \in \mathcal{U}} \phi(\bar{\mathbf{X}}, \mathbf{u}) \leq \epsilon$, i.e. it is an ϵ-saddle-point.

Our algorithm is based on a technique developed by Nesterov [24]. Since evaluating f involves solving an LP (see (6)), it is not differentiable. Therefore, we replace it by a smooth approximation f_α (see (20) below). This approach is very similar to the approach taken in the packing-covering literature (see, e.g. [26, 12, 4, 7]). In each iteration t, the algorithm computes a primal iterate $\mathbf{X}^{(t)} \in \mathcal{X}$ such that the gradient $\nabla f_\alpha(\mathbf{u}^{(t)}) = 5n(\mathbf{1} - n\,\mathbf{diag}(\mathbf{X}^{(t)}))$. This gradient is then used to compute the next dual iterate. The ϵ-saddle-point $(\bar{\mathbf{X}}, \bar{\mathbf{u}})$ is a weighted combination of all the primal-dual iterates generated by the algorithm.

Define f_α as follows.

$$f_\alpha(\mathbf{u}) = 5n \sum_{i=1}^{n} u_i + \frac{1}{\alpha} \ln\left(1 + \sum_{i=1}^{n} e^{\alpha n \lambda_i}\right). \tag{20}$$

Straightforward calculations show that for any $\alpha > 0$, $f(\mathbf{u}) \leq f_\alpha(\mathbf{u}) \leq f(\mathbf{u}) + \frac{\ln(n+1)}{\alpha}$. Therefore, by setting $\alpha = \frac{2\log(n+1)}{\epsilon}$, our problem reduces to computing an $\frac{\epsilon}{2}$-optimal solution for f_α.

Let $\mathbf{V}\,\mathbf{diag}(\boldsymbol{\lambda})\mathbf{V}^T = \mathbf{L} - 5n\,\mathbf{diag}(\mathbf{u})$ denote the eigendecomposition of $\mathbf{L} - 5n\,\mathbf{diag}(\mathbf{u})$. Then the gradient $\nabla f_\alpha(\mathbf{u})$ of $f_\alpha(\mathbf{u})$ is given by

$$\nabla f_\alpha(\mathbf{u}) = 5n(\mathbf{1} - n\,\mathbf{diag}(\mathbf{X}_u)), \tag{21}$$

where

$$\mathbf{X}_u = \frac{\mathbf{diag}\left(\mathbf{V}\,\mathbf{diag}(e^{n\alpha\boldsymbol{\lambda}})\mathbf{V}^T\right)}{\left(1 + \sum_{i=1}^{n} e^{n\alpha\lambda_i}\right)}, \tag{22}$$

and $\mathbf{diag}(e^{n\alpha\boldsymbol{\lambda}})$ denotes a diagonal matrix with $e^{n\alpha\lambda_i}$ as the i-th entry. Computing this gradient will be the subject of Section 7. For now we assume the existence of a procedure SmoothGrad which takes as input a vector $\mathbf{u}$ and returns $\nabla f_\alpha(\mathbf{u})$ and $\mathbf{X}_u$ as defined in equations (21) and (22).

Once the gradient is explicitly known, a Frank-Wolfe-type gradient descent method takes $O(\epsilon^{-2})$ iterations to compute ϵ-optimal $\bar{\mathbf{u}}$. The method proposed by Klein and Lu [17] can be interpreted as such a first-order method. In order to develop an algorithm in which the number of iterations grows as $O(\epsilon^{-1})$ one has to use a second-order Taylor series expansion and a more involved procedure for the inner loop. In Appendix A, we prove that for all $\mathbf{u}, \mathbf{u}' \in \mathcal{U}$

$$f_\alpha(\mathbf{u}) \leq f_\alpha(\mathbf{u}') + \nabla f_\alpha(\mathbf{u}')^T(\mathbf{u} - \mathbf{u}') + \frac{n^2\alpha}{2}\|\mathbf{u} - \mathbf{u}'\|_1^2, \tag{23}$$

```
SmoothApprox($\mathbf{L}, \epsilon$)
    $T \leftarrow 4n\log(n+1)/\epsilon$;    $\alpha \leftarrow 2n\log(n+1)/\epsilon$;    $\mathbf{u}^{(0)} \leftarrow \frac{1}{n+1}\mathbf{1}$
    $(\mathbf{g}^{(0)}, \mathbf{X}^{(0)}) \leftarrow$ SmoothGrad($\mathbf{u}^{(0)}$)
    $\mathbf{s} \leftarrow \frac{1}{2}\mathbf{g}^{(0)}$;    $\widehat{\mathbf{X}} \leftarrow \mathbf{X}^{(0)}$
    $\mathbf{y}^{(0)} \leftarrow$ SmoothOpt($\mathbf{u}^{(0)}, \frac{1}{2n^2\alpha}\mathbf{g}^{(0)}$)
    for $k \leftarrow 0$ to $T$
        do
            $\tau_k \leftarrow 2/(k+3)$
            $\mathbf{z}^{(k)} \leftarrow$ SmoothOpt($\mathbf{u}^{(0)}, \frac{1}{n^2\alpha}\mathbf{s}$)
            $\mathbf{u}^{(k+1)} \leftarrow \tau_k \mathbf{z}^{(k)} + (1-\tau_k)\mathbf{y}^{(k)}$.
            $(\mathbf{g}^{(k+1)}, \mathbf{X}^{(k+1)}) \leftarrow$ SmoothGrad($\mathbf{u}^{(k+1)}$)
            $\mathbf{s} \leftarrow \mathbf{s} + \frac{k+2}{2}\mathbf{g}^{(k+1)}$
            $\widehat{\mathbf{X}} \leftarrow \widehat{\mathbf{X}} + (k+2)\mathbf{X}^{(k+1)}$
            $\widehat{\mathbf{u}}^{(k+1)} \leftarrow$ SmoothOpt($\mathbf{z}^{(k)}, \frac{k+2}{2n^2\alpha}\mathbf{g}^{(k+1)}$)
            $\mathbf{y}^{(k+1)} \leftarrow \tau_k \widehat{\mathbf{u}}^{(k+1)} + (1-\tau_k)\mathbf{y}^{(k)}$

    $\bar{\mathbf{u}} \leftarrow \mathbf{y}^{(T)}$    $\bar{\mathbf{X}} \leftarrow \frac{2}{(T+1)(T+2)}\widehat{\mathbf{X}}$
    return $(\bar{\mathbf{X}}, \bar{\mathbf{u}})$
```

Figure 1: Procedure SmoothApprox($\mathbf{L}, \epsilon$)

where $\|\mathbf{u}\|_1 = \sum_{i=1}^n |u_i|$ denotes the $\mathcal{L}_1$-norm. We want to compute the iterates by minimizing the second-order bound in (23) and, to minimize the cost of each step, we would like to write the iterate in closed form. Since this is impossible to do when the distance between iterates is measured in terms of the $\mathcal{L}_1$-norm, we replace it by the relative entropy or the Kullback-Leibler (K-L) distance $d(\mathbf{u}, \mathbf{u}')$ defined as follows

$$d(\mathbf{u}, \mathbf{u}') = \sum_{i=1}^{n+1} u_i \log\left(\frac{u_i}{u'_i}\right) \tag{24}$$

where $u_{n+1} = 1 - \sum_{i=1}^n u_i$ and $u'_{n+1} = 1 - \sum_{i=1}^n u'_i$. In Appendix A we show that $d(\mathbf{u}, \mathbf{u}') \geq \frac{1}{2}\|\mathbf{u} - \mathbf{u}'\|_1^2$. Thus, we get the following bound

$$f_\alpha(\mathbf{u}) \leq f_\alpha(\mathbf{u}') + \nabla f_\alpha(\mathbf{u}')^T(\mathbf{u} - \mathbf{u}') + n^2\alpha d(\mathbf{u}, \mathbf{u}'). \tag{25}$$

By using the Lagrange multipliers, we can show (see Appendix A for details) that $\mathrm{argmin}_{\mathbf{u}\in\mathcal{U}}\left\{d(\mathbf{u}, \mathbf{u}') + \mathbf{g}^T\mathbf{u}\right\}$ is given by

$$u_i = \frac{u'_i e^{-g_i}}{u'_{n+1} + \sum_{k=1}^n u'_k e^{-g_k}}, \quad i = 1, \ldots, n, \tag{26}$$

where $u'_{n+1} = 1 - \sum_{i=1}^n u'_i$. We define a procedure SmoothOpt that given vectors $\mathbf{u}'$ and $\mathbf{g}$, returns the vector $\mathbf{u}$ defined by equation (26).

We now have all the ingredients necessary to describe the procedure SmoothApprox displayed in Figure 1. The algorithm wants to compute iterates that converge to the minimum of f_α over $\mathcal{U}$ by sequentially minimizing the approximate second-order Taylors' series expansion (25). Note that the bound (25) does not use the Hessian of f_α. This is because including the Hessian in the Taylors' series leads to a complicated optimization problem that cannot be solved in closed form. On the other hand, without the Hessian term, (25) is not likely to be a good estimate of the function f_α, at least when the iterates are far away from the minimum.

In SmoothApprox we compensate for the Hessian by using a technique proposed by Nesterov [24]. This technique computes the estimates $\mathbf{y}^{(k)}$ of the minimizer of f_α by keeping track of two sets of iterates: one set uses the gradient $\nabla f_\alpha(\mathbf{u}^{(k)})$ computed at the current iterate, and the other set uses a weighted combination of all the previous iterates $\nabla f_\alpha(\mathbf{u}^{(i)})$, $i = 0, \ldots, k$. From Line 5 of SmoothApprox, we have that at the beginning of iteration k of the variable $\mathbf{s} = \mathbf{s} + \left(\frac{k+1}{2}\right)\mathbf{g}^{(k+1)} = \sum_{i=1}^{k} \left(\frac{i+1}{2}\right)$. Therefore, the iterates $\mathbf{z}^{(k)}$ computed in Line 2 depend on a weighted combination of the gradients at all the previous iterates. The iterate $\widehat{\mathbf{u}}^{(k+1)}$ computed in Line 7 "corrects" the iterate $\mathbf{z}^{(k)}$ using the "local" gradient information $\mathbf{g}^{(k+1)}$ from the current $\mathbf{u}^{(k+1)}$. The new estimate $\mathbf{y}^{(k+1)}$ computed in Line 8 is a convex combination of the previous iterate $\mathbf{y}^{(k)}$ and the iterate $\widehat{\mathbf{u}}^{(k+1)}$.

Using results from [24] we derive the following complexity bound for SmoothApprox. See Appendix A for a proof.

Theorem 2 *For any $\epsilon > 0$, the output $(\bar{\mathbf{X}}, \bar{\mathbf{u}})$ of* SmoothApprox *is an ϵ-saddle-point. The running time is $O(\epsilon^{-1} Q(n) n \log n)$, where $Q(n)$ is the running time of* SmoothGrad.

Corollary 1 *Suppose* SmoothGrad *computes $\mathbf{X}_u$ in (22) via an eigendecomposition. Then the running time of* SmoothGrad *is $O(\epsilon^{-1} n^4 \log n)$.*

5 Semidefinite relaxation of graph coloring

The SDP relaxation for graph coloring corresponding to a graph $\mathcal{G} = (V, E)$ can be formulated as

$$\begin{array}{ll} \max & \zeta, \\ \text{s.t.} & x_{ii} \leq 1, \quad i = 1, \ldots, n, \\ & x_{ij} + \zeta \leq 0, \quad (i,j) \in E, \\ & \mathbf{X} \in \mathcal{X} \equiv \{\mathbf{X} : \mathbf{X} \succeq \mathbf{0}, \mathbf{Tr}(\mathbf{X}) \leq n\}. \end{array} \tag{27}$$

For k-colorable graphs ($k \geq 2$), $1 \geq \zeta^* \geq \frac{1}{k}$, hence, an ϵ-optimal solution has a relative error of $k\epsilon$ [16, 17]. Since, without loss of generality, $x^*_{ii} = 1$ in any optimal solution of (27), the second constraint in (27) can be reformulated as $\frac{1}{2}(x_{ii} + x_{jj}) + x_{ij} + \zeta \leq 1$. With this reformulation, the problem (27) reduces to an SDP with packing constraints. The dual of this SDP is given by

$$\begin{array}{ll} \min & \sum_{i=1}^{n} u_i \\ \text{s.t.} & \sum_{j=1}^{m} v_j = 1, \\ & \mathbf{diag}(\mathbf{u}) + \mathbf{A}\mathbf{v} \succeq \mathbf{0}, \\ & \mathbf{u}, \mathbf{v} \geq \mathbf{0}, \end{array} \tag{28}$$

where $m = |E|$ denotes the number of edges in the graph, $\mathbf{A} : \mathbf{R}^m \mapsto \mathbf{R}^{n \times n}$ is the linear operator $\mathbf{A}\mathbf{v} = \frac{1}{2}\sum_{(i,j)\in E} v_{(i,j)} \mathbf{E}_{ij}$, and $\mathbf{E}_{ij} \in \mathbf{R}^{n \times n}$ with 1's in the (i,j) and (j,i) position and zeros everywhere else.

As in the case with Maxcut, we dualize both the constraints $\mathbf{diag}(\mathbf{X}) \leq \mathbf{1}$ and $\max_{(i,j)\in E}\{x_{ij}\} + \zeta \leq 0$ to define a saddle-point problem:

$$\max_{\mathbf{X}\in\mathcal{X}} \min_{(\mathbf{u},\mathbf{v})\in\mathcal{U}\times\mathcal{V}} \phi(\mathbf{X}, \mathbf{u}, \mathbf{v}),$$

where

$$\phi(\mathbf{X}, \mathbf{u}, \mathbf{v}) = n \sum_{i=1}^{m} u_i - n(n\mathbf{u} + \mathbf{A}\mathbf{v}) \bullet \mathbf{X}, \tag{29}$$

$$\begin{array}{rcl} \mathcal{X} & = & \{\mathbf{X} : \mathbf{X} \succeq \mathbf{0}, \mathbf{Tr}(\mathbf{X}) \leq 1\}, \\ \mathcal{U} & = & \{\mathbf{u} : \mathbf{u} \geq 0, \sum_{i=1}^{n} u_i \leq 1\}, \\ \mathcal{V} & = & \left\{\mathbf{v} : \mathbf{v} \geq \mathbf{0}, \sum_{j=1}^{m} v_j = 1\right\}. \end{array} \tag{30}$$

Theorem 3 *Fix $\epsilon > 0$. Suppose $(\bar{\mathbf{X}}, (\bar{\mathbf{u}}, \bar{\mathbf{v}})) \in \mathcal{X} \times (\mathcal{U} \times \mathcal{V})$ is an ϵ-saddle-point. Let $\bar{\zeta} = -n \max_{(i,j)\in E}\{\bar{x}_{ij}\}$, $\bar{d} = n \max_{i=1,\ldots,n}\{\bar{x}_{ii}\}$,*

$$\widehat{\mathbf{u}} = n\bar{\mathbf{u}} - \big(\min\{\lambda_{\min}(\mathbf{diag}(\bar{\mathbf{u}}) + \mathbf{A}\bar{\mathbf{v}}), 0\}\big)\mathbf{1}, \qquad \widehat{\mathbf{v}} = \bar{\mathbf{v}}, \tag{31}$$

and

$$\begin{aligned} \widehat{x}_{ii} &= \begin{cases} n\bar{x}_{ii}, & \bar{d} \le 1,\\ \frac{n\bar{x}_{ii}}{\bar{d}}, & \bar{d} > 1,\end{cases}\\ \widehat{x}_{ij} &= \begin{cases} n\bar{x}_{ij}, & \bar{d} \le 1, \bar{\zeta} > 0,\\ \frac{n\bar{x}_{ij}}{\bar{d}}, & \bar{d} > 1, \bar{\zeta} > 0,\\ 0, & \bar{\zeta} \le 0.\end{cases}\\ \widehat{\zeta} &= -\max_{(i,j)\in E}\{\widehat{x}_{ij}\}.\end{aligned} \tag{32}$$

Then $(\widehat{\zeta}, \widehat{\mathbf{X}})$ *and* $(\widehat{\mathbf{u}}, \widehat{\mathbf{v}})$ *are an* ϵ*-optimal primal-dual pair for (27) and (28).*

Proof: One can check that $(\widehat{\mathbf{u}}, \widehat{\mathbf{v}})$ defined in (31) is feasible for the dual SDP (28). From the definition of $\mathcal{X}$, it follows that

$$\max_{\mathbf{X}\in\mathcal{X}} \phi(X, \bar{\mathbf{u}}, \bar{\mathbf{v}}) = n\sum_{i=1}^{m} \bar{u}_i - n\big(\min\{\lambda_{\min}(\mathbf{diag}(\bar{\mathbf{u}}) + n\mathbf{A}\bar{\mathbf{v}}), 0\}\big) = \sum_{i=1}^{n} \widehat{u}_i. \tag{33}$$

The pair $(\widehat{\zeta}, \widehat{\mathbf{X}})$ defined in (32) is feasible for the primal SDP (27). Next, we show that the objective value $\widehat{\zeta}$ is lower bounded by

$$\min_{(\mathbf{u},\mathbf{v})\in\mathcal{U}\times\mathcal{V}} \phi(\bar{\mathbf{X}}, \mathbf{u}, \mathbf{v}) = \bar{\zeta} - n\max\{\bar{d} - 1, 0\}. \tag{34}$$

(a) $\bar{d} \le 1, \bar{\zeta} > 0$: In this case $(\widehat{\zeta}, \widehat{\mathbf{X}}) = (\bar{\zeta}, \bar{\mathbf{X}})$, and $\widehat{\zeta} = \bar{\zeta} = \bar{\zeta} - n\max\{\bar{d} - 1, 0\}$.

(b) $\bar{d} \ge 1, \bar{\zeta} > 0$: In this case, $\widehat{\zeta} = \frac{\bar{\zeta}}{\bar{d}} \ge \bar{\zeta}\big(1 - (\bar{d} - 1)\big) \ge \bar{\zeta} - n(\bar{d} - 1)$, where the first inequality follows from the fact that $\frac{1}{d} \ge 1 - (d - 1)$ for all $d > 0$, and the second inequality follows from the bound $|\bar{x}_{ij}| \le \sqrt{\bar{x}_{ii}\bar{x}_{jj}} \le 1$.

(c) $\bar{\zeta} \le 0$. In this case, $\widehat{\zeta} = 0 \ge \bar{\zeta} + n\max\{\bar{d} - 1, 0\}$.

Thus, we have established the existence of an ϵ-optimal primal-dual pair $(\widehat{\lambda}, \widehat{\mathbf{X}})$ and $(\widehat{u}, \widehat{v})$. ■

As in Section 3 we work with the function $f(\mathbf{u}, \mathbf{v}) = \max_{\mathbf{X}\in\mathcal{X}} \phi(\mathbf{X}, \mathbf{u}, \mathbf{v})$, and smooth it to obtain

$$f_\alpha(\mathbf{u}, \mathbf{v}) = n\sum_{i=1}^{n} u_i - \frac{1}{\alpha}\log\Big(1 + \sum_{i=1}^{n} e^{-n\alpha\lambda_i}\Big),$$

where $\lambda_i = \lambda_i(n\,\mathbf{diag}(\mathbf{u}) + \mathbf{A}\mathbf{v}), i = 1, \ldots, n$. As before,

$$f(\mathbf{u}, \mathbf{v}) \le f_\alpha(\mathbf{u}, \mathbf{v}) \le f(\mathbf{u}, \mathbf{v}) + \frac{\log(n+1)}{\alpha},$$

and

$$\begin{bmatrix} \nabla_u f_\alpha(\mathbf{u}, \mathbf{v}) \\ \nabla_v f_\alpha(\mathbf{u}, \mathbf{v}) \end{bmatrix} = \begin{bmatrix} n^2\,\mathbf{diag}(\mathbf{X}_u) - n\mathbf{1} \\ \mathbf{A}^T\mathbf{X}_u \end{bmatrix}, \qquad \mathbf{X}_u = \frac{e^{-n\alpha(n\,\mathbf{diag}(\mathbf{u}) + \mathbf{A}\mathbf{v})}}{\big(1 + \sum_{i=1}^{n} e^{-n\alpha\lambda_i}\big)}. \tag{35}$$

The function f_α can be optimized using a procedure very similar to SmoothApprox except that each iteration we solve two optimization problems: one in $\mathbf{u}$ and the other in $\mathbf{v}$. The details of the procedure SmoothApproxColoring are in Appendix B.

Theorem 4 *For any* $\epsilon > 0$*, the output* $(\bar{\mathbf{X}}, \bar{\mathbf{u}}, \bar{\mathbf{v}})$ *of* SmoothApproxColoring *is an* ϵ*-saddle-point. The running time is* $O(\epsilon^{-1}Q(n)n\sqrt{\log(n)\log(m)})$*, where* $Q(n)$ *is the running time of* SmoothGrad.

The $\sqrt{\log(m)}$ factor appears because the number of constraints in the graph coloring problem is m as opposed to n in the case of the Maxcut problem. Again the bottleneck step is the computation of the smoothed gradient $\mathbf{X}_u$.

Corollary 2 *If* $\mathbf{X}_u$ *computed via eigendecomposition then the running time of* SmoothApproxColoring *is* $O\left(\epsilon^{-1}n^4\log n\right)$.

6 The Lovász Shannon-capacity problem

The Lovász Shannon-capacity problem (see [21], [14]) on a graph with edge set E can be formulated as the SDP:

$$\begin{array}{ll} \max & \sum_{i,j} x_{ij}, \\ \text{s.t.} & \mathbf{Tr}(\mathbf{X}) + x_{ij} \leq 1, \quad (i,j) \in E \\ & \mathbf{Tr}(\mathbf{X}) - x_{ij} \leq 1, \quad (ij) \in E \\ & \mathbf{X} \in \{\mathbf{X} \succeq 0 : \mathbf{Tr}(\mathbf{X}) = 1\}, \end{array} \tag{36}$$

This is an SDP with packing constraints. Our method can be used to solve this problem, which, for space reasons we do not include in this extended abstract.

7 Computing the matrix exponential

The most expensive step in SmoothApprox and SmoothApproxColoring is SmoothGrad that computes

$$\mathbf{X_u} = \frac{\mathbf{V}\,\mathbf{diag}(e^{n\alpha\boldsymbol{\lambda}})\mathbf{V}^T}{\left(1 + \sum_{i=1}^n e^{n\alpha\lambda_i}\right)} = \frac{e^{n\alpha\mathbf{A}}}{\mathbf{Tr}(e^{n\alpha\mathbf{A}})},$$

where $e^{n\mathbf{A}}$ denotes the *matrix exponential* [22, 23] of

$$\mathbf{A} = \begin{bmatrix} 0 & \mathbf{0}^T \\ \mathbf{0} & \mathbf{L} - 5n\,\mathbf{diag}(\mathbf{u}). \end{bmatrix}$$

The direct method for computing $e^{n\alpha\mathbf{A}}$ involves computing an eigendecomposition of the matrix $\mathbf{A}$ and then using this to compute $\mathbf{X}_u$ via (22). In this section we discuss a method that computes an ϵ-approximation for $e^{n\alpha\mathbf{A}}$ without first computing the eigendecomposition.

We use a method which we refer to as SI-Lanczos, or the *shift-and-invert* Lanczos, developed by van den Eshof and Hochbruck [30]. The main improvement in [30] results from using Krylov subspaces generated by $(\mathbf{I} + \gamma\mathbf{A})^{-1}$, where $\gamma > 0$ (see also [15]). Therefore, at each iteration, we need to compute a solution to the linear system $\mathbf{y}_{k+1} = (\mathbf{I} + \gamma\mathbf{A})\mathbf{y}_k$. SI-Lanczos works best when $\mathbf{A} \succ 0$, so we approximate the exponential of $\bar{\mathbf{A}} = \mathbf{A} + (6n-2)\mathbf{I}$ (resp. $\bar{\mathbf{A}} = \mathbf{A} + 2n\mathbf{I}$) for the Maxcut (resp. coloring) SDP. This shift ensures that $\bar{\mathbf{A}} \succ 0$ and the condition number $\kappa(\bar{\mathbf{A}}) = \lambda_{\max}(\bar{\mathbf{A}})/\lambda_{\min}(\bar{\mathbf{A}}) \leq 2$. Note that this shift does *not* change the value of $\mathbf{X_u}$ defined in (7). Since the condition number, $\kappa(\bar{\mathbf{A}})$, is bounded, SI-Lanczos can use the well-known conjugate gradient method (see, e.g., Golub and Van Loan [11]) to solve the system of linear equations required at each iteration. The conjugate gradient method computes solutions to linear systems that, at iteration k, have residual error bounded by $[(\sqrt{\kappa(\mathbf{A})} - 1)/(\sqrt{\kappa(\mathbf{A})} + 1)]^k$, which implies the convergence is geometric. Thus, this solves systems of linear equations in $O((n+m)\log(1/\epsilon))$ iterations where $\epsilon > 0$ is the relative error and m is the number of nonzeros in the matrix $\mathbf{A}$. Overall, Theorem 3.3 of [30] indicates that $O(\log^2(1/\epsilon))$ iterations are required to approximate each column of the exponential. This results an overall complexity of $O(n(n+m)\log^3(1/\epsilon))$. Thus, we have the following corollary.

Corollary 3 *The complexity of computing* $\mathbf{X}_u$ *via* SI-Lanczos *is* $O\big(n(n+m)\log^3(\frac{1}{\epsilon})\big)$. *Therefore, using* SI-Lanczos *for* SmoothGrad *in* SmoothApprox *and* SmoothApproxColoring *results in a complexity of* $O\Big(\epsilon^{-1}n^2(n+m)\log(n)\log^3(\frac{1}{\epsilon})\Big)$.

References

[1] F. Alizadeh. Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM J. Optim.*, 5(1):13–51, 1995.

[2] S. Arora, S. Rao, and U. Vazirani. Expander flows, geometric embeddings, and graph partitionings. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing*, pages 222–231, 2004.

[3] S. J. Benson, Yinyu Ye, and X. Zhang. Solving large-scale sparse semidefinite programs for combinatorial optimization. *SIAM J. Optim.*, 10(2):443–461 (electronic), 2000.

[4] D. Bienstock and G. Iyengar. Solving fractional packing problems in $O^*(\frac{1}{\epsilon})$ iterations. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing*, pages 146–155, 2004.

[5] Daniel Bienstock. *Potential function methods for approximately solving linear programming problems: theory and practice*. International Series in Operations Research & Management Science, 53. Kluwer Academic Publishers, Boston, MA, 2002.

[6] S. Burer and R. D. C. Monteiro. A projected gradient algorithm for solving the maxcut SDP relaxation. *Optim. Methods Softw.*, 15(3-4):175–200, 2001.

[7] L. Fleischer. Fast approximation algorithms for fractional covering problems with box constraint. In *Proceedings of the 15th ACM-SIAM Symposium on Discrete Algorithms*, 2004.

[8] Lisa K. Fleischer. Approximating fractional multicommodity flow independent of the number of commodities. *SIAM J. Discrete Math.*, 13(4):505–520 (electronic), 2000.

[9] Naveen Garg and Jochen Konemann. Faster and simpler algorithms for multicommodity flow and other fractional packing problems. In *Proceedings of the 39th Annual Symposium on Foundations of Computer Science*, pages 300–309, 1998.

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

[11] Gene H. Golub and Charles F. Van Loan. *Matrix computations*, volume 3 of *Johns Hopkins Series in the Mathematical Sciences*. Johns Hopkins University Press, Baltimore, MD, 1983.

[12] M. D. Grigoriadis and L. G. Khachiyan. Fast approximation schemes for convex programs with many blocks and coupling constraints. *SIAM Journal on Optimization*, 4(1):86–107, February 1994.

[13] M. D. Grigoriadis and L. G. Khachiyan. An exponential-function reduction method for block angular convex programs. *Networks*, 26:59–68, 1995.

[14] M. Grötschel, L. Lovász, and A. Schrijver. Polynomial algorithms for perfect graphs. In *Topics on perfect graphs*, volume 88 of *North-Holland Math. Stud.*, pages 325–356. North-Holland, Amsterdam, 1984.

[15] Marlis Hochbruck and Christian Lubich. On Krylov subspace approximations to the matrix exponential operator. *SIAM J. Numer. Anal.*, 34(5):1911–1925, 1997.

[16] D. Karger, R. Motwani, and M. Sudan. Approximate graph coloring by semidefinite programming. *J. ACM*, 45(2):246–265, 1998.

[17] P. Klein and H-I Lu. Efficient approximation algorithms for semidefinite programs arising from MAX CUT and COLORING. In *Proceedings of the Twenty-eighth Annual ACM Symposium on the Theory of Computing (Philadelphia, PA, 1996)*, pages 338–347, New York, 1996. ACM.

[18] P. Klein and H-I Lu. Space-efficient approximation algorithms for MAXCUT and COLORING semidefinite programs. In *Algorithms and computation (Taejon, 1998)*, volume 1533 of *Lecture Notes in Comput. Sci.*, pages 387–396. Springer, Berlin, 1998.

[19] P. Klein, S. A. Plotkin, C. Stein, and É. Tardos. Faster approximation algorithms for the unit capacity concurrent flow problem with applications to routing and finding sparse cuts. *SIAM Journal on Computing*, 23(3):466–487, June 1994.

[20] T. Leighton, F. Makedon, S. Plotkin, C. Stein, É. Tardos, and S. Tragoudas. Fast approximation algorithms for multicommodity flow problems. *Journal of Computer and System Sciences*, 50:228–243, 1995.

[21] László Lovász. On the Shannon capacity of a graph. *IEEE Trans. Inform. Theory*, 25(1):1–7, 1979.

[22] C. Moler and C. Van Loan. Nineteen dubious ways to compute the exponential of a matrix. *SIAM Rev.*, 20(4):801–836, 1978.

[23] C. Moler and C. Van Loan. Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM Rev.*, 45(1):3–49 (electronic), 2003.

[24] Yu. Nesterov. Smooth minimization of nonsmooth functions. Technical report, CORE DP, 2003.

[25] Yurii Nesterov and Arkadii Nemirovskii. *Interior-point polynomial algorithms in convex programming*, volume 13 of *SIAM Studies in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1994.

[26] S. Plotkin, D. B. Shmoys, and E. Tardos. Fast approximation algorithms for fractional packing and covering problems. *Mathematics of Operations Research*, 20:257–301, 1995.

[27] T. Radzik. Fast deterministic approximation for the multicommodity flow problem. In *Proceedings of the 6th ACM-SIAM Symposium on Discrete Algorithms*, pages 486–496, 1995.

[28] F. Shahrokhi and D. W. Matula. The maximum concurrent flow problem. *Journal of the ACM*, 37:318 – 334, 1990.

[29] M. Skutella. Convex quadratic and semidefinite programming relaxations in scheduling,. *Journal of the ACM*, 48(2):206–242, 2001.

[30] J. van den Eshof and M. Hochbruck. Preconditioning lanczos approximations to the matrix exponential, 2004.

A Details of SmoothApprox

We begin with the solution of the optimization problem

$$\begin{array}{ll} \min & d(\mathbf{u}, \mathbf{u}') + \mathbf{g}^T \mathbf{u}, \\ \text{s. t.} & \sum_{i=1}^{n+1} u_i = 1, \\ & u \geq \mathbf{0}, \end{array} \tag{37}$$

where $u'_{n+1} = 1 - \sum_{i=1}^{n} u_i$. Using Lagrange multipliers, the optimal solution $(\mathbf{u}^*, s^*)$ can be characterized as follows:

$$\begin{array}{rcl} u_i^* & = & u'_i e^{-g_i} e^{\beta + \rho_i}, \quad i = 1, \ldots, n, \\ u_{n+1}^* & = & u'_{n+1} e^{\beta + \rho_{n+1}}, \end{array}$$

where $\boldsymbol{\rho} \geq \mathbf{0}$ and satisfies the complementary slackness condition $\rho_i u_i^* = 0$, $i = 1, \ldots, n+1$. Since $\mathbf{u}' \geq \mathbf{0}$ and $u'_{n+1} \geq 0$, it follows that $\boldsymbol{\rho} = \mathbf{0}$. Since $\sum_{i=1}^{n+1} u_i^* = 1$, we have that

$$u_i^* = \frac{u'_i e^{-g_i}}{u'_{n+1} + u'_i e^{-g_i}}, \quad i = 1, \ldots, n.$$

The next step is a new characterization for the smoothed function $f_\alpha(\mathbf{u})$.

$$f_\alpha(\mathbf{u}) = -n \sum_{i=1}^{n} u_i + \begin{array}{ll} \max & n\mathbf{A} \bullet \mathbf{X} - \frac{1}{\alpha} \Big(\sum_{i=1}^{n} \mu_i(\mathbf{X}) \log(\mu_i(\mathbf{X})) + s \log(s) \Big), \\ \text{s.t.} & \mathbf{Tr}(\mathbf{X}) + s = 1, \\ & \mathbf{X} \succeq \mathbf{0}, s \geq 0, \end{array} \tag{38}$$

where $\mathbf{A} = \mathbf{L} - 5n\, \mathbf{diag}(\mathbf{u})$, and $\{\mu_i(\mathbf{X}) : 1 \leq i \leq n\}$ denote the eigenvalues of $\mathbf{X}$. This result follows from the following observations.

(i) Let $\mathbf{A} = \mathbf{V} \boldsymbol{\Lambda} \mathbf{V}^T$ with $\{\lambda_i : 1 \leq i \leq n\}$ arranged in decreasing order. Fix $\boldsymbol{\mu} \geq \mathbf{0}$ and $s \geq 0$ such that $\mu_1 \geq \mu_2 \geq \ldots \geq \mu_n$ and $\sum_{i=1}^{n} \mu_i + s = 1$. Let $\mathbf{X} = \mathbf{W}\, \mathbf{diag}(\boldsymbol{\mu}) \mathbf{W}^T$ for some orthonormal matrix $\mathbf{W}$. Then

$$n\mathbf{A} \bullet \mathbf{X} = n \sum_{i=1}^{n} \mu_i (\mathbf{w}_i^T \mathbf{A} \mathbf{w}_i) \leq n \sum_{i=1}^{n} \mu_i \lambda_i, \tag{39}$$

with equality if and only if each $\mathbf{w}_i$ is the eigenvector of $\mathbf{A}$ corresponding to λ_i. Thus, for a fixed $\boldsymbol{\mu}$, the optimal choice $\mathbf{W}$ is $\mathbf{V}$, the set of eigenvectors of $\mathbf{A}$.

(ii) Thus, we have that

$$f_\alpha(\mathbf{u}) = -n \sum_{i=1}^{m} u_i + \begin{array}{ll} \max & n \sum_{i=1}^{n} \mu_i \lambda_i - \frac{1}{\alpha} \Big(\sum_{i=1}^{n} \mu_i \log(\mu_i) + s \log(s) \Big), \\ \text{s.t.} & \sum_{i=1}^{n} \mu_i + s = 1, \\ & \mu \geq \mathbf{0}, s \geq 0, \end{array}$$

This is an optimization problem of the form (37). Therefore, the optimal $\boldsymbol{\mu}^*$ is given by

$$\mu_i^* = \frac{e^{n\alpha\lambda_i}}{1 + \sum_{k=1}^{n} e^{n\alpha\lambda_k}}, \quad i = 1, \ldots, n,$$

and the optimal value $\mathbf{X}_u$ is given by

$$\mathbf{X}_u = \frac{\mathbf{diag}\left(\mathbf{V}\, \mathbf{diag}(e^{n\alpha\boldsymbol{\lambda}}) \mathbf{V}^T\right)}{\left(1 + \sum_{i=1}^{n} e^{n\alpha\lambda_i}\right)},$$

¿From the new formulation for $f_\alpha(\mathbf{u})$, it follows that $\nabla f_\alpha(\mathbf{u}) = 5n(\mathbf{1} - n\,\mathbf{diag}(\mathbf{X}_u))$.

The next step is to establish the approximate second-order Taylor series expansion

$$f_\alpha(\mathbf{u}) \leq f_\alpha(\mathbf{u}') + \nabla f_\alpha(\mathbf{u}')^T(\mathbf{u} - \mathbf{u}') + \frac{n^2\alpha}{2} \left\|\mathbf{u} - \mathbf{u}'\right\|_1^2,$$

where $\left\|\cdot\right\|_1$ denotes the $\mathcal{L}_1$-norm. This result essentially follows from the Cauchy-Schwartz inequality. See [24] or [4] for a proof.

Let $H(\mathbf{u}) = \sum_{i=1}^{n+1} u_i \log(u_i)$, where $u_{n+1} = 1 - \sum_{i=1}^n u_i$. Then H is a convex function of $\mathbf{u}$. Also, for any vector $\mathbf{w} \in \mathbf{R}^n$

$$\begin{aligned} \mathbf{w}^T \nabla^2 H(\mathbf{u})\mathbf{w} &= \sum_{i=1}^n \frac{w_i^2}{u_i}, \\ &= \Big(\sum_{i=1}^n \frac{w_i^2}{u_i}\Big)\Big(\sum_{i=1}^n u_i\Big), \\ &\geq (\sum_{i=1}^n \frac{|w_i|}{\sqrt{u_i}}\sqrt{u_i})^2, \qquad (40) \\ &= \left\|\mathbf{w}\right\|_1^2, \qquad (41) \end{aligned}$$

where (40) follows from the Cauchy-Schwartz inequality $|\mathbf{w}^T\mathbf{v}|^2 \leq \|\mathbf{w}\|\,\|\mathbf{v}\|$. From the second-order Taylor series expansion we have that for some $\theta \in [0, 1]$

$$\begin{aligned} H(\mathbf{u}) - H(\mathbf{u}') &= \nabla H(\mathbf{u}')^T(\mathbf{u} - \mathbf{u}') + \frac{1}{2}(\mathbf{u} - \mathbf{u}')^T \nabla^2 H(\theta\mathbf{u} + (1-\theta)\mathbf{u}')(\mathbf{u} - \mathbf{u}'), \\ &\geq \nabla H(\mathbf{u}')^T(\mathbf{u} - \mathbf{u}') + \frac{1}{2}\left\|\mathbf{u} - \mathbf{u}'\right\|^2. \end{aligned}$$

Simple algebra shows that the entropy distance

$$d(\mathbf{u}, \mathbf{u}') = H(\mathbf{u}) - H(\mathbf{u}') - \nabla H(\mathbf{u}')^T(\mathbf{u} - \mathbf{u}') \geq \frac{1}{2}\left\|\mathbf{u} - \mathbf{u}'\right\|^2. \qquad (42)$$

We now have all the ingredients to prove the complexity bound for SmoothApprox. The result of the proof closely follows [24].

Lemma 1 *Fix $\alpha = \frac{2\log(n+1)}{2}$. Suppose that for all $k \geq 0$,*

$$(\mathcal{R}_k) \qquad \Gamma_k f_\alpha(\mathbf{y}^{(k)}) \leq \psi^{(k)} := \min_{\mathbf{u}\in\mathcal{U}} \left\{ n^2\alpha d(\mathbf{u}, \mathbf{u}^{(0)}) + \sum_{i=0}^k \gamma_i [f_\alpha(\mathbf{u}^{(i)}) + \nabla f_\alpha(\boldsymbol{\delta}_i)^T(\mathbf{u} - \mathbf{u}^{(i)})] \right\}. \qquad (43)$$

where $\gamma_k = \frac{k+1}{2}$, and $\Gamma_k = \sum_{i=0}^k \gamma_i = \frac{(k+1)(k+2)}{4}$. Then for all $T > \frac{4n\log(n+1)}{\epsilon}$ we have $f(\mathbf{y}^{(T)}) \leq f^* + \epsilon$.

Proof: Let $\mathbf{u}^* = \operatorname{argmin}\{f_\alpha(\mathbf{u})\}$. From the convexity of f_α we have that

$$f_\alpha(\mathbf{u}^{(i)}) + \nabla f_\alpha(\mathbf{u}^{(i)})^T(\mathbf{u}^* - \mathbf{u}^{(i)}) \leq f_\alpha(\mathbf{u}^*).$$

Thus, $(\mathcal{R}_k)$ implies that

$$\begin{aligned} f_\alpha(\mathbf{y}^{(k)}) &\leq \frac{n^2\alpha \max_{\mathbf{u}\in\mathcal{U}}\{d(\mathbf{u}, \mathbf{u}^{(0)})\}}{\Gamma_k} + \frac{1}{\Gamma_k}\sum_{i=1}^k \gamma_i[f_\alpha(\mathbf{u}^{(i)}) + \nabla f_\alpha(\boldsymbol{\delta}_i)^T(\mathbf{u}^* - \mathbf{u}^{(i)})], \\ &\leq \frac{n^2\alpha\log(n)}{\Gamma_k} + f_\alpha^*, \qquad (44) \end{aligned}$$

where (44) follows from that that $\max_{\mathbf{u}\in\mathcal{U}}\{d(\mathbf{u},\mathbf{u}^{(0)})\}$ is achieved at one of the extreme point of $\mathcal{U}$. From the bounds $f(\mathbf{u}) \leq f_\alpha(\mathbf{u}) \leq f(\mathbf{u}) + \frac{\ln(n+1)}{\alpha}$, we have that

$$f(\mathbf{y}^{(k)}) \leq f_\alpha(\mathbf{y}^{(k)}) \leq \frac{n^2\alpha\log(n)}{\Gamma_k} + f_\alpha^* \leq \frac{n^2\alpha\log(n)}{\Gamma_k} + \frac{\log(n+1)}{\alpha} + f^*.$$

The result now follows by substituting $\alpha = \frac{2\log(n+1)}{\epsilon}$. ■

All that remains to show is that $(\mathcal{R}_k)$ holds for all $k \geq 0$. The base case $(\mathcal{R}_0)$ is established as follows. Since $\gamma_0 = \frac{1}{2} < 1$,

$$\begin{aligned}
&\min_{\mathbf{u}\in\mathcal{U}}\Big\{n^2\alpha d(\mathbf{u},\mathbf{u}^{(0)}) + \gamma_0[f_\alpha(\mathbf{u}^{(0)}) + \nabla f_\alpha(\mathbf{u}^{(0)})^T(\mathbf{u}-\mathbf{u}^{(0)})]\Big\}\\
&\quad\geq \gamma_0 \min_{\mathbf{u}\in\mathcal{U}}\left\{\frac{n^2\alpha}{2}\left\|\mathbf{u}-\mathbf{u}^{(0)}\right\|_1^2 + f_\alpha(\mathbf{u}^{(0)}) + \nabla f_\alpha(\mathbf{u}^{(0)})^T(\mathbf{u}-\mathbf{u}^{(0)})]\right\}, &(45)\\
&\quad\geq \gamma_0\left(\frac{n^2\alpha}{2}\left\|\mathbf{y}^{(0)}-\mathbf{u}^{(0)}\right\|_1^2 + f_\alpha(\mathbf{u}^{(0)}) + \nabla f_\alpha(\mathbf{u}^{(0)})^T(\mathbf{y}^{(0)}-\mathbf{u}^{(0)})]\right),\\
&\quad\geq \gamma_0 f_\alpha(\mathbf{y}^{(0)}), &(46)
\end{aligned}$$

where (45) follows from (42) and (46) follows from the approximate Taylor's series (23).

Assume $(\mathcal{R}_k)$ is true. At the beginning of iteration k, the cumulative gradient $\mathbf{s}$ computed in Line 5 of SmoothApprox equals

$$\mathbf{s} = \mathbf{s} + \Big(\frac{k+1}{2}\Big)\mathbf{g}^{(k)} = \sum_{i=0}^{k}\gamma_i\mathbf{g}^{(i)}$$

Therefore, it follows that

$$\begin{aligned}
\mathbf{z}^{(k)} &= \textsc{SmoothOpt}\big(\mathbf{u}^{(0)}, 1/(n^2\alpha)\mathbf{s}\big),\\
&= \operatorname*{argmin}_{\mathbf{u}\in\mathcal{U}}\Big\{d(\mathbf{u},\mathbf{u}^{(0)}) + (n^2\alpha)^{-1}\mathbf{s}^T\mathbf{u}\Big\},\\
&= \operatorname*{argmin}_{\mathbf{u}\in\mathcal{U}}\Big\{n^2\alpha d(\mathbf{u},\mathbf{u}^{(0)}) + \mathbf{s}^T\mathbf{u}\Big\},\\
&= \operatorname*{argmin}_{\mathbf{u}\in\mathcal{U}}\Big\{n^2\alpha d(\mathbf{u},\mathbf{u}^{(0)}) + \sum_{i=0}^{k}\gamma_i[f_\alpha(\mathbf{u}^{(i)}) + \nabla f_\alpha(\mathbf{u}^{(i)})^T(\mathbf{u}-\mathbf{u}^{(i)})]\Big\}.
\end{aligned}$$

Thus, we have that

$$\psi^{(k)} = f_\alpha(\mathbf{z}^{(k)}), \tag{47}$$

and

$$\Big(\nabla_u d(\mathbf{z}^{(k)},\mathbf{u}^{(0)}) + \sum_{i=1}^{k}\gamma_i\nabla f_\alpha(\mathbf{u}^{(i)})\Big)^T(\mathbf{u}-\mathbf{z}^{(k)}) \geq 0, \tag{48}$$

where $\nabla_u d(\mathbf{u},\mathbf{u}^{(0)})$ denotes the gradient with respect to the first argument. Since $\mathbf{u}^{(0)} = \frac{1}{n+1}\mathbf{1}$, it follows that

$$d(\mathbf{u},\mathbf{u}^{(0)}) = d(\mathbf{u},\mathbf{z}^{(k)}) + d(\mathbf{z}^{(k)},\mathbf{u}^{(0)}) + \nabla_u d(\mathbf{z}^{(k)},\mathbf{u}^{(0)}).$$

Thus, we have that

$$\begin{aligned}
& n^2\alpha d(\mathbf{u},\mathbf{u}^{(0)}) + \sum_{i=0}^{k} \gamma_i[f_\alpha(\mathbf{u}^{(i)}) + \nabla f_\alpha(\boldsymbol{\delta}_i)^T(\mathbf{u}-\mathbf{u}^{(i)})] \\
= {} & n^2\alpha d(\mathbf{u},\mathbf{z}^{(k)}) + \underbrace{\Big(\nabla_u d(\mathbf{z}^{(k)},\mathbf{u}^{(0)}) + \sum_{i=1}^{k} \gamma_i \nabla f_\alpha(\boldsymbol{\delta}_i)\Big)^T(\mathbf{u}-\mathbf{z}^{(k)})}_{\geq 0} \\
& + \underbrace{\Big(d(\mathbf{z}^{(k)},\mathbf{u}^{(0)}) + \sum_{i=0}^{k} \gamma_i[f_\alpha(\mathbf{u}^{(i)}) + \nabla f_\alpha(\boldsymbol{\delta}_i)^T(\mathbf{z}^{(k)}-\mathbf{u}^{(i)})]\Big)}_{=\psi^{(k)}}, \\
\geq {} & \Gamma_k f_\alpha(\mathbf{y}^{(k)}) + n^2\alpha d(\mathbf{u},\mathbf{z}^{(k)}), \qquad (49)
\end{aligned}$$

where the last inequality follows from the induction hypothesis.

Thus, we have that

$$\begin{aligned}
& \psi^{(k+1)} \\
& \geq \min_{\mathbf{u}\in\mathcal{U}} \Big\{ n^2\alpha d(\mathbf{u},\mathbf{z}^{(k)}) + \Gamma_k f_\alpha(\mathbf{y}^{(k)}) + \gamma_{k+1}[f_\alpha(\mathbf{u}^{(k+1)}) + \nabla f_\alpha(\mathbf{u}^{(k+1)})^T(\mathbf{u}-\mathbf{u}^{(k+1)})] \Big\}. \qquad (50)
\end{aligned}$$

¿From the convexity of f_α and the rule in Line 3 of SmoothApprox we have that

$$\begin{aligned}
& \Gamma_k f_\alpha(\mathbf{y}^{(k)}) + \gamma_{k+1}[f_\alpha(\mathbf{u}^{(k+1)}) + \nabla f_\alpha(\mathbf{u}^{(k+1)})^T(\mathbf{u}-\mathbf{u}^{(k+1)})] \\
& \quad \geq \Gamma_k[f_\alpha(\mathbf{u}^{(k+1)}) + \nabla f_\alpha(\mathbf{u}^{(k+1)})^T(\mathbf{y}^{(k)}-\mathbf{u}^{(k+1)})] \\
& \qquad + \gamma_{k+1}[f_\alpha(\mathbf{u}^{(k+1)}) + \nabla f_\alpha(\mathbf{u}^{(k+1)})^T(\mathbf{u}-\mathbf{u}^{(k+1)})], \\
& \quad = \Gamma_{k+1} f_\alpha(\mathbf{u}^{(k+1)}) + \gamma_{k+1}\nabla f_\alpha(\mathbf{u}^{(k+1)})^T(\mathbf{u}-\mathbf{z}^{(k)}). \qquad (51)
\end{aligned}$$

¿From Line 7, we get

$$\begin{aligned}
\widehat{\mathbf{u}}^{(k+1)} &= \textsc{SmoothOpt}\big(\mathbf{z}^{(k)}, (k+2)/(2n^2\alpha)\mathbf{g}^{(k+1)}\big), \\
&= \operatorname*{argmin}_{\mathbf{u}\in\mathcal{U}} \Big\{ n^2\alpha d(\mathbf{u},\mathbf{z}^{(k)}) + \gamma_{k+1}\mathbf{g}^{(k+1)} \Big\}.
\end{aligned}$$

Thus, (50) and (51) imply that

$$\begin{aligned}
& \psi^{(k+1)} \\
& \geq n^2\alpha d(\widehat{\mathbf{u}}^{(k+1)},\mathbf{z}^{(k)}) + \Gamma_{k+1} f_\alpha(\mathbf{u}^{(k+1)}) + \gamma_{k+1}\nabla f_\alpha(\mathbf{u}^{(k+1)})^T\big(\widehat{\mathbf{u}}^{(k+1)}-\mathbf{z}^{(k)}\big), \\
& \geq \frac{n^2\alpha}{2}\left\|\widehat{\mathbf{u}}^{(k+1)}-\mathbf{z}^{(k)}\right\|_1^2 + \Gamma_{k+1} f_\alpha(\mathbf{u}^{(k+1)}) + \gamma_{k+1}\nabla f_\alpha(\mathbf{u}^{(k+1)})^T\big(\widehat{\mathbf{u}}^{(k+1)}-\mathbf{z}^{(k)}\big), \\
& \geq \Gamma_{k+1}\Big(f_\alpha(\mathbf{u}^{(k+1)}) + \nabla f_\alpha(\mathbf{u}^{(k+1)})^T\big(\tau_k(\widehat{\mathbf{u}}^{(k+1)}-\mathbf{z}^{(k)})\big) + \frac{n^2\alpha}{2}\left\|\tau_k(\widehat{\mathbf{u}}^{(k+1)}-\mathbf{z}^{(k)})\right\|_1^2\Big), \qquad (52) \\
& \geq \Gamma_{k+1}\Big(f_\alpha(\mathbf{u}^{(k+1)}) + \nabla f_\alpha(\mathbf{u}^{(k+1)})^T\big(\mathbf{y}^{(k+1)}-\mathbf{u}^{(k+1)}\big) + \frac{n^2\alpha}{2}\left\|\mathbf{y}^{(k+1)}-\mathbf{u}^{(k+1)}\right\|_1^2\Big), \qquad (53) \\
& \geq \Gamma_{k+1} f_\alpha(\mathbf{y}^{(k+1)}), \qquad (54)
\end{aligned}$$

where (52) follows from the fact that $\tau_k^2 \leq 1/\Gamma_{k+1}$, (53) follows from the rule in Line 8 in SmoothApprox, and (54) follows from the approximate Taylor's series bound (23). Thus, we have that $(\mathcal{R}_k)$ holds for all $k \geq 0$.

Thus, we have established that the dual variable $\bar{\mathbf{u}}$ produced by SmoothApprox is close to optimal. All that remains to be shown is that the primal variable $\bar{\mathbf{X}}$ is also close to optimal. ¿From the definition of $f_\alpha(\mathbf{u})$ in (38), we have that

$$\begin{aligned} f_\alpha(\mathbf{u}) &= \nabla f_\alpha(\mathbf{u})^T\mathbf{u} - \frac{1}{\alpha}\Big(\sum_{i=1}^{n}\mu_i(\mathbf{X}_u)\log(\mu_i(\mathbf{X}_u)) + s_u\log(s_u)\Big), \\ &\leq \nabla f_\alpha(\mathbf{u})^T\mathbf{u} + n\mathbf{L}\bullet\mathbf{X}_u + \frac{\log(n+1)}{\alpha}. \end{aligned}$$

Then, the induction hypothesis, Line 6 and Line 8 of SmoothApprox imply that

$$\begin{aligned} \Gamma_k f_\alpha(\mathbf{y}^{(k)}) &\leq n^2\alpha\log(n) + \min_{\mathbf{u}\in\mathcal{U}}\left\{\sum_{i=0}^{k}\gamma_i[f_\alpha(\mathbf{u}^{(i)}) + \nabla f_\alpha(\mathbf{u}^{(i)})^T(\mathbf{u}-\mathbf{u}^{(i)})]\right\}, \\ &\leq n^2\alpha\log(n) + \frac{\Gamma_k\log(n+1)}{\alpha} + \Gamma_k\min_{\mathbf{u}\in\mathcal{U}}\left\{n\mathbf{L}\bullet\bar{\mathbf{X}} - 5\sum_{i=1}^{n}u_i(n\bar{x}_{ii}-1)\right\}, \\ &= n^2\alpha\log(n) + \frac{\Gamma_k\log(n+1)}{\alpha} + \Gamma_k\min_{\mathbf{u}\in\mathcal{U}}\phi(\bar{\mathbf{X}},\mathbf{u}). \end{aligned}$$

Thus, we have that for all $T \geq \frac{4\sqrt{n\log(n)}}{\epsilon}$, we have that

$$\min_{\mathbf{u}\in\mathcal{U}}\phi(\widehat{\mathbf{X}},\mathbf{u}) + \epsilon \geq f_\alpha(\mathbf{y}^{(T)}) \geq f(\mathbf{y}^{(T)}) \geq f^*.$$

B Procedure SmoothApproxColoring

The procedure for computing an ϵ-approximate saddle-point for the coloring problem is as follows.

```
SmoothApproxColoring($\mathcal{G}, \epsilon$)
    $SQiters \leftarrow 4n \log n \log^{.5} m/\epsilon$
    $\alpha \leftarrow 2n \log(n+1)/\epsilon$
    $\mathbf{u}^{(0)} \leftarrow \mathbf{v}^{(0)} \leftarrow \frac{1}{n}\mathbf{1}$.
    $(\mathbf{g}_1^{(0)}, \mathbf{g}_2^{(0)}, \mathbf{X}^{(0)}) \leftarrow$ SmoothGradColoring$(\mathbf{u}^{(0)}, \mathbf{v}^{(0)})$
    $\mathbf{s}_i \leftarrow \frac{1}{2}\mathbf{g}_i^{(0)}, i = 1, 2$
    $\widehat{\mathbf{X}} \leftarrow \mathbf{X}^{(0)}$
    $\mathbf{y}_1^{(0)} \leftarrow$ SmoothOpt$\big(\mathbf{u}^{(0)}, \frac{1}{2n^2\alpha}\mathbf{g}_1^{(0)}\big)$
    $\mathbf{y}_2^{(0)} \leftarrow$ SmoothOpt$\big(\mathbf{v}^{(0)}, \frac{1}{2n^2\alpha}\mathbf{g}_2^{(0)}\big)$
    for $k \leftarrow 0$ to $T$
        do
            $\tau_k \leftarrow 2/(k+3)$
            $\mathbf{z}_1^{(k)} \leftarrow$ SmoothOpt$\big(\mathbf{u}^{(0)}, \frac{1}{n^2\alpha}\mathbf{s}_1\big)$
            $\mathbf{z}_2^{(k)} \leftarrow$ SmoothOpt$\big(\mathbf{v}^{(0)}, \frac{1}{n^2\alpha}\mathbf{s}_2\big)$
            $\mathbf{u}^{(k+1)} \leftarrow \tau_k \mathbf{z}_1^{(k)} + (1-\tau_k)\mathbf{y}_1^{(k)}$.
            $\mathbf{v}^{(k+1)} \leftarrow \tau_k \mathbf{z}_2^{(k)} + (1-\tau_k)\mathbf{y}_2^{(k)}$.
            $(\mathbf{g}_1^{(k+1)}, \mathbf{g}_2^{(k+1)}, \mathbf{X}^{(k+1)}) \leftarrow$ SmoothGradColoring$(\mathbf{u}^{(k+1)}, \mathbf{v}^{(k+1)})$
            $(\mathbf{s}_1, \mathbf{s}_2) \leftarrow \frac{k+2}{2}(\mathbf{g}_1^{(0)}, \mathbf{g}_2^{(0)})$
            $\widehat{\mathbf{X}} \leftarrow \widehat{\mathbf{X}} + (k+2)\mathbf{X}^{(k+1)}$
            $\widehat{\mathbf{u}}^{(k+1)} \leftarrow$ SmoothOpt$\big(\mathbf{z}_1^{(k)}, \frac{k+2}{2n^2\alpha}\mathbf{g}_1^{(k+1)}\big)$
            $\widehat{\mathbf{v}}^{k+1} \leftarrow$ SmoothOpt$\big(\mathbf{z}_2^{(k)}, \frac{k+2}{2n^2\alpha}\mathbf{g}_2^{(k+1)}\big)$
            $\mathbf{y}_1^{(k+1)} \leftarrow \tau_k \widehat{\mathbf{u}}_1^{(k+1)} + (1-\tau_k)\mathbf{y}_1^{(k)}$
            $\mathbf{y}_2^{(k+1)} \leftarrow \tau_k \widehat{\mathbf{v}}^{(k+1)} + (1-\tau_k)\mathbf{y}_2^{(k)}$

    $\bar{\mathbf{u}} \leftarrow \mathbf{y}_1^{(T)}$    $\bar{\mathbf{v}} \leftarrow \mathbf{y}_2^{(T)}$    $\bar{\mathbf{X}} \leftarrow \frac{2}{(T+1)(T+2)}\widehat{\mathbf{X}}$
    return $(\bar{\mathbf{X}}, \bar{\mathbf{u}}, \bar{\mathbf{v}})$
```

The procedure SmoothGradColoring$(\mathbf{u}, \mathbf{v})$ returns the gradient $\nabla f_\alpha(\mathbf{u})$ and the matrix $\mathbf{X}_u$ defined in (35). Note that the dual variables $\mathbf{u}$ and $\mathbf{v}$ interact only via SmoothGradColoring.