

The non-convex Burer–Monteiro approach works on smooth semidefinite programs

Nicolas Boumal_{*}
Department of Mathematics
Princeton University
nboumal@math.princeton.edu

Vladislav Voroninski_{*}
Department of Mathematics
Massachusetts Institute of Technology
vvlad@math.mit.edu

Afonso S. Bandeira
Department of Mathematics and Center for Data Science
Courant Institute of Mathematical Sciences, New York University
bandeira@cims.nyu.edu

Abstract

Semidefinite programs (SDP's) can be solved in polynomial time by interior point methods, but scalability can be an issue. To address this shortcoming, over a decade ago, Burer and Monteiro proposed to solve SDP's with few equality constraints via rank-restricted, non-convex surrogates. Remarkably, for some applications, local optimization methods seem to converge to global optima of these non-convex surrogates reliably. Although some theory supports this empirical success, a complete explanation of it remains an open question. In this paper, we consider a class of SDP's which includes applications such as max-cut, community detection in the stochastic block model, robust PCA, phase retrieval and synchronization of rotations. We show that the low-rank Burer–Monteiro formulation of SDP's in that class almost never has any spurious local optima.

1 Introduction

We consider semidefinite programs (SDP's) of the form

$$f^* = \min_{X \in \mathbb{S}^{n \times n}} \langle C, X \rangle \quad \text{subject to} \quad \mathcal{A}(X) = b, \quad X \succeq 0, \quad (\text{SDP})$$

where $\langle C, X \rangle = \text{Tr}(C^\top X)$, $C \in \mathbb{S}^{n \times n}$ is the symmetric cost matrix, $\mathcal{A}: \mathbb{S}^{n \times n} \rightarrow \mathbb{R}^m$ is a linear operator capturing m equality constraints with right hand side $b \in \mathbb{R}^m$ and the variable X is symmetric, positive semidefinite. Interior point methods solve (SDP) in polynomial time [Nesterov, 2004]. In practice however, for n beyond a few thousands, such algorithms run out of memory (and time), prompting research for alternative solvers.

If (SDP) has a compact search space, then it admits a global optimum of rank at most r , where $\frac{r(r+1)}{2} \leq m$ [Pataki, 1998, Barvinok, 1995]. Thus, if one restricts the search space of (SDP) to matrices of rank at most p with $\frac{p(p+1)}{2} \geq m$, then the globally optimal value remains unchanged. This restriction is easily enforced by factorizing $X = YY^\top$ where Y has size $n \times p$, yielding an equivalent quadratically constrained quadratic program:

$$q^* = \min_{Y \in \mathbb{R}^{n \times p}} \langle CY, Y \rangle \quad \text{subject to} \quad \mathcal{A}(YY^\top) = b. \quad (\text{P})$$

_{*}The first two authors contributed equally.

In general, (P) is non-convex, making it a priori unclear how to solve it globally. Still, the benefits are that it is lower dimensional than (SDP) and has no conic constraint. This has motivated Burer and Monteiro [2003, 2005] to try and solve (P) using local optimization methods, with surprisingly good results. They developed theory in support of this observation (details below). About their results, Burer and Monteiro [2005, §3] write (mutatis mutandis):

“How large must we take p so that the local minima of (P) are guaranteed to map to global minima of (SDP)? Our theorem asserts that we need only¹ $\frac{p(p+1)}{2} > m$ (with the important caveat that positive-dimensional faces of (SDP) which are ‘flat’ with respect to the objective function can harbor non-global local minima).”

The caveat—the existence or non-existence of non-global local optima, or their potentially adverse effect for local optimization algorithms—was not further discussed.

In this paper, assuming $\frac{p(p+1)}{2} > m$, we show that if the search space of (SDP) is *compact* and if the search space of (P) is a *smooth manifold*, then, for almost all cost matrices C , if Y satisfies first- and second-order necessary optimality conditions for (P), then Y is a global optimum of (P) and, since $\frac{p(p+1)}{2} \geq m$, $X = YY^\top$ is a global optimum of (SDP); in other words, first- and second-order necessary optimality conditions for (P) are also *sufficient* for global optimality—an unusual theoretical guarantee in non-convex optimization.

Notice that this is a statement about the optimization problem itself, not about specific algorithms. Interestingly, known algorithms for optimization on manifolds converge to *second-order critical points*,² regardless of initialization [Boumal et al., 2016].

For the specified class of SDP’s, our result improves on those of [Burer and Monteiro, 2005] in two important ways. Firstly, for almost all C , we formally exclude the existence of spurious local optima.³ Secondly, we only require the computation of second-order critical points of (P) rather than local optima (which is hard in general [Vavasis, 1991]). Below, we make a statement about computational complexity, and we illustrate the practical efficiency of the proposed methods through numerical experiments.

SDP’s which satisfy the compactness and smoothness assumptions occur in a number of applications including Max-Cut, robust PCA, \mathbb{Z}_2 -synchronization, community detection, cut-norm approximation, phase synchronization, phase retrieval, synchronization of rotations and the trust-region subproblem—see Section 4 for references.

A simple example: the Max-Cut problem

Given an undirected graph, Max-Cut is the NP-hard problem of clustering the n nodes of this graph in two classes, $+1$ and -1 , such that as many edges as possible join nodes of different signs. If C is the adjacency matrix of the graph, Max-Cut is expressed as

$$\max_{x \in \mathbb{R}^n} \frac{1}{4} \sum_{i,j=1}^n C_{ij}(1 - x_i x_j) \quad \text{s.t.} \quad x_1^2 = \dots = x_n^2 = 1. \quad (\text{Max-Cut})$$

Introducing the positive semidefinite matrix $X = xx^\top$, both the cost and the constraints may be expressed linearly in terms of X . Ignoring that X has rank 1 yields the well-known convex relaxation in the form of a semidefinite program (up to an affine transformation of the cost):

$$\min_{X \in \mathbb{S}^{n \times n}} \langle C, X \rangle \quad \text{s.t.} \quad \text{diag}(X) = \mathbf{1}, \quad X \succeq 0. \quad (\text{Max-Cut SDP})$$

If a solution X of this SDP has rank 1, then $X = xx^\top$ for some x which is then an optimal cut. In the general case of higher rank X , Goemans and Williamson [1995] exhibited the celebrated rounding scheme to produce approximately optimal cuts (within a ratio of .878) from X .

¹The condition on p and m is slightly, but inconsequentially, different in [Burer and Monteiro, 2005].

²Second-order critical points satisfy first- and second-order necessary optimality conditions.

³Before Prop. 2.3 in [Burer and Monteiro, 2005], the authors write: “The change of variables $X = YY^\top$ does not introduce any extraneous local minima.” This is sometimes misunderstood to mean (P) does not have spurious local optima, when it actually means that the local optima of (P) are in exact correspondence with the local optima of “(SDP) with the extra constraint $\text{rank}(X) \leq p$,” which is also non-convex and thus also liable to having local optima. Unfortunately, this misinterpretation has led to some confusion in the literature.

The corresponding Burer–Monteiro non-convex problem with rank bounded by p is:

$$\min_{Y \in \mathbb{R}^{n \times p}} \langle CY, Y \rangle \quad \text{s.t.} \quad \text{diag}(YY^\top) = \mathbf{1}. \quad (\text{Max-Cut BM})$$

The constraint $\text{diag}(YY^\top) = \mathbf{1}$ requires each row of Y to have unit norm; that is: Y is a point on the Cartesian product of n unit spheres in \mathbb{R}^p , which is a smooth manifold. Furthermore, all X feasible for the SDP have identical trace equal to n , so that the search space of the SDP is compact. Thus, our results stated below apply:

For $p = \lceil \sqrt{2n} \rceil$, for almost all C , even though (Max-Cut BM) is non-convex, any local optimum Y is a global optimum (and so is $X = YY^\top$), and all saddle points have an escape (the Hessian has a negative eigenvalue).

We note that, for $p > n/2$, the same holds for all C [Boumal, 2015].

Notation

$\mathbb{S}^{n \times n}$ is the set of real, symmetric matrices of size n . A symmetric matrix X is positive semidefinite ($X \succeq 0$) if and only if $u^\top Xu \geq 0$ for all $u \in \mathbb{R}^n$. For matrices A, B , the standard Euclidean inner product is $\langle A, B \rangle = \text{Tr}(A^\top B)$. The associated (Frobenius) norm is $\|A\| = \sqrt{\langle A, A \rangle}$. Id is the identity operator and I_n is the identity matrix of size n .

2 Main results

Our main result establishes conditions under which first- and second-order necessary optimality conditions for (P) are sufficient for global optimality. Under those conditions, it is a fortiori true that global optima of (P) map to global optima of (SDP), so that local optimization methods on (P) can be used to solve the higher-dimensional, cone-constrained (SDP).

We now specify the necessary optimality conditions of (P). Under the assumptions of our main result below (Theorem 2), the search space

$$\mathcal{M} = \mathcal{M}_p = \{Y \in \mathbb{R}^{n \times p} : \mathcal{A}(YY^\top) = b\} \quad (1)$$

is a smooth and compact manifold. As such, it can be linearized at each point $Y \in \mathcal{M}$ by a tangent space, simply by differentiating the constraints [Absil et al., 2008, eq. (3.19)]:

$$\text{T}_Y \mathcal{M} = \{\dot{Y} \in \mathbb{R}^{n \times p} : \mathcal{A}(\dot{Y}Y^\top + Y\dot{Y}^\top) = 0\}. \quad (2)$$

Endowing the tangent spaces of \mathcal{M} with the (restricted) Euclidean metric $\langle A, B \rangle = \text{Tr}(A^\top B)$ turns \mathcal{M} into a Riemannian submanifold of $\mathbb{R}^{n \times p}$. In general, second-order optimality conditions can be intricate to handle [Ruszczynski, 2006]. Fortunately, here, the smoothness of both the search space (1) and the cost function

$$f(Y) = \langle CY, Y \rangle \quad (3)$$

make for straightforward conditions. In spirit, they coincide with the well-known conditions for unconstrained optimization. As further detailed in Appendix A, the Riemannian gradient $\text{grad} f(Y)$ is the orthogonal projection of the classical gradient of f to the tangent space $\text{T}_Y \mathcal{M}$. The Riemannian Hessian of f at Y is a similarly restricted version of the classical Hessian of f to the tangent space.

Definition 1. A (first-order) critical point for (P) is a point $Y \in \mathcal{M}$ such that

$$\text{grad} f(Y) = 0, \quad (1\text{st order nec. opt. cond.})$$

where $\text{grad} f(Y) \in \text{T}_Y \mathcal{M}$ is the Riemannian gradient at Y of f restricted to \mathcal{M} . A second-order critical point for (P) is a critical point Y such that

$$\text{Hess} f(Y) \succeq 0, \quad (2\text{nd order nec. opt. cond.})$$

where $\text{Hess} f(Y) : \text{T}_Y \mathcal{M} \rightarrow \text{T}_Y \mathcal{M}$ is the Riemannian Hessian at Y of f restricted to \mathcal{M} (a symmetric linear operator).

Proposition 1. *All local (and global) optima of (P) are second-order critical points.*

Proof. See [Yang et al., 2014, Rem. 4.2 and Cor. 4.2]. \square

We can now state our main result. In the theorem statement below, “for almost all C ” means potentially troublesome cost matrices form at most a (Lebesgue) zero-measure subset of $\mathbb{S}^{n \times n}$, in the same way that almost all square matrices are invertible. In particular, given any matrix $C \in \mathbb{S}^{n \times n}$, perturbing C to $C + \sigma W$ where W is a Wigner random matrix results in an acceptable cost matrix with probability 1, for arbitrarily small $\sigma > 0$.

Theorem 2. *Given constraints $\mathcal{A}: \mathbb{S}^{n \times n} \rightarrow \mathbb{R}^m$, $b \in \mathbb{R}^m$ and p satisfying $\frac{p(p+1)}{2} > m$, if*

- (i) *the search space of (SDP) is compact; and*
- (ii) *the search space of (P) is a smooth manifold,*

then for almost all cost matrices $C \in \mathbb{S}^{n \times n}$, any second-order critical point of (P) is globally optimal. Under these conditions, if Y is globally optimal for (P), then the matrix $X = YY^\top$ is globally optimal for (SDP).

The assumptions are discussed in the next section. The proof—see Appendix A—follows directly from the combination of two intermediate results:

1. If Y is *rank deficient* and second-order critical for (P), then it is globally optimal and $X = YY^\top$ is optimal for (SDP); and
2. If $\frac{p(p+1)}{2} > m$, then, for almost all C , every first-order critical Y is rank-deficient.

The first step holds in a more general context, as previously established by Burer and Monteiro [2003, 2005]. The second step is new and crucial, as it allows to formally exclude the existence of spurious local optima, generically in C , thus resolving the caveat mentioned in the introduction.

The smooth structure of (P) naturally suggests using *Riemannian optimization* to solve it [Absil et al., 2008], which is something that was already proposed by Journée et al. [2010] in the same context. Importantly, known algorithms converge to second-order critical points regardless of initialization. We state here a recent computational result to that effect.

Proposition 3. *Under the numbered assumptions of Theorem 2, the Riemannian trust-region method (RTR) [Absil et al., 2007] initialized with any $Y_0 \in \mathcal{M}$ returns in $\mathcal{O}(1/\varepsilon_g^2 \varepsilon_H + 1/\varepsilon_H^3)$ iterations a point $Y \in \mathcal{M}$ such that*

$$f(Y) \leq f(Y_0), \quad \|\text{grad} f(Y)\| \leq \varepsilon_g, \quad \text{and} \quad \text{Hess} f(Y) \succeq -\varepsilon_H \text{Id}.$$

Proof. Apply the main results of [Boumal et al., 2016] using that f has locally Lipschitz continuous gradient and Hessian in $\mathbb{R}^{n \times p}$ and \mathcal{M} is a compact submanifold of $\mathbb{R}^{n \times p}$. \square

Essentially, each iteration of RTR requires evaluation of one cost and one gradient, a bounded number of Hessian-vector applications, and one projection from $\mathbb{R}^{n \times p}$ to \mathcal{M} . In many important cases, this projection amounts to Gram–Schmidt orthogonalization of small blocks of Y —see Section 4.

Proposition 3 bounds worst-case iteration counts for arbitrary initialization. In practice, a good initialization point may be available, making the local convergence rate of RTR more informative. For RTR, one may expect superlinear or even quadratic local convergence rates near isolated local minimizers [Absil et al., 2007]. While minimizers are not isolated in our case [Journée et al., 2010], experiments show a characteristically superlinear local convergence rate in practice [Boumal, 2015]. This means high accuracy solutions can be achieved, as demonstrated in Appendix B.

Thus, under the conditions of Theorem 2, generically in C , RTR converges to global optima. In practice, the algorithm returns after a finite number of steps, and only *approximate* second-order criticality is guaranteed. Hence, it is interesting to bound the optimality gap in terms of the approximation quality. Unfortunately, we do not establish such a result for small p . Instead, we give an a posteriori computable optimality gap bound which holds for *all* p and for *all* C . In the following statement, the dependence of \mathcal{M} on p is explicit, as \mathcal{M}_p . The proof is in Appendix A.

Theorem 4. Let $R < \infty$ be the maximal trace of any X feasible for (SDP). For any p such that \mathcal{M}_p and \mathcal{M}_{p+1} are smooth manifolds (even if $\frac{p(p+1)}{2} \leq m$) and for any $Y \in \mathcal{M}_p$, form $\tilde{Y} = [Y | 0_{n \times 1}]$ in \mathcal{M}_{p+1} . The optimality gap at Y is bounded as

$$0 \leq 2(f(Y) - f^*) \leq \sqrt{R} \|\text{grad} f(Y)\| - R \lambda_{\min}(\text{Hess} f(\tilde{Y})). \quad (4)$$

If all feasible X have the same trace R and there exists a positive definite feasible X , then the bound simplifies to

$$0 \leq 2(f(Y) - f^*) \leq -R \lambda_{\min}(\text{Hess} f(\tilde{Y})) \quad (5)$$

so that $\|\text{grad} f(Y)\|$ needs not be controlled explicitly. If $p > n$, the bounds hold with $\tilde{Y} = Y$.

In particular, for $p = n + 1$, the bound can be controlled a priori: approximate second-order critical points are approximately optimal, for any C .⁴

Corollary 5. Under the assumptions of Theorem 4, if $p = n + 1$ and $Y \in \mathcal{M}$ satisfies both $\|\text{grad} f(Y)\| \leq \varepsilon_g$ and $\text{Hess} f(Y) \succeq -\varepsilon_H \text{Id}$, then Y is approximately optimal in the sense that

$$0 \leq 2(f(Y) - f^*) \leq \sqrt{R} \varepsilon_g + R \varepsilon_H.$$

Under the same condition as in Theorem 4, the bound can be simplified to $R \varepsilon_H$.

This works well with Proposition 3. For any p , equation (4) also implies the following:

$$\lambda_{\min}(\text{Hess} f(\tilde{Y})) \leq -\frac{2(f(Y) - f^*) - \sqrt{R} \|\text{grad} f(Y)\|}{R}.$$

That is, for any p and any C , an approximate critical point Y in \mathcal{M}_p which is far from optimal maps to a comfortably-escapable approximate saddle point \tilde{Y} in \mathcal{M}_{p+1} .

This suggests an algorithm as follows. For a starting value of p such that \mathcal{M}_p is a manifold, use RTR to compute an approximate second-order critical point Y . Then, form \tilde{Y} in \mathcal{M}_{p+1} and test the left-most eigenvalue of $\text{Hess} f(\tilde{Y})$.⁵ If it is close enough to zero, this provides a good bound on the optimality gap. If not, use an (approximate) eigenvector associated to $\lambda_{\min}(\text{Hess} f(\tilde{Y}))$ to escape the approximate saddle point and apply RTR from that new point in \mathcal{M}_{p+1} ; iterate. In the worst-case scenario, p grows to $n + 1$, at which point all approximate second-order critical points are approximate optima. Theorem 2 suggests $p = \lceil \sqrt{2m} \rceil$ should suffice for C bounded away from a zero-measure set. Such an algorithm already features with less theory in [Journée et al., 2010] and [Boumal, 2015]; in the latter, it is called the *Riemannian staircase*, for it lifts (P) floor by floor.

Related work

Low-rank approaches to solve SDP's have featured in a number of recent research papers. We highlight just two which illustrate different classes of SDP's of interest.

Shah et al. [2016] tackle SDP's with linear cost and linear constraints (both equalities and inequalities) via low-rank factorizations, assuming the matrices appearing in the cost and constraints are positive semidefinite. They propose a non-trivial initial guess to partially overcome non-convexity with great empirical results, but do not provide optimality guarantees.

Bhojanapalli et al. [2016a] on the other hand consider the minimization of a convex cost function over positive semidefinite matrices, without constraints. Such problems could be obtained from generic SDP's by penalizing the constraints in a Lagrangian way. Here too, non-convexity is partially overcome via non-trivial initialization, with global optimality guarantees under some conditions.

Also of interest are recent results about the harmlessness of non-convexity in low-rank matrix completion [Ge et al., 2016, Bhojanapalli et al., 2016b]. Similarly to the present work, the authors there show there is no need for special initialization despite non-convexity.

⁴With $p = n + 1$, problem (P) is no longer lower dimensional than (SDP), but retains the advantage of not involving a positive semidefiniteness constraint.

⁵It may be more practical to test $\lambda_{\min}(S)$ (14) rather than $\lambda_{\min}(\text{Hess} f)$. Lemma 7 relates the two. See [Journée et al., 2010, §3.3] to construct escape tangent vectors from S .

3 Discussion of the assumptions

Our main result, Theorem 2, comes with geometric assumptions on the search spaces of both (SDP) and (P) which we now discuss. Examples of SDP's which fit the assumptions of Theorem 2 are featured in the next section.

The assumption that the search space of (SDP),

$$\mathcal{C} = \{X \in \mathbb{S}^{n \times n} : \mathcal{A}(X) = b, X \succeq 0\}, \quad (6)$$

is *compact* works in pair with the assumption $\frac{p(p+1)}{2} > m$ as follows. For (P) to reveal the global optima of (SDP), it is necessary that (SDP) admits a solution of rank at most p . One way to ensure this is via the Pataki–Barvinok theorems [Pataki, 1998, Barvinok, 1995], which state that all *extreme points* of \mathcal{C} have rank r bounded as $\frac{r(r+1)}{2} \leq m$. Extreme points are faces of dimension zero (such as vertices for a cube). When optimizing a linear cost function $\langle C, X \rangle$ over a compact convex set \mathcal{C} , at least one extreme point is a global optimum [Rockafellar, 1970, Cor. 32.3.2]—this is not true in general if \mathcal{C} is not compact. Thus, under the assumptions of Theorem 2, there is a point $Y \in \mathcal{M}$ such that $X = YY^\top$ is an optimal extreme point of (SDP); then, of course, Y itself is optimal for (P).

In general, the Pataki–Barvinok bound is tight, in that there exist extreme points of rank up to that upper-bound (rounded down)—see for example [Laurent and Poljak, 1996] for the Max-Cut SDP and [Boumal, 2015] for the Orthogonal-Cut SDP. Let C (the cost matrix) be the negative of such an extreme point. Then, the unique optimum of (SDP) is that extreme point, showing that $\frac{p(p+1)}{2} \geq m$ is necessary for (SDP) and (P) to be equivalent for all C . We further require a strict inequality because our proof relies on properties of rank deficient Y 's in \mathcal{M} .

The assumption that \mathcal{M} (eq. (1)) is a *smooth manifold* works in pair with the ambition that the result should hold for (almost) all cost matrices C . The starting point is that, for a given non-convex smooth optimization problem—even a quadratically constrained quadratic program—computing local optima is hard in general [Vavasis, 1991]. Thus, we wish to restrict our attention to efficiently computable points, such as points which satisfy first- and second-order KKT conditions for (P)—see [Burer and Monteiro, 2003, §2.2] and [Ruszczyński, 2006, §3]. This only makes sense if global optima satisfy the latter, that is, if KKT conditions are necessary for optimality. A global optimum Y necessarily satisfies KKT conditions if *constraint qualifications* (CQ's) hold at Y [Ruszczyński, 2006]. The standard CQ's for equality constrained programs are Robinson's conditions or metric regularity (they are here equivalent). They read as follows, assuming $\mathcal{A}(YY^\top)_i = \langle A_i, YY^\top \rangle$ for some matrices $A_1, \dots, A_m \in \mathbb{S}^{n \times n}$:

$$\text{CQ's hold at } Y \text{ if } A_1 Y, \dots, A_m Y \text{ are linearly independent in } \mathbb{R}^{n \times p}. \quad (7)$$

Considering almost all C , global optima could, a priori, be almost anywhere in \mathcal{M} . To simplify, we require CQ's to hold at all Y 's in \mathcal{M} rather than only at the (unknown) global optima. This turns out to be a sufficient condition for \mathcal{M} to be a smooth manifold of codimension m [Absil et al., 2008, Prop. 3.3.3]. Indeed, tangent vectors $\dot{Y} \in T_Y \mathcal{M}$ (2) are exactly those vectors that satisfy $\langle A_i Y, \dot{Y} \rangle = 0$: under CQ's, the $A_i Y$'s form a basis of the normal space to the manifold at Y .

Once it is decided that \mathcal{M} must be a manifold, we can step away from the specific representation of it via the matrices A_1, \dots, A_m and reason about optimality conditions on the manifold directly. Adding redundant constraints (for example, duplicating A_1) would break the CQ's, but not the manifold structure. Hence, stating Theorem 2 in terms of manifolds better captures the role of \mathcal{M} than stating it in terms of CQ's. See also [Andreani et al., 2010, Thm. 3.3] for a proof that requiring \mathcal{M} to be a manifold around Y is a type of CQ.

Finally, we note that Theorem 2 only applies for *almost* all C , rather than all C . To justify this restriction, if indeed it is justified, one should exhibit a matrix C that leads to suboptimal second-order critical points while other assumptions are satisfied. We do not have such an example. We do observe that (Max-Cut SDP) on cycles of certain even lengths has a unique solution of rank 1, while the corresponding (Max-Cut BM) with $p = 2$ has suboptimal local optima (strictly, if we quotient out symmetries). This at least suggests it is not enough, for generic C , to set p just larger than the rank of the solutions of the SDP. (For those same examples, at $p = 3$, we consistently observe convergence to global optima.)

4 Examples of smooth SDP's

The canonical examples of SDP's which satisfy the assumptions in Theorem 2 are those where the diagonal blocks of X or their traces are fixed. We note that the algorithms and the theory continue to hold for complex matrices, where the set of Hermitian matrices of size n is treated as a real vector space of dimension n^2 (instead of $\frac{n(n+1)}{2}$ in the real case) with inner product $\langle H_1, H_2 \rangle = \Re \{ \text{Tr}(H_1^* H_2) \}$, so that occurrences of $\frac{p(p+1)}{2}$ are replaced by p^2 .

Certain concrete examples of SDP's include:

$$\begin{aligned} \min_X \langle C, X \rangle \text{ s.t. } \text{Tr}(X) = 1, X \succeq 0; & \quad (\text{fixed trace}) \\ \min_X \langle C, X \rangle \text{ s.t. } \text{diag}(X) = \mathbf{1}, X \succeq 0; & \quad (\text{fixed diagonal}) \\ \min_X \langle C, X \rangle \text{ s.t. } X_{ii} = I_d, X \succeq 0. & \quad (\text{fixed diagonal blocks}) \end{aligned}$$

Their rank-constrained counterparts read as follows (matrix norms are Frobenius norms):

$$\begin{aligned} \min_{Y: n \times p} \langle CY, Y \rangle \text{ s.t. } \|Y\| = 1; & \quad (\text{sphere}) \\ \min_{Y: n \times p} \langle CY, Y \rangle \text{ s.t. } Y^\top = [y_1 \ \cdots \ y_n] \text{ and } \|y_i\| = 1 \text{ for all } i; & \quad (\text{product of spheres}) \\ \min_{Y: qd \times p} \langle CY, Y \rangle \text{ s.t. } Y^\top = [Y_1 \ \cdots \ Y_q] \text{ and } Y_i^\top Y_i = I_d \text{ for all } i. & \quad (\text{product of Stiefel}) \end{aligned}$$

The first example has only one constraint: the SDP always admits an optimal rank 1 solution, corresponding to an eigenvector associated to the left-most eigenvalue of C . This generalizes to the trust-region subproblem as well.

For the second example, in the real case, $p = 1$ forces $y_i = \pm 1$, allowing to capture combinatorial problems such as Max-Cut [Goemans and Williamson, 1995], \mathbb{Z}_2 -synchronization [Javanmard et al., 2015] and community detection in the stochastic block model [Abbe et al., 2016, Bandeira et al., 2016b]. The same SDP is central in a formulation of robust PCA [McCoy and Tropp, 2011] and is used to approximate the cut-norm of a matrix [Alon and Naor, 2006]. Theorem 2 states that for almost all C , $p = \lceil \sqrt{2n} \rceil$ is sufficient. In the complex case, $p = 1$ forces $|y_i| = 1$, allowing to capture problems where phases must be recovered; in particular, phase synchronization [Bandeira et al., 2016a, Singer, 2011] and phase retrieval via Phase-Cut [Waldspurger et al., 2015]. For almost all C , it is then sufficient to set $p = \lfloor \sqrt{n} + 1 \rfloor$.

In the third example, Y of size $n \times p$ is divided in q slices of size $d \times p$, with $p \geq d$. Each slice has orthonormal rows. For $p = d$, the slices are orthogonal (or unitary) matrices, allowing to capture Orthogonal-Cut [Bandeira et al., 2016c] and the related problems of synchronization of rotations [Wang and Singer, 2013] and permutations. Synchronization of rotations is an important step in simultaneous localization and mapping, for example. Here, it is sufficient for almost all C to let $p = \lceil \sqrt{d(d+1)q} \rceil$.

SDP's with constraints that are combinations of the above examples can also have the smoothness property; the right-hand sides 1 and I_d can be replaced by any positive definite right-hand sides by a change of variables. Another simple rule to check is if the constraint matrices $A_1, \dots, A_m \in \mathbb{S}^{n \times n}$ such that $\mathcal{A}(X)_i = \langle A_i, X \rangle$ satisfy $A_i A_j = 0$ for all $i \neq j$ (note that this is stronger than requiring $\langle A_i, A_j \rangle = 0$), see [Journée et al., 2010].

5 Conclusions

The Burer–Monteiro approach consists in replacing optimization of a linear function $\langle C, X \rangle$ over the convex set $\{X \succeq 0 : \mathcal{A}(X) = b\}$ with optimization of the quadratic function $\langle CY, Y \rangle$ over the non-convex set $\{Y \in \mathbb{R}^{n \times p} : \mathcal{A}(YY^\top) = b\}$. It was previously known that, if the convex set is compact and p satisfies $\frac{p(p+1)}{2} \geq m$ where m is the number of constraints, then these two problems have the same global optimum. It was also known from [Burer and Monteiro, 2005] that spurious local optima Y , if they exist, must map to special faces of the compact convex set, but without statement as to the prevalence of such faces or the risk they pose for local optimization methods. In

this paper we showed that, if the set of X 's is compact and the set of Y 's is a smooth manifold, and if $\frac{p(p+1)}{2} > m$, then for almost all C , the non-convexity of the problem in Y is benign, in that all Y 's which satisfy second-order necessary optimality conditions are in fact globally optimal.

We further reference the Riemannian trust-region method [Absil et al., 2007] to solve the problem in Y , as it was recently guaranteed to converge from any starting point to a point which satisfies second-order optimality conditions, with global convergence rates [Boumal et al., 2016]. In addition, for $p = n + 1$, we guarantee that approximate satisfaction of second-order conditions implies approximate global optimality. We note that the $1/\varepsilon^3$ convergence rate in our results may be pessimistic. Indeed, the numerical experiments clearly show that high accuracy solutions can be computed fast using optimization on manifolds, at least for certain applications.

Addressing a broader class of SDP's, such as those with inequality constraints or equality constraints that may violate our smoothness assumptions, could perhaps be handled by penalizing those constraints in the objective in an augmented Lagrangian fashion. We also note that, algorithmically, the Riemannian trust-region method we use applies just as well to nonlinear costs in the SDP. We believe that extending the theory presented here to broader classes of problems is a good direction for future work.

Acknowledgment

VV was partially supported by the Office of Naval Research. ASB was supported by NSF Grant DMS-1317308. Part of this work was done while ASB was with the Department of Mathematics at the Massachusetts Institute of Technology. We thank Wotao Yin and Michel Goemans for helpful discussions.

References

- E. Abbe, A.S. Bandeira, and G. Hall. Exact recovery in the stochastic block model. *Information Theory, IEEE Transactions on*, 62(1):471–487, 2016.
- P.-A. Absil, C. G. Baker, and K. A. Gallivan. Trust-region methods on Riemannian manifolds. *Foundations of Computational Mathematics*, 7(3):303–330, 2007. doi:[10.1007/s10208-005-0179-9](https://doi.org/10.1007/s10208-005-0179-9).
- P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, Princeton, NJ, 2008. ISBN 978-0-691-13298-3.
- N. Alon and A. Naor. Approximating the cut-norm via Grothendieck's inequality. *SIAM Journal on Computing*, 35(4):787–803, 2006. doi:[10.1137/S0097539704441629](https://doi.org/10.1137/S0097539704441629).
- R. Andreani, C. E. Echagüe, and M. L. Schuverdt. Constant-rank condition and second-order constraint qualification. *Journal of Optimization Theory and Applications*, 146(2):255–266, 2010. doi:[10.1007/s10957-010-9671-8](https://doi.org/10.1007/s10957-010-9671-8).
- A.S. Bandeira, N. Boumal, and A. Singer. Tightness of the maximum likelihood semidefinite relaxation for angular synchronization. *Mathematical Programming*, pages 1–23, 2016a. doi:[10.1007/s10107-016-1059-6](https://doi.org/10.1007/s10107-016-1059-6).
- A.S. Bandeira, N. Boumal, and V. Voroninski. On the low-rank approach for semidefinite programs arising in synchronization and community detection. In *Proceedings of The 29th Conference on Learning Theory, COLT 2016, New York, NY, June 23–26, 2016b*.
- A.S. Bandeira, C. Kennedy, and A. Singer. Approximating the little Grothendieck problem over the orthogonal and unitary groups. *Mathematical Programming*, pages 1–43, 2016c. doi:[10.1007/s10107-016-0993-7](https://doi.org/10.1007/s10107-016-0993-7).
- A.I. Barvinok. Problems of distance geometry and convex properties of quadratic maps. *Discrete & Computational Geometry*, 13(1):189–202, 1995. doi:[10.1007/BF02574037](https://doi.org/10.1007/BF02574037).
- S. Bhojanapalli, A. Kyrillidis, and S. Sanghavi. Dropping convexity for faster semi-definite optimization. *Conference on Learning Theory (COLT)*, 2016a.
- S. Bhojanapalli, B. Neyshabur, and N. Srebro. Global optimality of local search for low rank matrix recovery. *arXiv preprint arXiv:1605.07221*, 2016b.
- N. Boumal. A Riemannian low-rank method for optimization over semidefinite matrices with block-diagonal constraints. *arXiv preprint arXiv:1506.00575*, 2015.

- N. Boumal, B. Mishra, P.-A. Absil, and R. Sepulchre. Manopt, a Matlab toolbox for optimization on manifolds. *Journal of Machine Learning Research*, 15:1455–1459, 2014. URL <http://www.manopt.org>.
- N. Boumal, P.-A. Absil, and C. Cartis. Global rates of convergence for nonconvex optimization on manifolds. *arXiv preprint arXiv:1605.08101*, 2016.
- S. Burer and R.D.C. Monteiro. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming*, 95(2):329–357, 2003. doi:[10.1007/s10107-002-0352-8](https://doi.org/10.1007/s10107-002-0352-8).
- S. Burer and R.D.C. Monteiro. Local minima and convergence in low-rank semidefinite programming. *Mathematical Programming*, 103(3):427–444, 2005.
- CVX. CVX: Matlab software for disciplined convex programming. <http://cvxr.com/cvx>, August 2012.
- R. Ge, J.D. Lee, and T. Ma. Matrix completion has no spurious local minimum. *arXiv preprint arXiv:1605.07272*, 2016.
- M.X. Goemans and D.P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, 42(6):1115–1145, 1995. doi:[10.1145/227683.227684](https://doi.org/10.1145/227683.227684).
- C. Helmberg, F. Rendl, R.J. Vanderbei, and H. Wolkowicz. An interior-point method for semidefinite programming. *SIAM Journal on Optimization*, 6(2):342–361, 1996. doi:[10.1137/0806020](https://doi.org/10.1137/0806020).
- A. Javanmard, A. Montanari, and F. Ricci-Tersenghi. Phase transitions in semidefinite relaxations. *arXiv preprint arXiv:1511.08769*, 2015.
- M. Journée, F. Bach, P.-A. Absil, and R. Sepulchre. Low-rank optimization on the cone of positive semidefinite matrices. *SIAM Journal on Optimization*, 20(5):2327–2351, 2010. doi:[10.1137/080731359](https://doi.org/10.1137/080731359).
- M. Laurent and S. Poljak. On the facial structure of the set of correlation matrices. *SIAM Journal on Matrix Analysis and Applications*, 17(3):530–547, 1996. doi:[10.1137/0617031](https://doi.org/10.1137/0617031).
- M. McCoy and J.A. Tropp. Two proposals for robust PCA using semidefinite programming. *Electronic Journal of Statistics*, 5:1123–1160, 2011. doi:[10.1214/11-EJS636](https://doi.org/10.1214/11-EJS636).
- Y. Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87 of *Applied optimization*. Springer, 2004. ISBN 978-1-4020-7553-7.
- G. Pataki. On the rank of extreme matrices in semidefinite programs and the multiplicity of optimal eigenvalues. *Mathematics of operations research*, 23(2):339–358, 1998. doi:[10.1287/moor.23.2.339](https://doi.org/10.1287/moor.23.2.339).
- R.T. Rockafellar. *Convex analysis*. Princeton University Press, Princeton, NJ, 1970.
- A.P. Ruszczyński. *Nonlinear optimization*. Princeton University Press, Princeton, NJ, 2006.
- S. Shah, A. Kumar, D. Jacobs, C. Studer, and T. Goldstein. Biconvex relaxation for semidefinite programming in computer vision. *arXiv preprint arXiv:1605.09527*, 2016.
- A. Singer. Angular synchronization by eigenvectors and semidefinite programming. *Applied and Computational Harmonic Analysis*, 30(1):20–36, 2011. doi:[10.1016/j.acha.2010.02.001](https://doi.org/10.1016/j.acha.2010.02.001).
- K.C. Toh, M.J. Todd, and R.H. Tütüncü. SDPT3—a MATLAB software package for semidefinite programming. *Optimization Methods and Software*, 11(1–4):545–581, 1999. doi:[10.1080/10556789908805762](https://doi.org/10.1080/10556789908805762).
- S.A. Vavasis. *Nonlinear optimization: complexity issues*. Oxford University Press, Inc., 1991.
- I. Waldspurger, A. d’Aspremont, and S. Mallat. Phase recovery, MaxCut and complex semidefinite programming. *Mathematical Programming*, 149(1–2):47–81, 2015. doi:[10.1007/s10107-013-0738-9](https://doi.org/10.1007/s10107-013-0738-9).
- L. Wang and A. Singer. Exact and stable recovery of rotations for robust synchronization. *Information and Inference*, 2(2):145–193, 2013. doi:[10.1093/imaiai/iat005](https://doi.org/10.1093/imaiai/iat005).
- Z. Wen and W. Yin. A feasible method for optimization with orthogonality constraints. *Mathematical Programming*, 142(1–2):397–434, 2013. doi:[10.1007/s10107-012-0584-1](https://doi.org/10.1007/s10107-012-0584-1).
- W.H. Yang, L.-H. Zhang, and R. Song. Optimality conditions for the nonlinear programming problems on Riemannian manifolds. *Pacific Journal of Optimization*, 10(2):415–434, 2014.

A Proofs and additional lemmas

We start by working out explicit formulas for the Riemannian gradient and Hessian which appear in Definition 1. Let $\text{Proj}_Y: \mathbb{R}^{n \times p} \rightarrow \text{T}_Y \mathcal{M}$ be the orthogonal projector to the tangent space at Y (eq. (2)), and let

$$\nabla f(Y) = 2CY, \quad \nabla^2 f(Y)[\dot{Y}] = 2C\dot{Y} \quad (8)$$

be the (Euclidean) gradient and Hessian of the cost function (3). The Riemannian gradient and Hessian of f on \mathcal{M} are related to these as follows [see Absil et al., 2008, eqs (3.37), (5.15)]:

$$\text{grad} f(Y) = \text{Proj}_Y \nabla f(Y), \quad (9)$$

$$\forall \dot{Y} \in \text{T}_Y \mathcal{M}, \quad \text{Hess} f(Y)[\dot{Y}] = \text{Proj}_Y D(Y \mapsto \text{grad} f(Y)) (Y)[\dot{Y}]. \quad (10)$$

Let us focus on the gradient first. Since $\text{grad} f(Y)$ is a tangent vector at Y (2),⁶

$$\mathcal{A}(\text{grad} f(Y) Y^\top) = 0, \quad (11)$$

and since it is the orthogonal projection of $\nabla f(Y)$ to the tangent space, there exists $\mu \in \mathbb{R}^m$ such that

$$\text{grad} f(Y) + 2\mathcal{A}^*(\mu)Y = \nabla f(Y) = 2CY, \quad (12)$$

where $\mathcal{A}^*: \mathbb{R}^m \rightarrow \mathbb{S}^{n \times n}$ is the adjoint of \mathcal{A} . Indeed, considering symmetric matrices A_1, \dots, A_m such that $\mathcal{A}(X)_i = \langle A_i, X \rangle$, matrices $\mathcal{A}^*(\mu)Y = \mu_1 A_1 Y + \dots + \mu_m A_m Y$ span the normal space to the manifold at Y . Right-multiply (12) with Y^\top and apply \mathcal{A} to obtain

$$\mathcal{A}(\mathcal{A}^*(\mu)Y Y^\top) = \mathcal{A}(C Y Y^\top). \quad (13)$$

If the $A_i Y$'s are linearly independent as in (7), then μ is the unique solution to this linear system—for KKT points, these are the Lagrange multipliers. In general, μ may not be unique but $\mathcal{A}^*(\mu)Y$ always is, since (13) is the result of an orthogonal projection. Furthermore, contrary to classical KKT conditions, $\mathcal{A}^*(\mu)Y$ is defined for *all* feasible Y (not only for KKT points) and can be found by solving (13).⁷ If μ is not unique, for ease of exposition, let $\mu = \mu(Y) = \mu(Y Y^\top)$ be the smallest-norm acceptable μ : this is a well-defined, differentiable function of Y .⁸ Using this definition of μ , let

$$S = S(Y) = S(Y Y^\top) = C - \mathcal{A}^*(\mu). \quad (14)$$

First-order critical points then satisfy (using (12)):

$$\frac{1}{2} \text{grad} f(Y) = SY = 0. \quad (15)$$

We note in passing that $\mu(Y)$ is feasible for the dual of (SDP) exactly when $S(Y) \succeq 0$:

$$d^* = \max_{\mu \in \mathbb{R}^m} b^\top \mu \text{ subject to } C - \mathcal{A}^*(\mu) \succeq 0, \quad (\text{DSDP})$$

which illustrates the importance of S as a dual certificate for (SDP).

Now let us turn to the Hessian of f . Equation (10) requires computation of the differential of $\text{grad} f(Y)$, which is

$$D(Y \mapsto \text{grad} f(Y))(Y)[\dot{Y}] = D(Y \mapsto 2SY)(Y)[\dot{Y}] = 2S\dot{Y} + 2\dot{S}Y,$$

where $\dot{S} \triangleq DS(Y)[\dot{Y}]$ is a symmetric matrix. Because of eq. (14), $\dot{S} = \mathcal{A}^*(\nu)$ for some $\nu \in \mathbb{R}^m$. Hence, for any tangent vector $\dot{Z} \in \text{T}_Y \mathcal{M}$ (2), we have $\langle \dot{Z}, \dot{S}Y \rangle = \langle \dot{Z} Y^\top, \mathcal{A}^*(\nu) \rangle = \langle \mathcal{A}(\dot{Z} Y^\top), \nu \rangle = 0$: $\dot{S}Y$ is orthogonal to the tangent space at Y . Using (10), we find that

$$\frac{1}{2} \text{Hess} f(Y)[\dot{Y}] = \text{Proj}_Y S\dot{Y}. \quad (16)$$

⁶For ease of notation, for non-symmetric $B \in \mathbb{R}^{n \times n}$, $\mathcal{A}(B)$ refers to $\mathcal{A}\left(\frac{B+B^\top}{2}\right)$.

⁷For the Max-Cut SDP for example, $\mathcal{A} = \text{diag}$ and $\mu = \text{diag}(C Y Y^\top)$.

⁸Eq. (13) is equivalent to $G\mu = \mathcal{A}(C Y Y^\top)$, where $G_{ij} = \langle A_i Y, A_j Y \rangle$. For all $Y \in \mathcal{M}$, G has constant rank equal to the codimension of \mathcal{M} . Hence, $\mu = G^\dagger \mathcal{A}(C Y Y^\top)$ is differentiable in Y at all $Y \in \mathcal{M}$ (G^\dagger is the pseudoinverse of G).

The second-order condition for Y is that $\text{Hess}f(Y)$ be positive semidefinite on $T_Y\mathcal{M}$. Using that Proj_Y is a self-adjoint operator, it follows that this condition is equivalent to:

$$\forall \dot{Y} \in T_Y\mathcal{M}, \quad \frac{1}{2} \langle \dot{Y}, \text{Hess}f(Y)[\dot{Y}] \rangle = \langle \dot{Y}, S\dot{Y} \rangle \geq 0. \quad (17)$$

We now show that *rank-deficient* second-order critical points are globally optimal. We obtain this result as a corollary to a more informative statement about optimality gap at approximately second-order critical points (assuming exact rank deficiency). The lemmas also show how S can be used to control the optimality gap at approximate critical points without requiring rank deficiency. This is valid for any p and any C .

Lemma 6. *For any Y on the manifold \mathcal{M} , if $\|\text{grad}f(Y)\| \leq \varepsilon_g$ and $S(Y) \succeq -\frac{\varepsilon_H}{2}I_n$, then the optimality gap at Y with respect to (SDP) is bounded as*

$$0 \leq 2(f(Y) - f^*) \leq \varepsilon_H R + \varepsilon_g \sqrt{R}, \quad (18)$$

where $R = \max_{X \in \mathcal{C}} \text{Tr}(X) < \infty$ measures the size of the compact set \mathcal{C} (6). If $I_n \in \text{im}(\mathcal{A}^*)$, the right hand side of (18) simplifies to $\varepsilon_H R$. This holds in particular if all $X \in \mathcal{C}$ have same trace and \mathcal{C} has a relative interior point (Slater condition).

Proof. By assumption on $S(Y)$ (eq. (14)),

$$\forall \tilde{X} \in \mathcal{C}, \quad -\frac{\varepsilon_H}{2} \text{Tr}(\tilde{X}) \leq \langle S(Y), \tilde{X} \rangle = \langle C, \tilde{X} \rangle - \langle \mathcal{A}^*(\mu(Y)), \tilde{X} \rangle = \langle C, \tilde{X} \rangle - \langle \mu(Y), b \rangle.$$

This holds in particular for \tilde{X} optimal for (SDP). Thus, we may set $\langle C, \tilde{X} \rangle = f^*$; and certainly, $\text{Tr}(\tilde{X}) \leq R$. Furthermore,

$$\langle \mu(Y), b \rangle = \langle \mu(Y), \mathcal{A}(YY^\top) \rangle = \langle C - S(Y), YY^\top \rangle = f(Y) - \langle S(Y)Y, Y \rangle.$$

Combining the typeset equations and using $\text{grad}f(Y) = 2S(Y)Y$, we find

$$0 \leq 2(f(Y) - f^*) \leq \varepsilon_H R + \langle \text{grad}f(Y), Y \rangle. \quad (19)$$

In general, we do not assume $I_n \in \text{im}(\mathcal{A}^*)$ and we get the result by Cauchy–Schwarz on (19) and $\|Y\| = \sqrt{\text{Tr}(YY^\top)} \leq \sqrt{R}$:

$$0 \leq 2(f(Y) - f^*) \leq \varepsilon_H R + \varepsilon_g \sqrt{R}.$$

But if $I_n \in \text{im}(\mathcal{A}^*)$, then we show that Y is a normal vector at Y , so that it is orthogonal to $\text{grad}f(Y)$. Formally: there exists $\nu \in \mathbb{R}^m$ such that $I_n = \mathcal{A}^*(\nu)$, and

$$\langle \text{grad}f(Y), Y \rangle = \langle \text{grad}f(Y)Y^\top, I_n \rangle = \langle \mathcal{A}(\text{grad}f(Y)Y^\top), \nu \rangle = 0,$$

since $\text{grad}f(Y) \in T_Y\mathcal{M}$ (2). This indeed allows to simplify (19).

To conclude, we show that if \mathcal{C} has a relative interior point X' (that is, $\mathcal{A}(X') = b$ and $X' \succ 0$) and if $\text{Tr}(X)$ is a constant for all X in \mathcal{C} , then $I_n \in \text{im}(\mathcal{A}^*)$. Indeed, $\mathbb{S}^{n \times n} = \text{im}(\mathcal{A}^*) \oplus \ker \mathcal{A}$, so there exist $\nu \in \mathbb{R}^m$ and $M \in \ker \mathcal{A}$ such that $I_n = \mathcal{A}^*(\nu) + M$. Thus, for all X in \mathcal{C} ,

$$0 = \text{Tr}(X - X') = \langle \mathcal{A}^*(\nu) + M, X - X' \rangle = \langle M, X - X' \rangle.$$

This implies that M is orthogonal to all $X - X'$. These span $\ker \mathcal{A}$ since X' is interior. Indeed, for any $H \in \ker \mathcal{A}$, since $X' \succ 0$, there exists $\varepsilon > 0$ such that $X \triangleq X' + \varepsilon H \succeq 0$ and $\mathcal{A}(X) = b$, so that $X \in \mathcal{C}$. Hence, $M \in \ker \mathcal{A}$ is orthogonal to $\ker \mathcal{A}$. Consequently, $M = 0$ and $I_n = \mathcal{A}^*(\nu)$. \square

Lemma 7. *If $Y \in \mathcal{M}$ is column rank deficient and $\text{Hess}f(Y) \succeq -\varepsilon_H \text{Id}$, then $S(Y) \succeq -\frac{\varepsilon_H}{2}I_n$.*

Proof. By assumption, there exists $z \in \mathbb{R}^p$, $\|z\| = 1$ such that $Yz = 0$. Thus, for any $x \in \mathbb{R}^n$, we can form $\dot{Y} = xz^\top$: it is a tangent vector since $Y\dot{Y}^\top = 0$ (2). Then, condition (17) combined with the assumption on $\text{Hess}f(Y)$ tells us

$$-\varepsilon_H \|x\|^2 \leq \langle \dot{Y}, \text{Hess}f(Y)[\dot{Y}] \rangle = 2 \langle \dot{Y}, S\dot{Y} \rangle = 2 \langle xz^\top z x^\top, S \rangle = 2x^\top Sx.$$

This holds for all $x \in \mathbb{R}^n$, hence $S \succeq -\frac{\varepsilon_H}{2}I_n$ as required. \square

Corollary 8. *If $Y \in \mathcal{M}_p$ is a column rank-deficient second-order critical point for (P), then it is optimal for (P) and $X = YY^\top$ is optimal for (SDP). In particular, for $p > n$, all second-order critical points are optimal.*

The first part of this corollary also appears as [Burer and Monteiro, 2003, Prop. 4], where the statement is made about local optima rather than second-order critical points.

At this point, we can already give a short proof of Theorem 4.

Proof of Theorem 4. Since $\tilde{Y}\tilde{Y}^\top = YY^\top$, $S(\tilde{Y}) = S(Y)$; in particular, $f(\tilde{Y}) = f(Y)$ and $\|\text{grad}f(\tilde{Y})\| = \|\text{grad}f(Y)\|$. Since \tilde{Y} has deficient column rank, apply Lemmas 6 and 7. For $p > n$, there is no need to form \tilde{Y} as Y necessarily has deficient column rank. \square

Based on Corollary 8, to establish Theorem 2 it is sufficient to show that, for almost all C , all second-order critical points are rank deficient already for small p . We show that in fact this is true even for first-order critical points. The argument is by dimensionality counting on $\mathbb{S}^{n \times n}$: the set of all possible cost matrices C .

Lemma 9. *Under the assumptions of Theorem 2, for almost all C , all critical points of (P) are rank deficient.*

Proof. Let Y be a critical point for (P). By the first-order condition $S(Y)Y = 0$ (15) and the definition of $S(Y) = C - \mathcal{A}^*(\mu(Y))$ (14), there exists $\mu \in \mathbb{R}^m$ such that

$$\text{rank } Y \leq \text{null}(C - \mathcal{A}^*(\mu)) \leq \max_{\nu \in \mathbb{R}^m} \text{null}(C - \mathcal{A}^*(\nu)), \quad (20)$$

where null denotes the nullity (dimension of the kernel). This first step in the proof is inspired by [Wen and Yin, 2013, Thm. 3]. If the right hand side evaluates to ℓ , then there exists ν such that $M = C - \mathcal{A}^*(\nu)$ and $\text{null}(M) = \ell$. Writing $C = M + \mathcal{A}^*(\nu)$, we find that

$$C \in \mathcal{N}_\ell + \text{im}(\mathcal{A}^*) \quad (21)$$

where the $+$ is a set-sum and \mathcal{N}_ℓ denotes the set of symmetric matrices of size n with nullity ℓ . This set has dimension

$$\dim \mathcal{N}_\ell = \frac{n(n+1)}{2} - \frac{\ell(\ell+1)}{2}, \quad (22)$$

whereas $\dim \text{im}(\mathcal{A}^*) = \text{rank}(\mathcal{A}^*) \leq m$. Assume the right hand side of (20) evaluates to p or more. Then, a fortiori,

$$C \in \bigcup_{\ell=p, \dots, n} \mathcal{N}_\ell + \text{im}(\mathcal{A}^*). \quad (23)$$

The set on the right hand side contains all “bad” C ’s, that is, those for which (20) offers no information about the rank of Y . The dimension of that set is bounded as follows, using that the dimension of a finite union is at most the maximal dimension, and the dimension of a finite sum of sets is at most the sum of the set dimensions:

$$\dim \left(\bigcup_{\ell=p, \dots, n} \mathcal{N}_\ell + \text{im}(\mathcal{A}^*) \right) \leq \dim(\mathcal{N}_p + \text{im}(\mathcal{A}^*)) \leq \frac{n(n+1)}{2} - \frac{p(p+1)}{2} + m.$$

Since $C \in \mathbb{S}^{n \times n}$ lives in a space of dimension $\frac{n(n+1)}{2}$, almost no C verifies (23) if

$$\frac{n(n+1)}{2} - \frac{p(p+1)}{2} + m < \frac{n(n+1)}{2}.$$

Hence, if $\frac{p(p+1)}{2} > m$, then, for almost all C , critical points verify $\text{rank}(Y) < p$. \square

Theorem 2 follows as a corollary of Corollary 8 and Lemma 9.

B Numerical experiments

As an example, we run five different solvers on (Max-Cut SDP) with a collection of graphs used in [Burer and Monteiro, 2003, 2005] known as the Gset.⁹ The solvers are as follows, all run in Matlab. The first three are based on a low-rank factorization while the last two are interior point methods (IPM).

Manopt runs the Riemannian Trust-Region method on (Max-Cut BM), via the Manopt toolbox [Boumal et al., 2014], with $p = \left\lceil \frac{\sqrt{8n+1}}{2} \right\rceil$ and random initialization. The number of inner iterations allowed to solve the trust-region subproblem is 500. The solver returns when $\frac{1}{2}\|\text{grad}f(Y)\| = \|SY\| \leq 10^{-6}$. Code is in Matlab.

Manopt+ runs the same algorithm as above, but with p increasing from 2 to $\left\lceil \frac{\sqrt{8n+1}}{2} \right\rceil$ in 5 steps. The point Y computed at a lower p is appended with columns of i.i.d. random Gaussian variables with standard deviation 10^{-5} and mean 0, then rows are normalized to produce Y_+ : the initial point for the next value of p . The randomization allows to escape near-saddle points (in practice). Code is in Matlab.

SDPLR runs the original Burer–Monteiro algorithm implemented by its authors [Burer and Monteiro, 2003]. Code is in C interfaced through C-mex.

HRVW runs an IPM whose implementation is tailored to (Max-Cut SDP), implemented by its authors [Helmberg et al., 1996]. Code is in Matlab.

CVX runs SDPT3 [Toh et al., 1999] on (Max-Cut SDP) via CVX [CVX, 2012]. Code is in C interfaced through C-mex.

After the solvers return, we project their answers to the feasible set. Manopt and SDPLR return a matrix Y : it is sufficient to normalize each row to ensure $X = YY^\top$ is feasible for (Max-Cut SDP) (for Manopt, this step is not necessary). HRVW and CVX return a symmetric matrix X . We compute its Cholesky factorization $X = RR^\top$ —if X is not positive semidefinite, we first project using an eigenvalue decomposition. Then, each row of R is normalized so that $X = RR^\top$ is feasible for (Max-Cut SDP). Computation time required for these projections is not included in the timings.

We report three metrics for each graph and each solver.

Cut bound: a bound on the maximal cut value (lower is better). If C is the adjacency matrix of the graph and D is the degree matrix, then $L = D - C$ is the Laplacian and $\max_X \frac{1}{4} \langle L, X \rangle$ s.t. $\text{diag}(X) = 1, X \succeq 0$ is a bound on the maximal cut. Using Lemma 6 applied to (Max-Cut SDP), a candidate optimizer X yields a bound $\frac{1}{4} \langle L, X \rangle - \frac{n}{4} \lambda_{\min}(S)$.

$\lambda_{\min}(S)$: by Lemma 6, this is a measure of optimality for X (feasible), where $S = C - \text{diag}(\text{diag}(CX))$. It is nonpositive and must be as close to 0 as possible. We compute it using bisection and the Cholesky factorization to ensure accuracy.

Time: computation time in seconds for the solver to run¹⁰ (this excludes time taken to project the solution to the feasible set and to compute the reported metrics.)

Based on the results reported in Table 1, we make the following main observations: (i) the Manopt approach (optimization on manifolds, also advocated in [Journée et al., 2010]) consistently reaches high accuracy solutions, being often orders of magnitude more accurate than other methods, as judged from $\lambda_{\min}(S)$; (ii) incremental rank solvers (Manopt+ and SDPLR) are often the fastest solvers for large instances; and (iii) the tailored IPM HRVW is faster and typically more accurate than the IPM called by CVX (which is generic software). The latter point hints that one must be careful in dismissing IPM’s based on experiments using generic software, although it remains clear from Table 1 that IPM’s scale poorly compared to the low-rank factorization methods tested here. In particular, CVX runs into memory trouble for the larger problem instances reported.¹¹ To save time, we did not run CVX on the largest graphs.

⁹Downloaded from: <http://web.stanford.edu/~yyye/yyye/Gset/> on June 6, 2016.

¹⁰Matlab R2015a on 2×6 cores processors with hyperthreading, Intel(R) Xeon(R) CPU E5-2640 @ 2.50GHz, 256Gb RAM, Springdale Linux 6.

¹¹On Graph 77, running CVX leads to Matlab error “Number of elements exceeds maximum flint $2^{53} - 1$.”

C Numerical experiments: results

Table 1: Results of the experiments described in Section B.

Graph	Metric	Manopt	Manopt+	SDPLR	HRVW	CVX
Graph 1	Cut bound	12083.2	12083.2	12083.2	12083.2	12083.2
800 nodes	$\lambda_{\min}(S)$	$-3 \cdot 10^{-11}$	$-2 \cdot 10^{-11}$	$-9 \cdot 10^{-6}$	$-2 \cdot 10^{-5}$	$-3 \cdot 10^{-6}$
19176 edges	Time [s]	2.1	3.2	6.6	1.9	35.0
Graph 2	Cut bound	12089.4	12089.4	12089.4	12089.4	12089.4
800 nodes	$\lambda_{\min}(S)$	$-2 \cdot 10^{-10}$	$-8 \cdot 10^{-12}$	$-5 \cdot 10^{-6}$	$-3 \cdot 10^{-5}$	$-7 \cdot 10^{-7}$
19176 edges	Time [s]	1.6	3.1	7.8	2.0	33.7
Graph 3	Cut bound	12084.3	12084.3	12085.5	12084.3	12084.3
800 nodes	$\lambda_{\min}(S)$	$-3 \cdot 10^{-11}$	$-1 \cdot 10^{-11}$	$-6 \cdot 10^{-3}$	$-4 \cdot 10^{-5}$	$-2 \cdot 10^{-6}$
19176 edges	Time [s]	2.1	4.5	9.8	2.0	34.0
Graph 4	Cut bound	12111.5	12111.5	12111.5	12111.5	12111.5
800 nodes	$\lambda_{\min}(S)$	$-2 \cdot 10^{-11}$	$-2 \cdot 10^{-10}$	$-1 \cdot 10^{-5}$	$-3 \cdot 10^{-5}$	$-6 \cdot 10^{-6}$
19176 edges	Time [s]	1.8	3.2	10.6	2.2	33.7
Graph 5	Cut bound	12099.9	12099.9	12099.9	12099.9	12099.9
800 nodes	$\lambda_{\min}(S)$	$-3 \cdot 10^{-12}$	$-8 \cdot 10^{-12}$	$-1 \cdot 10^{-5}$	$-3 \cdot 10^{-5}$	$-1 \cdot 10^{-6}$
19176 edges	Time [s]	1.5	2.5	6.7	2.2	33.7
Graph 6	Cut bound	2656.2	2656.2	2660.8	2656.2	2656.2
800 nodes	$\lambda_{\min}(S)$	$-4 \cdot 10^{-12}$	$-8 \cdot 10^{-12}$	$-2 \cdot 10^{-2}$	$-7 \cdot 10^{-6}$	$-9 \cdot 10^{-6}$
19176 edges	Time [s]	1.4	2.6	5.5	2.4	34.1
Graph 7	Cut bound	2489.3	2489.3	2489.3	2489.3	2489.3
800 nodes	$\lambda_{\min}(S)$	$-2 \cdot 10^{-11}$	$-2 \cdot 10^{-11}$	$-1 \cdot 10^{-5}$	$-9 \cdot 10^{-6}$	$-4 \cdot 10^{-7}$
19176 edges	Time [s]	6.4	2.6	5.9	2.0	35.7
Graph 8	Cut bound	2506.9	2506.9	2506.9	2506.9	2506.9
800 nodes	$\lambda_{\min}(S)$	$-5 \cdot 10^{-12}$	$-9 \cdot 10^{-12}$	$-4 \cdot 10^{-5}$	$-1 \cdot 10^{-5}$	$-1 \cdot 10^{-6}$
19176 edges	Time [s]	1.2	1.8	10.6	2.2	34.0
Graph 9	Cut bound	2528.7	2528.7	2528.7	2528.7	2528.7
800 nodes	$\lambda_{\min}(S)$	$-1 \cdot 10^{-9}$	$-8 \cdot 10^{-12}$	$-8 \cdot 10^{-6}$	$-1 \cdot 10^{-5}$	$-1 \cdot 10^{-6}$
19176 edges	Time [s]	0.9	1.8	5.7	2.4	34.8
Graph 10	Cut bound	2485.1	2485.1	2485.1	2485.1	2485.1
800 nodes	$\lambda_{\min}(S)$	$-5 \cdot 10^{-11}$	$-8 \cdot 10^{-12}$	$-6 \cdot 10^{-6}$	$-8 \cdot 10^{-6}$	$-2 \cdot 10^{-6}$
19176 edges	Time [s]	1.2	1.6	5.3	2.1	33.9
Graph 11	Cut bound	629.2	629.2	629.2	629.2	629.2
800 nodes	$\lambda_{\min}(S)$	$-3 \cdot 10^{-9}$	$-7 \cdot 10^{-12}$	$-5 \cdot 10^{-6}$	$-1 \cdot 10^{-6}$	$-4 \cdot 10^{-8}$
1600 edges	Time [s]	13.6	13.6	3.9	2.0	31.5
Graph 12	Cut bound	623.9	623.9	623.9	623.9	623.9
800 nodes	$\lambda_{\min}(S)$	$-1 \cdot 10^{-10}$	$-4 \cdot 10^{-12}$	$-3 \cdot 10^{-6}$	$-3 \cdot 10^{-6}$	$-9 \cdot 10^{-8}$
1600 edges	Time [s]	8.8	7.3	1.9	2.0	31.7
Graph 13	Cut bound	647.1	647.1	647.1	647.1	647.1
800 nodes	$\lambda_{\min}(S)$	$-1 \cdot 10^{-9}$	$-2 \cdot 10^{-12}$	$-2 \cdot 10^{-6}$	$-2 \cdot 10^{-6}$	$-1 \cdot 10^{-7}$
1600 edges	Time [s]	6.9	6.7	1.3	2.2	31.4
Graph 14	Cut bound	3191.6	3191.6	3191.6	3191.6	3191.6
800 nodes	$\lambda_{\min}(S)$	$-1 \cdot 10^{-10}$	$-3 \cdot 10^{-12}$	$-3 \cdot 10^{-5}$	$-3 \cdot 10^{-5}$	$-1 \cdot 10^{-6}$
4694 edges	Time [s]	1.5	5.3	4.4	2.5	34.1
Graph 15	Cut bound	3171.6	3171.6	3171.6	3171.6	3171.6
800 nodes	$\lambda_{\min}(S)$	$-1 \cdot 10^{-10}$	$-5 \cdot 10^{-12}$	$-6 \cdot 10^{-6}$	$-5 \cdot 10^{-6}$	$-3 \cdot 10^{-7}$
4661 edges	Time [s]	3.4	6.5	5.4	3.2	34.6
Graph 16	Cut bound	3175.0	3175.0	3175.1	3175.0	3175.0
800 nodes	$\lambda_{\min}(S)$	$-9 \cdot 10^{-12}$	$-2 \cdot 10^{-12}$	$-6 \cdot 10^{-4}$	$-1 \cdot 10^{-5}$	$-6 \cdot 10^{-7}$
4672 edges	Time [s]	6.6	6.2	3.8	3.1	34.8
Graph 17	Cut bound	3171.3	3171.3	3171.5	3171.3	3171.3
800 nodes	$\lambda_{\min}(S)$	$-5 \cdot 10^{-12}$	$-2 \cdot 10^{-12}$	$-1 \cdot 10^{-3}$	$-1 \cdot 10^{-5}$	$-1 \cdot 10^{-7}$
4667 edges	Time [s]	6.1	6.3	3.5	2.9	34.5

Graph	Metric	Manopt	Manopt+	SDPLR	HRVW	CVX
Graph 18	Cut bound	1166.0	1166.0	1166.0	1166.0	1166.0
800 nodes	$\lambda_{\min}(S)$	$-4 \cdot 10^{-12}$	$-3 \cdot 10^{-12}$	$-3 \cdot 10^{-6}$	$-4 \cdot 10^{-6}$	$-1 \cdot 10^{-6}$
4694 edges	Time [s]	1.8	2.9	4.2	3.2	35.1
Graph 19	Cut bound	1082.0	1082.0	1082.0	1082.0	1082.0
800 nodes	$\lambda_{\min}(S)$	$-4 \cdot 10^{-10}$	$-4 \cdot 10^{-12}$	$-4 \cdot 10^{-6}$	$-3 \cdot 10^{-6}$	$-8 \cdot 10^{-7}$
4661 edges	Time [s]	1.9	2.8	4.3	3.4	34.5
Graph 20	Cut bound	1111.4	1111.4	1112.1	1111.4	1111.4
800 nodes	$\lambda_{\min}(S)$	$-2 \cdot 10^{-12}$	$-3 \cdot 10^{-12}$	$-3 \cdot 10^{-3}$	$-4 \cdot 10^{-6}$	$-2 \cdot 10^{-6}$
4672 edges	Time [s]	2.8	3.7	2.9	3.6	34.1
Graph 21	Cut bound	1104.3	1104.3	1104.3	1104.3	1104.3
800 nodes	$\lambda_{\min}(S)$	$-2 \cdot 10^{-11}$	$-6 \cdot 10^{-12}$	$-4 \cdot 10^{-6}$	$-2 \cdot 10^{-6}$	$-6 \cdot 10^{-6}$
4667 edges	Time [s]	2.7	4.3	3.5	3.7	34.1
Graph 22	Cut bound	14135.9	14135.9	14136.0	14135.9	14137.2
2000 nodes	$\lambda_{\min}(S)$	$-8 \cdot 10^{-12}$	$-8 \cdot 10^{-12}$	$-3 \cdot 10^{-5}$	$-3 \cdot 10^{-5}$	$-2 \cdot 10^{-3}$
19990 edges	Time [s]	5.5	4.9	22.5	25.7	177.7
Graph 23	Cut bound	14142.1	14142.1	14142.1	14142.1	14143.5
2000 nodes	$\lambda_{\min}(S)$	$-2 \cdot 10^{-11}$	$-3 \cdot 10^{-11}$	$-8 \cdot 10^{-6}$	$-3 \cdot 10^{-5}$	$-3 \cdot 10^{-3}$
19990 edges	Time [s]	7.0	9.1	16.3	23.8	182.8
Graph 24	Cut bound	14140.9	14140.9	14140.9	14140.9	14142.1
2000 nodes	$\lambda_{\min}(S)$	$-1 \cdot 10^{-11}$	$-7 \cdot 10^{-12}$	$-1 \cdot 10^{-5}$	$-2 \cdot 10^{-5}$	$-2 \cdot 10^{-3}$
19990 edges	Time [s]	4.5	5.7	24.3	24.8	173.3
Graph 25	Cut bound	14144.2	14144.2	14148.8	14144.2	14145.8
2000 nodes	$\lambda_{\min}(S)$	$-1 \cdot 10^{-9}$	$-9 \cdot 10^{-12}$	$-9 \cdot 10^{-3}$	$-9 \cdot 10^{-6}$	$-3 \cdot 10^{-3}$
19990 edges	Time [s]	4.8	18.1	16.7	23.8	175.0
Graph 26	Cut bound	14132.9	14132.9	14132.9	14132.9	14134.2
2000 nodes	$\lambda_{\min}(S)$	$-7 \cdot 10^{-12}$	$-1 \cdot 10^{-11}$	$-4 \cdot 10^{-6}$	$-2 \cdot 10^{-5}$	$-3 \cdot 10^{-3}$
19990 edges	Time [s]	6.8	6.5	14.4	23.1	177.6
Graph 27	Cut bound	4141.7	4141.7	4145.0	4141.7	4143.1
2000 nodes	$\lambda_{\min}(S)$	$-1 \cdot 10^{-11}$	$-7 \cdot 10^{-12}$	$-7 \cdot 10^{-3}$	$-9 \cdot 10^{-6}$	$-3 \cdot 10^{-3}$
19990 edges	Time [s]	3.7	4.4	10.8	23.5	175.9
Graph 28	Cut bound	4100.8	4100.8	4100.8	4100.8	4102.2
2000 nodes	$\lambda_{\min}(S)$	$-2 \cdot 10^{-9}$	$-6 \cdot 10^{-12}$	$-3 \cdot 10^{-5}$	$-7 \cdot 10^{-6}$	$-3 \cdot 10^{-3}$
19990 edges	Time [s]	3.0	8.0	19.6	26.5	176.8
Graph 29	Cut bound	4208.9	4208.9	4208.9	4208.9	4210.0
2000 nodes	$\lambda_{\min}(S)$	$-2 \cdot 10^{-11}$	$-2 \cdot 10^{-11}$	$-5 \cdot 10^{-6}$	$-2 \cdot 10^{-6}$	$-2 \cdot 10^{-3}$
19990 edges	Time [s]	12.2	8.3	17.7	24.5	180.6
Graph 30	Cut bound	4215.4	4215.4	4215.4	4215.4	4216.6
2000 nodes	$\lambda_{\min}(S)$	$-7 \cdot 10^{-11}$	$-6 \cdot 10^{-12}$	$-5 \cdot 10^{-6}$	$-6 \cdot 10^{-6}$	$-2 \cdot 10^{-3}$
19990 edges	Time [s]	19.8	10.5	11.6	25.2	176.7
Graph 31	Cut bound	4116.7	4116.7	4119.1	4116.7	4118.0
2000 nodes	$\lambda_{\min}(S)$	$-2 \cdot 10^{-11}$	$-5 \cdot 10^{-12}$	$-5 \cdot 10^{-3}$	$-7 \cdot 10^{-6}$	$-3 \cdot 10^{-3}$
19990 edges	Time [s]	4.1	8.9	16.2	26.2	170.6
Graph 32	Cut bound	1567.6	1567.6	1567.6	1567.6	1567.8
2000 nodes	$\lambda_{\min}(S)$	$-2 \cdot 10^{-10}$	$-8 \cdot 10^{-12}$	$-1 \cdot 10^{-6}$	$-1 \cdot 10^{-6}$	$-3 \cdot 10^{-4}$
4000 edges	Time [s]	45.6	25.4	13.9	21.7	142.6
Graph 33	Cut bound	1544.3	1544.3	1544.3	1544.3	1544.4
2000 nodes	$\lambda_{\min}(S)$	$-7 \cdot 10^{-10}$	$-5 \cdot 10^{-12}$	$-1 \cdot 10^{-6}$	$-9 \cdot 10^{-7}$	$-1 \cdot 10^{-4}$
4000 edges	Time [s]	31.2	17.3	9.9	23.0	141.2
Graph 34	Cut bound	1546.7	1546.7	1546.7	1546.7	1546.8
2000 nodes	$\lambda_{\min}(S)$	$-1 \cdot 10^{-9}$	$-5 \cdot 10^{-12}$	$-2 \cdot 10^{-6}$	$-1 \cdot 10^{-6}$	$-2 \cdot 10^{-4}$
4000 edges	Time [s]	31.3	22.0	7.7	23.6	143.9
Graph 35	Cut bound	8014.7	8014.7	8014.7	8014.7	8015.3
2000 nodes	$\lambda_{\min}(S)$	$-1 \cdot 10^{-9}$	$-4 \cdot 10^{-11}$	$-5 \cdot 10^{-6}$	$-9 \cdot 10^{-6}$	$-1 \cdot 10^{-3}$
11778 edges	Time [s]	19.4	17.4	26.0	34.5	187.7

Graph	Metric	Manopt	Manopt+	SDPLR	HRVW	CVX
Graph 36	Cut bound	8006.0	8006.0	8006.0	8006.0	8006.6
2000 nodes	$\lambda_{\min}(S)$	$-9 \cdot 10^{-10}$	$-3 \cdot 10^{-11}$	$-1 \cdot 10^{-5}$	$-2 \cdot 10^{-5}$	$-1 \cdot 10^{-3}$
11766 edges	Time [s]	12.0	36.9	41.1	37.0	193.3
Graph 37	Cut bound	8018.6	8018.6	8019.4	8018.6	8019.5
2000 nodes	$\lambda_{\min}(S)$	$-2 \cdot 10^{-10}$	$-1 \cdot 10^{-11}$	$-1 \cdot 10^{-3}$	$-1 \cdot 10^{-5}$	$-2 \cdot 10^{-3}$
11785 edges	Time [s]	11.2	15.4	38.4	35.2	191.1
Graph 38	Cut bound	8015.0	8015.0	8015.0	8015.0	8015.5
2000 nodes	$\lambda_{\min}(S)$	$-1 \cdot 10^{-10}$	$-1 \cdot 10^{-11}$	$-2 \cdot 10^{-5}$	$-1 \cdot 10^{-5}$	$-1 \cdot 10^{-3}$
11779 edges	Time [s]	13.1	14.2	44.7	37.5	193.0
Graph 39	Cut bound	2877.6	2877.6	2877.8	2877.6	2878.4
2000 nodes	$\lambda_{\min}(S)$	$-4 \cdot 10^{-9}$	$-7 \cdot 10^{-12}$	$-3 \cdot 10^{-4}$	$-4 \cdot 10^{-6}$	$-2 \cdot 10^{-3}$
11778 edges	Time [s]	16.9	12.2	31.9	39.3	195.8
Graph 40	Cut bound	2864.8	2864.8	2866.2	2864.8	2865.6
2000 nodes	$\lambda_{\min}(S)$	$-1 \cdot 10^{-11}$	$-2 \cdot 10^{-11}$	$-3 \cdot 10^{-3}$	$-3 \cdot 10^{-6}$	$-2 \cdot 10^{-3}$
11766 edges	Time [s]	9.2	9.4	40.8	40.9	189.0
Graph 41	Cut bound	2865.2	2865.2	2868.1	2865.2	2865.8
2000 nodes	$\lambda_{\min}(S)$	$-4 \cdot 10^{-10}$	$-1 \cdot 10^{-11}$	$-6 \cdot 10^{-3}$	$-4 \cdot 10^{-6}$	$-1 \cdot 10^{-3}$
11785 edges	Time [s]	5.3	8.6	30.8	40.9	189.8
Graph 42	Cut bound	2946.3	2946.3	2948.3	2946.3	2947.0
2000 nodes	$\lambda_{\min}(S)$	$-9 \cdot 10^{-12}$	$-7 \cdot 10^{-12}$	$-4 \cdot 10^{-3}$	$-6 \cdot 10^{-6}$	$-1 \cdot 10^{-3}$
11779 edges	Time [s]	7.9	8.1	32.9	41.8	188.4
Graph 43	Cut bound	7032.2	7032.2	7032.2	7032.2	7033.2
1000 nodes	$\lambda_{\min}(S)$	$-3 \cdot 10^{-12}$	$-4 \cdot 10^{-12}$	$-6 \cdot 10^{-6}$	$-2 \cdot 10^{-5}$	$-4 \cdot 10^{-3}$
9990 edges	Time [s]	1.9	2.3	3.6	3.8	36.4
Graph 44	Cut bound	7027.9	7027.9	7029.2	7027.9	7029.4
1000 nodes	$\lambda_{\min}(S)$	$-1 \cdot 10^{-8}$	$-3 \cdot 10^{-12}$	$-5 \cdot 10^{-3}$	$-2 \cdot 10^{-5}$	$-6 \cdot 10^{-3}$
9990 edges	Time [s]	2.9	3.9	3.7	3.6	38.0
Graph 45	Cut bound	7024.8	7024.8	7024.8	7024.8	7025.9
1000 nodes	$\lambda_{\min}(S)$	$-1 \cdot 10^{-9}$	$-5 \cdot 10^{-12}$	$-2 \cdot 10^{-5}$	$-8 \cdot 10^{-6}$	$-5 \cdot 10^{-3}$
9990 edges	Time [s]	1.3	6.1	4.9	3.5	37.4
Graph 46	Cut bound	7029.9	7029.9	7029.9	7029.9	7030.8
1000 nodes	$\lambda_{\min}(S)$	$-2 \cdot 10^{-10}$	$-3 \cdot 10^{-12}$	$-2 \cdot 10^{-5}$	$-1 \cdot 10^{-5}$	$-4 \cdot 10^{-3}$
9990 edges	Time [s]	12.9	2.3	3.1	3.7	38.3
Graph 47	Cut bound	7036.7	7036.7	7036.7	7036.7	7037.8
1000 nodes	$\lambda_{\min}(S)$	$-8 \cdot 10^{-10}$	$-9 \cdot 10^{-12}$	$-1 \cdot 10^{-5}$	$-1 \cdot 10^{-5}$	$-5 \cdot 10^{-3}$
9990 edges	Time [s]	10.4	4.1	8.2	3.8	39.2
Graph 48	Cut bound	6000.0	6000.0	6000.0	6000.0	6000.0
3000 nodes	$\lambda_{\min}(S)$	$4 \cdot 10^{-16}$	$3 \cdot 10^{-16}$	$-6 \cdot 10^{-10}$	$-3 \cdot 10^{-6}$	$5 \cdot 10^{-18}$
6000 edges	Time [s]	2.8	4.3	3.5	47.7	307.3
Graph 49	Cut bound	6000.0	6000.0	6000.0	6000.0	6000.0
3000 nodes	$\lambda_{\min}(S)$	$4 \cdot 10^{-16}$	$4 \cdot 10^{-16}$	$-1 \cdot 10^{-9}$	$-3 \cdot 10^{-6}$	$-4 \cdot 10^{-16}$
6000 edges	Time [s]	3.9	5.1	4.9	46.1	299.7
Graph 50	Cut bound	5988.2	5988.2	5988.2	5988.2	5988.2
3000 nodes	$\lambda_{\min}(S)$	$-2 \cdot 10^{-12}$	$-1 \cdot 10^{-14}$	$-1 \cdot 10^{-7}$	$-3 \cdot 10^{-6}$	$2 \cdot 10^{-16}$
6000 edges	Time [s]	6.0	5.0	5.4	45.7	318.4
Graph 51	Cut bound	4006.3	4006.3	4006.3	4006.3	4006.9
1000 nodes	$\lambda_{\min}(S)$	$-2 \cdot 10^{-9}$	$-4 \cdot 10^{-12}$	$-8 \cdot 10^{-6}$	$-1 \cdot 10^{-5}$	$-3 \cdot 10^{-3}$
5909 edges	Time [s]	5.8	7.8	10.7	5.4	41.4
Graph 52	Cut bound	4009.6	4009.6	4010.0	4009.6	4010.2
1000 nodes	$\lambda_{\min}(S)$	$-4 \cdot 10^{-12}$	$-9 \cdot 10^{-12}$	$-1 \cdot 10^{-3}$	$-5 \cdot 10^{-6}$	$-2 \cdot 10^{-3}$
5916 edges	Time [s]	6.4	8.8	6.5	5.2	39.6
Graph 53	Cut bound	4009.7	4009.7	4009.7	4009.7	4010.5
1000 nodes	$\lambda_{\min}(S)$	$-1 \cdot 10^{-10}$	$-1 \cdot 10^{-11}$	$-6 \cdot 10^{-6}$	$-1 \cdot 10^{-5}$	$-3 \cdot 10^{-3}$
5914 edges	Time [s]	4.2	8.5	8.3	5.0	39.1

Graph	Metric	Manopt	Manopt+	SDPLR	HRVW	CVX
Graph 54	Cut bound	4006.2	4006.2	4006.2	4006.2	4006.9
1000 nodes	$\lambda_{\min}(S)$	$-2 \cdot 10^{-10}$	$-3 \cdot 10^{-12}$	$-3 \cdot 10^{-5}$	$-5 \cdot 10^{-6}$	$-3 \cdot 10^{-3}$
5916 edges	Time [s]	2.9	6.6	6.1	4.8	39.1
Graph 55	Cut bound	11039.5	11039.5	11039.5	11039.5	11039.7
5000 nodes	$\lambda_{\min}(S)$	$-2 \cdot 10^{-12}$	$-3 \cdot 10^{-12}$	$-5 \cdot 10^{-6}$	$-6 \cdot 10^{-6}$	$-2 \cdot 10^{-4}$
12498 edges	Time [s]	26.6	20.6	22.2	411.4	1588.0
Graph 56	Cut bound	4760.0	4760.0	4760.0	4760.0	4760.3
5000 nodes	$\lambda_{\min}(S)$	$-7 \cdot 10^{-12}$	$-2 \cdot 10^{-12}$	$-1 \cdot 10^{-5}$	$-2 \cdot 10^{-6}$	$-3 \cdot 10^{-4}$
12498 edges	Time [s]	20.1	16.3	32.9	475.9	1550.1
Graph 57	Cut bound	3885.5	3885.5	3885.5	3885.5	3885.7
5000 nodes	$\lambda_{\min}(S)$	$-1 \cdot 10^{-9}$	$-8 \cdot 10^{-12}$	$-2 \cdot 10^{-6}$	$-2 \cdot 10^{-6}$	$-1 \cdot 10^{-4}$
10000 edges	Time [s]	218.0	78.8	38.3	269.8	1012.4
Graph 58	Cut bound	20136.2	20136.2	20138.1	20136.2	20136.7
5000 nodes	$\lambda_{\min}(S)$	$-3 \cdot 10^{-9}$	$-5 \cdot 10^{-11}$	$-2 \cdot 10^{-3}$	$-7 \cdot 10^{-6}$	$-4 \cdot 10^{-4}$
29570 edges	Time [s]	55.4	44.0	321.5	497.9	1865.7
Graph 59	Cut bound	7312.3	7312.3	7315.0	7312.3	7313.0
5000 nodes	$\lambda_{\min}(S)$	$-7 \cdot 10^{-12}$	$-3 \cdot 10^{-11}$	$-2 \cdot 10^{-3}$	$-4 \cdot 10^{-6}$	$-5 \cdot 10^{-4}$
29570 edges	Time [s]	51.3	35.6	353.1	511.3	1869.0
Graph 60	Cut bound	15222.3	15222.3	15222.3	15222.3	15222.6
7000 nodes	$\lambda_{\min}(S)$	$-3 \cdot 10^{-11}$	$-4 \cdot 10^{-12}$	$-2 \cdot 10^{-5}$	$-2 \cdot 10^{-6}$	$-2 \cdot 10^{-4}$
17148 edges	Time [s]	58.6	30.9	63.6	1326.9	3581.9
Graph 61	Cut bound	6828.1	6828.1	6828.2	6828.1	6828.4
7000 nodes	$\lambda_{\min}(S)$	$-2 \cdot 10^{-11}$	$-4 \cdot 10^{-12}$	$-7 \cdot 10^{-5}$	$-2 \cdot 10^{-6}$	$-2 \cdot 10^{-4}$
17148 edges	Time [s]	113.4	40.2	55.8	1263.3	3795.6
Graph 62	Cut bound	5430.9	5430.9	5430.9	5430.9	5431.1
7000 nodes	$\lambda_{\min}(S)$	$-1 \cdot 10^{-9}$	$-6 \cdot 10^{-11}$	$-9 \cdot 10^{-7}$	$-2 \cdot 10^{-6}$	$-1 \cdot 10^{-4}$
14000 edges	Time [s]	813.8	242.8	110.8	862.4	2124.3
Graph 63	Cut bound	28244.4	28244.4	28245.9	28244.4	28245.0
7000 nodes	$\lambda_{\min}(S)$	$-7 \cdot 10^{-9}$	$-8 \cdot 10^{-9}$	$-8 \cdot 10^{-4}$	$-9 \cdot 10^{-6}$	$-3 \cdot 10^{-4}$
41459 edges	Time [s]	238.9	97.6	663.0	1454.7	4583.9
Graph 64	Cut bound	10465.9	10465.9	10466.6	10465.9	10466.6
7000 nodes	$\lambda_{\min}(S)$	$-3 \cdot 10^{-9}$	$-2 \cdot 10^{-11}$	$-4 \cdot 10^{-4}$	$-5 \cdot 10^{-6}$	$-4 \cdot 10^{-4}$
41459 edges	Time [s]	140.4	109.5	1014.8	1609.4	4439.8
Graph 65	Cut bound	6205.5	6205.5	6205.5	6205.5	6205.7
8000 nodes	$\lambda_{\min}(S)$	$-1 \cdot 10^{-9}$	$-1 \cdot 10^{-11}$	$-6 \cdot 10^{-7}$	$-1 \cdot 10^{-6}$	$-1 \cdot 10^{-4}$
16000 edges	Time [s]	567.2	168.5	154.4	1075.2	2861.5
Graph 66	Cut bound	7077.2	7077.2	7077.2	7077.2	7077.4
9000 nodes	$\lambda_{\min}(S)$	$-2 \cdot 10^{-9}$	$-5 \cdot 10^{-11}$	$-2 \cdot 10^{-7}$	$-9 \cdot 10^{-7}$	$-6 \cdot 10^{-5}$
18000 edges	Time [s]	762.6	215.3	218.1	1525.7	3915.7
Graph 67	Cut bound	7744.4	7744.4	7744.4	7744.4	-
10000 nodes	$\lambda_{\min}(S)$	$-1 \cdot 10^{-9}$	$-3 \cdot 10^{-11}$	$-3 \cdot 10^{-7}$	$-1 \cdot 10^{-6}$	-
20000 edges	Time [s]	816.4	339.0	267.3	2005.4	-
Graph 70	Cut bound	9861.5	9861.5	9861.5	9861.5	-
10000 nodes	$\lambda_{\min}(S)$	$-2 \cdot 10^{-10}$	$-6 \cdot 10^{-13}$	$-2 \cdot 10^{-6}$	$-2 \cdot 10^{-6}$	-
9999 edges	Time [s]	143.3	82.9	102.2	3167.3	-
Graph 72	Cut bound	7808.5	7808.5	7808.5	7808.5	-
10000 nodes	$\lambda_{\min}(S)$	$-6 \cdot 10^{-10}$	$-8 \cdot 10^{-12}$	$-8 \cdot 10^{-7}$	$-1 \cdot 10^{-6}$	-
20000 edges	Time [s]	720.8	262.6	199.0	1902.7	-
Graph 77	Cut bound	11045.7	11045.7	11045.7	11045.7	-
14000 nodes	$\lambda_{\min}(S)$	$-8 \cdot 10^{-10}$	$-4 \cdot 10^{-11}$	$-7 \cdot 10^{-7}$	$-1 \cdot 10^{-6}$	-
28000 edges	Time [s]	1578.5	513.0	515.1	5249.1	-
Graph 81	Cut bound	15656.2	15656.2	15656.2	15656.2	-
20000 nodes	$\lambda_{\min}(S)$	$-5 \cdot 10^{-10}$	$-6 \cdot 10^{-11}$	$-1 \cdot 10^{-6}$	$-3 \cdot 10^{-6}$	-
40000 edges	Time [s]	4152.8	1539.7	1035.6	16576.6	-