AN OPTIMAL-STORAGE APPROACH TO SEMIDEFINITE PROGRAMMING USING APPROXIMATE COMPLEMENTARITY

LIJUN DING*, ALP YURTSEVER†, VOLKAN CEVHER†, JOEL A. TROPP‡, AND MADELEINE UDELL*

Abstract. This paper develops a new storage-optimal algorithm that provably solves generic semidefinite programs (SDPs) in standard form. This method is particularly effective for weakly constrained SDPs. The key idea is to formulate an approximate complementarity principle: Given an approximate solution to the dual SDP, the primal SDP has an approximate solution whose range is contained in the eigenspace with small eigenvalues of the dual slack matrix. For weakly constrained SDPs, this eigenspace has very low dimension, so this observation significantly reduces the search space for the primal solution. This result suggests an algorithmic strategy that can be implemented with minimal storage: (1) Solve the dual SDP approximately; (2) compress the primal SDP to the eigenspace with small eigenvalues of the dual slack matrix; (3) solve the compressed primal SDP. The paper also provides numerical experiments showing that this approach is successful for a range of interesting large-scale SDPs.

Key words. Semidefinite programs, Storage-optimality, Low rank, Complementary slackness, Primal recovery.

AMS subject classifications. 90C06, 90C2, 49M05

1. Introduction.

Consider a semidefinite program (SDP) in the standard form

$$\text{(P)} \qquad \begin{array}{ll} \text{minimize} & \mathbf{tr}(CX) \\ \text{subject to} & \mathcal{A}X = b \quad \text{and} \quad X \succeq 0. \end{array}$$

The primal variable is the symmetric, positive-semidefinite matrix $X \in \mathbf{S}_+^n$. The problem data comprises a symmetric (but possibly indefinite) objective matrix $C \in \mathbf{S}^n$, a linear map $\mathcal{A} : \mathbf{R}^{n\times n} \to \mathbf{R}^m$ with rank m, and a righthand side $b \in \mathbf{R}^m$.

SDPs form a class of convex optimization problems with remarkable modeling power. But SDPs are challenging to solve because they involve a matrix variable $X \in \mathbf{S}_+^n \subset \mathbf{R}^{n\times n}$ whose dimension n can rise into the millions or billions. For example, when using a matrix completion SDP in a recommender system, n is the number of users and products; when using a phase retrieval SDP to visualize a biological sample, n is the number of pixels in the recovered image. In these applications, most algorithms are prohibitively expensive because their storage costs are quadratic in n.

How much memory should be required to solve this problem? Any algorithm must be able to query the problem data and to report a representation of the solution. Informally, we say that an algorithm uses *optimal storage* if the working storage is no more than a constant multiple of the storage required for these operations [72]. (See Subsection 1.2 for a formal definition.)

It is not obvious how to develop storage-optimal SDP algorithms. To see why, recall that all *weakly-constrained* SDPs ($m = O(n)$) admit low-rank solutions [8, 54], which can be expressed compactly in factored form. For these problems, a storage-optimal algorithm cannot even instantiate the matrix variable! One natural idea is to

*Operations Research and Information Engineering, Cornell University, Ithaca, NY, USA (ld446@cornell.edu, udell@cornell.edu).

†Laboratory for Information and Inference Systems (LIONS), EPFL, Lausanne, Vaud, Switzerland (alp.yurtsever@epfl.ch, volkan.cevher@epfl.ch).

‡Computing and Mathematical Sciences, California Institute of Technology, Pasadena, CA, USA (jtropp@cms.caltech.edu).

introduce an explicit low rank factorization of the primal variable X and to minimize the problem over the factors [17].

Methods built from this idea provably work when the size of the factors is sufficiently large [13]. However, recent work [67] shows that they cannot provably solve all SDPs with optimal storage; see Section 2.

In contrast, this paper develops a new algorithm that provably solves all *generic* SDPs using optimal storage. More precisely, our algorithm works when the primal and dual SDPs exhibit strong duality, the primal and dual solutions are unique, and strict complementarity holds; these standard conditions hold generically [4].

Our method begins with the Lagrange dual of the primal SDP (P),

$$\text{(D)} \qquad \begin{array}{ll} \text{maximize} & b^* y \\ \text{subject to} & C - \mathcal{A}^* y \succeq 0 \end{array}$$

with dual variable $y \in \mathbf{R}^m$. The vector b^* is the transpose of b, and the linear map $\mathcal{A}^* : \mathbf{R}^m \to \mathbf{R}^{n \times n}$ is the adjoint of $\mathcal{A}$. It is straightforward to compute an approximate solution to the dual SDP (D) with optimal storage. The challenge is to recover a primal solution from the approximate dual solution.

To meet this challenge, we develop a new *approximate complementarity principle* that holds for generic SDP: Given an approximate dual solution y, we prove that there is a primal approximate solution X whose range is contained in the eigenspace with small eigenvalues of the dual slack matrix $C - \mathcal{A}^* y$. This principle suggests an algorithm: we solve the primal SDP by searching over matrices with the appropriate range. This recovery problem is a (much smaller) SDP that can be solved with optimal storage.

1.1. Assumptions. First, assume that the primal (P) has a solution, say, $X_\star$ and the dual (D) has a *unique* solution $y_\star$. We require that strong duality holds:

$$p_\star := \mathbf{tr}(C X_\star) = b^* y_\star =: d_\star. \tag{1.1}$$

The condition (1.1) follows, for example, from Slater's constraint qualification.

Strong duality and feasibility imply that the solution $X_\star$ and the dual slack matrix $C - \mathcal{A}^* y_\star$ satisfy the complementary slackness condition:

$$X_\star (C - \mathcal{A}^* y_\star) = 0. \tag{1.2}$$

These conditions suffice for the results in Section 5 (and the associated lemmas Lemma 4.2 and Lemma 4.3), which bound the recovery error of our procedure if the linear map $\mathcal{A}$ restricted to a certain subspace is nonsingular (see (3.1) and Algorithm 5.1 for the choice of the subspace).

To ensure this condition, we make the stronger assumption that every solution pair $(X_\star, y_\star)$ satisfies the stronger *strict complementarity* condition:

$$\mathbf{rank}(X_\star) + \mathbf{rank}(C - \mathcal{A}^* y_\star) = n. \tag{1.3}$$

Note that these assumptions ensure that *all* solutions have the same rank, and therefore that the primal solution is also *unique* [42, Corollary 2.5]. In particular, the rank $r_\star$ of the solution $X_\star$ satisfies the Barvinok–Pataki bound $\binom{r_\star+1}{2} \le m$.

To summarize, all results in this paper hold under the assumptions of primal attainability, dual uniqueness, strong duality, and strict complementarity. These conditions hold generically conditioning on primal and dual attainability; i.e., for every SDP satisfying primal and dual attainability outside of a set of measure 0 [4].

1.2. Optimal Storage. Following [72], let us quantify the storage necessary to solve every SDP (P) that satisfies our assumptions in Subsection 1.1 and that admits a solution with rank $r_\star$.

First, it is easy to see that $\Theta(nr_\star)$ numbers are sufficient to represent the rank-$r_\star$ solution in factored form. This cost is also necessary because *every* rank-$r_\star$ matrix is the solution to some SDP from our problem class.

To hide the internal complexity of the optimization problem (P), we will interact with the problem data using *data access oracles*. Suppose we can perform any of the following operations on arbitrary vectors $u, v \in \mathbf{R}^n$ and $y \in \mathbf{R}^m$:

$$u \mapsto Cu \quad \text{and} \quad (u, v) \mapsto \mathcal{A}(uv^*) \quad \text{and} \quad (u, y) \mapsto (\mathcal{A}^* y)u. \tag{1.4}$$

These oracles enjoy simple implementations in many concrete applications. The input and output of these operations clearly involve storing $\Theta(m + n)$ numbers.

In summary, any method that uses these data access oracles to solve every SDP from our class must store $\Omega(m + nr_\star)$ numbers. We say a method has *optimal storage* if the working storage provably achieves this bound.

For many interesting problems, the number m of constraints is proportional to the dimension n. Moreover, the rank $r_\star$ of the solution is constant or logarithmic in n. In this case, a storage-optimal algorithm has working storage $\tilde{\mathcal{O}}(n)$, where the tilde suppresses log-like factors.

Remark 1.1 (Applications). The algorithmic framework we propose is most useful when the problem data has an efficient representation and the three operations in (1.4) can be implemented with low arithmetic cost. For example, it is often the case that the matrix C and the linear map $\mathcal{A}$ are sparse or structured. This situation occurs in the maxcut relaxation [32], matrix completion [61], phase retrieval [22, 66], and community detection [47]. See [56] for some other examples.

1.3. From Strict Complementarity to Storage Optimality. Suppose that we have computed the *exact* unique dual solution $y_\star$. Complementary slackness (1.2) and strict complementarity (1.3) ensure that

$$\mathbf{range}(X_\star) \subset \mathbf{null}(C - \mathcal{A}^* y_\star) \quad \text{and} \quad \dim(\mathbf{null}(C - \mathcal{A}^* y_\star)) = \mathbf{rank}(X_\star).$$

Therefore, the slack matrix identifies the range of the primal solution.

Let $r_\star$ be the rank of the primal solution. Construct an orthonormal matrix $V_\star \in \mathbf{R}^{n \times r_\star}$ whose columns span $\mathbf{null}(C - \mathcal{A}^* y_\star)$. The compression of the primal problem (P) to this subspace is

$$\begin{array}{ll} \text{minimize} & \mathbf{tr}(CV_\star S V_\star{}^*) \\ \text{subject to} & \mathcal{A}(V_\star S V_\star{}^*) = b \quad \text{and} \quad S \succeq 0. \end{array} \tag{1.5}$$

The variable $S \in \mathbf{S}_+^{r_\star}$ is a low-dimensional matrix when $r_\star$ is small. If $S_\star$ is a solution to (1.5), then $X_\star = V_\star S_\star V_\star{}^*$ is a solution to the original SDP (P).

This strategy for solving the primal SDP can be implemented with a storage-optimal algorithm. Indeed, the variable y in the dual SDP (D) has length m, so there is no obstacle to solving the dual with storage $\Theta(m + n)$ using the subgradient type method described in Section 6. We can compute the subspace $V_\star$ using the randomized range finder [33, Alg. 4.1] with storage cost $\Theta(nr_\star)$. Last, we can solve the compressed primal SDP (1.5) using working storage $\Theta(m + n + r_\star^2)$ via the matrix-free method from [25, 52]. The total storage is the optimal $\Theta(m + nr_\star)$. Furthermore, all of these algorithms can be implemented with the data access oracles (1.4).

Hence — assuming exact solutions to the optimization problems — we have developed a storage-optimal approach to the SDP (P), summarized in Table 1[left].

1.4. The Approximate Complementarity Principle. A major challenge remains: one very rarely has access to an exact dual solution! Rather, we usually have an approximate dual solution, obtained via some iterative dual solver.

This observation motivates us to formulate a new *approximate complementarity principle*. For now, assume that $r_\star$ is known. Given an approximate dual solution y, we can construct an orthonormal matrix $V \in \mathbf{R}^{n\times r_\star}$ whose columns are eigenvectors of $C - \mathcal{A}^*y$ with the $r_\star$ smallest eigenvalues. Roughly speaking, the primal problem (P) admits an *approximate* solution X whose range is contained in $\mathbf{range}(V)$. We show the approximate solution is close to the true solution as measured in terms of suboptimality, infeasibility, and distance to the solution set.

We propose to recover the approximate primal solution by solving the semidefinite least-squares problem

$$\text{(MinFeasSDP)} \qquad \begin{array}{ll} \text{minimize} & \frac{1}{2}\left\|\mathcal{A}(VSV^*) - b\right\|^2 \\ \text{subject to} & S \succeq 0 \end{array}$$

with variable $S \in \mathbf{S}_+^{r_\star}$. Given a solution $\hat{S}$ to (MinFeasSDP), we obtain an (infeasible) approximate solution $X_{\text{infeas}} = V\hat{S}V^*$ to the primal problem.

In fact, it is essential to relax our attention to infeasible solutions because the feasible set of (P) should almost never contains a matrix with range V! This observation was very surprising to us, but it seems evident in retrospect. (For example, using a dimension-counting argument together with Lemma A.1.)

The resulting framework appears in Table 1[right]. This approach for solving (P) leads to storage-optimal algorithms for the same reasons described in Subsection 1.3. Our first main result ensures that this technique results in a provably good solution to the primal SDP (P).

THEOREM 1.2 (Main theorem, informal). *Instate the assumptions of Subsection 1.1. Suppose we have found a dual vector y with suboptimality $\epsilon := d_\star - b^*y \le \text{const}$. Consider the primal reconstruction X_{infeas} obtained by solving (MinFeasSDP). Then we may bound the distance between X_{infeas} to the primal solution $X_\star$ by*

$$\left\|X_{\text{infeas}} - X_\star\right\|_F = \mathcal{O}(\sqrt{\epsilon}).$$

The constant in the $\mathcal{O}$ depends on the problem data $\mathcal{A}$, b, and C.

We state and prove the formal result as Theorem 4.1. As stated, this guarantee requires knowledge of the rank $r^\star$ of the solution; in Section 5, we obtain a similar guarantee using an estimate for $r^\star$.

1.5. Paper Organization. We situate our contributions relative to other work in Section 2. Section 3 contains an overview of our notation and more detailed problem assumptions. Section 4 uses the approximate complementarity principle to develop practical, robust, and theoretically justified algorithms for solving (P). The algorithms are accompanied by detailed bounds on the quality of the computed solutions as compared with the true solution. Section 5 contains algorithmically computable bounds that can be used to check the solution quality under weaker assumptions than those required in Section 4. These checks are important for building a reliable solver. Next, we turn to algorithmic issues: we explain how to compute an approximate dual solution efficiently in Section 6, which provides the last ingredient for a complete method to solve (P). Section 7 shows numerically that the method is effective in practice.

Table 1: Exact and Robust Primal Recovery

Step	**Exact Primal Recovery**	**Robust Primal Recovery**
1	Compute dual solution $y_\star$	Compute approximate dual solution y
2	Compute basis $V_\star$ for $\mathbf{null}(C - \mathcal{A}^* y_\star)$	Compute $r_\star$ eigenvectors of $C - \mathcal{A}^* y$ with smallest eigenvalues; collect as columns of matrix V
3	Solve the reduced SDP (1.5)	Solve (MinFeasSDP)

2. Related Work. Semidefinite programming can be traced to a 1963 paper of Bellman & Fan [10]. Related questions emerged earlier in control theory, starting from Lyapunov's 1890 work on stability of dynamical systems. There are many classic applications in matrix analysis, dating to the 1930s. Graph theory provides another rich source of examples, beginning from the 1970s. See [14, 65, 63, 11, 16] for more history and problem formulations.

2.1. Interior-Point Methods. The first reliable algorithms for semidefinite programming were interior-point methods (IPMs). These techniques were introduced independently by Nesterov & Nemirovski [50, 51] and Alizadeh [2, 3].

The success of these SDP algorithms motivated new applications. In particular, Goemans & Williamson [32] used semidefinite programming to design an approximation algorithm to compute the maximum-weight cut in a graph. Early SDP solvers could only handle graphs with a few hundred vertices [32, Sec. 5] although computational advances quickly led to IPMs that could solve problems with thousands of vertices [12].

IPMs form a series of unconstrained problems whose solutions are feasible for the original SDP, and move towards the solutions of these unconstrained problems using Newton's method. As a result, IPMs converge to high accuracy in very few iterations, but require substantial work per iteration. To solve a standard-form SDP with an $n \times n$ matrix variable and with m equality constraints, a typical IPM requires $\mathcal{O}(\sqrt{n}\log(\frac{1}{\epsilon}))$ iterations to reach a solution with accuracy ϵ (in terms of objective value) [49], and $\mathcal{O}(mn^3 + m^2n^2 + m^3)$ arithmetic operations per iteration (when no structure is concerned)[5], so $\mathcal{O}(\sqrt{n}\log(\frac{1}{\epsilon})(mn^3+m^2n^2+m^3))$ arithmetic operations in total. Further, a typical IPM requires at least $\Theta(n^2 + m + m^2)$ memory not including the storage of data representation (which takes $\Theta(n^2 m)$ memory if no structure is assumed)[5].

As a consequence, these algorithms are not effective for solving large problem instances, unless they enjoy a lot of structure. Hence researchers began to search for methods that could scale to larger problems.

2.2. First-Order Methods. One counterreaction to the expense of IPMs was to develop first-order optimization algorithms for SDPs. This line of work began in the late 1990s, and it accelerated as SDPs emerged in the machine learning and signal processing literature in the 2000s.

Early on, Helmberg & Rendl [35] proposed a spectral bundle method for solving an SDP in dual form, and they showed that it converges to a dual solution when the trace of $X_\star$ is constant. In contrast to IPMs, the spectral bundle method has low per iteration complexity. On the other hand, the convergence rate is not known, and there is no convergence guarantee on the primal side. so there is no explicit control on the storage and arithmetic costs.

In machine learning, popular first-order algorithms include the proximal gradient method [58], accelerated variants [9] and the alternating direction method of multipliers [30, 31, 15, 53]. These methods provably solve the original convex formulation of (P), but they converge slowly in practice. These algorithms all store the full primal matrix variable, so they are not storage-efficient.

Recently, Friedlander & Macedo [29] have proposed a novel first-order method that is based on gauge duality, rather than Lagrangian duality. This approach converts an SDP into an eigenvalue optimization problem. The authors propose a mechanism for using a dual solution to construct a primal solution. This paper is similar in spirit to our approach, but it lacks an analysis of the accuracy of the primal solution. Moreover, it only applies to problems with a positive-definite objective, i.e., $C \succ 0$.

2.3. Storage-Efficient First-Order Methods. Motivated by problems in signal processing and machine learning, a number of authors have revived the conditional gradient method (CGM) [28, 43, 39]. In particular, Hazan [34] suggested using CGM for semidefinite programming. Clarkson [24] developed a new analysis, and Jaggi [37], showed how this algorithm applies to a wide range of interesting problems.

The appeal of the CGM is that it computes an approximate solution to an SDP as a sum of rank-one updates; each rank-one update is obtained from an approximate eigenvector computation. In particular, after t iterations, the iterate has rank at most t. This property has led to the exaggeration that CGM is a "storage-efficient" optimization method. Unfortunately, CGM converges very slowly, so the iterates do not have controlled rank. The literature describes many heuristics for attempting to control the rank of the iterates [55, 71], but these methods all lack guarantees.

Very recently, some of the authors of this paper [72] have shown how to use CGM to design a storage-optimal algorithm for a class of semidefinite programs by sketching the decision variable. This algorithm does not apply to standard-form SDPs, and it inherits the slow convergence of CGM. Nevertheless, the sketching methodology holds promise as a way to design storage optimal solvers, particularly together with algorithms that generalize CGM and that do apply to standard-form SDPs [70, 69].

We also mention a subgradient method developed by Renegar [57] that can be used to solve either the primal or dual SDP. Renegar's method has a computational profile similar to CGM, and it does not have controlled storage costs.

2.4. Factorization Methods. There is also a large class of heuristic SDP algorithms based on matrix factorization. The key idea is to factorize the matrix variable $X = FF^*, F \in \mathbf{R}^{n\times r}$ and to reformulate the SDP (P) as

$$\begin{array}{ll} \text{minimize} & \mathbf{tr}(CFF^*) \\ \text{subject to} & \mathcal{A}(FF^*) = b. \end{array} \tag{2.1}$$

We can apply a wide range of nonlinear programming methods to optimize (2.1) with respect to the variable F. In contrast to the convex methods described above, these techniques only offer incomplete guarantees on storage, arithmetic, and convergence.

The factorization idea originates in the paper [36] of Homer & Peinado. They focused on the MAXCUT SDP, and the factor F was a *square* matrix, i.e., $r = n$. These choices result in an unconstrained nonconvex optimization problem that can be tackled with a first-order optimization algorithm.

Theoretical work of Barvinok [8] and Pataki [54] demonstrates that the primal SDP (P) always admits a solution with rank r, provided that $\binom{r+1}{2} > m$. (Note, however, that the SDP can have solutions with much lower or higher rank.) Interestingly, it is possible to find a low rank approximate solution to an SDP, with high probability, by projecting any solution to the SDP onto a random subspace [7, Proposition 6.1], [60]. However, this approach requires having already computed a (possibly full rank) solution using some other method.

Inspired by the existence of low rank solutions to SDP, Burer & Monteiro [17] proposed to solve the optimization problem (2.1) where the variable $F \in \mathbf{R}^{n\times p}$ is constrained to be a *tall* matrix ($p \ll n$). The number p is called the factorization rank. It is clear that every rank-r solution to the SDP (P) induces a solution to the factorized problem (2.1) when $p \geq r$. Burer & Monteiro applied a limited-memory BFGS algorithm to solve (2.1) in an explicit effort to reduce storage costs.

In subsequent work, Burer & Monteiro [18] proved that, under technical conditions, the local minima of the nonconvex formulation (2.1) are global minima of the SDP (P), provided that the factorization rank p satisfies $\binom{p+1}{2} \geq m+1$. As a consequence, algorithms based on (2.1) often set the factorization rank $p \approx \sqrt{2m}$, so the storage costs are $\Omega(n\sqrt{m})$.

Unfortunately, a recent result of Waldspurger & Walters [67, Thm. 2] demonstrates that the formulation (2.1) cannot lead to storage-optimal algorithms for generic SDPs. In particular, suppose that the feasible set of (P) satisfies a mild technical condition and contains a matrix with rank *one*. Whenever the factorization rank satisfies $\binom{p+1}{2} + p \leq m$, there is a set of cost matrices C with positive Lebesgue measure for which the factorized problem (2.1) has (1) a unique global optimizer with rank one and (2) at least one suboptimal local minimizer, while the original SDP has a unique primal and dual solution that satisfy strict complementarity. In this situation, the variable in the factorized SDP actually requires $\Omega(n\sqrt{m})$ storage, which is not optimal if $m = \omega(1)$. In view of this negative result, we omit a detailed review of the literature on the analysis of factorization methods. See [67] for a full discussion.

2.5. Summary and Contribution. In short, all extant algorithms for solving the SDP (P) either lack the optimal storage guarantee, or they are heuristic, or both. This paper presents a new algorithm that provably solves the SDP (P) with optimal storage under assumptions that are standard in the optimization literature.

3. Basics and Notation.

Here we introduce some additional notation, and metrics for evaluating the quality of a solution and the conditioning of an SDP.

3.1. Notation. We will work with the Frobenius norm $\|\cdot\|_{\mathrm{F}}$, the ℓ_2 operator norm $\|\cdot\|_{\mathrm{op}}$, and its dual, the ℓ_2 nuclear norm $\|\cdot\|_*$. We reserve the symbols $\|\cdot\|$ and $\|\cdot\|_2$ for the norm induced by the canonical inner product of the underlying real vector space.

For a matrix $B \in \mathbf{R}^{d_1\times d_2}$, we arrange its singular values in decreasing order:

$$\sigma_1(B) \geq \cdots \geq \sigma_{\min(d_1,d_2)}(B).$$

Define $\sigma_{\min}(B) = \sigma_{\min(d_1,d_2)}(B)$ and $\sigma_{\max}(B) = \sigma_1(B)$. We also write $\sigma_{\min>0}(B)$ for the smallest *nonzero* singular value of B. For a linear operator $\mathcal{B} : \mathbf{S}^{d_1} \to \mathbf{R}^{d_2}$, we define

$$\sigma_{\min}(\mathcal{B}) = \min_{\|A\|=1} \|\mathcal{B}(A)\| \quad \text{and} \quad \|\mathcal{B}\|_{\mathrm{op}} = \max_{\|A\|=1} \|\mathcal{B}(A)\|.$$

We use analogous notation for the eigenvalues of a symmetric matrix. In particular, the map $\lambda_i(\cdot) : \mathbf{S}^n \to \mathbf{R}$ reports the ith largest eigenvalue of its argument.

Table 2: Quality of a primal matrix $X \in \mathbf{S}_+^n$ and a dual vector $y \in \mathbf{R}^m$

	primal matrix X	dual vector y
suboptimality (ϵ)	$\mathbf{tr}(CX) - p_\star$	$d_\star - b^* y$
infeasibility (δ)	$\\|((\mathcal{A}X - b), (-\lambda_{\min}(X))_+)\\|$	$(-\lambda_{\min}(Z(y)))_+$
distance to solution set (d)	$\\|X - X^\star\\|_{\mathrm{F}}$	$\\|y - y_\star\\|_2$

3.2. Optimal Solutions. Instate the notation and assumptions from Subsection 1.1. Define the slack operator $Z : \mathbf{R}^n \to \mathbf{S}^n$ that maps a putative dual solution $y \in \mathbf{R}^m$ to its associated slack matrix $Z(y) := C - \mathcal{A}^* y$. We omit the dependence on y if it is clear from the context.

Let the rank of primal solution being $r_\star$ and denote the range as $\mathcal{V}_\star$. We also fix an orthonormal matrix $V_\star \in \mathbf{R}^{n \times r_\star}$ whose columns span $\mathcal{V}_\star$. Introduce the subspace $\mathcal{U}_\star = \mathbf{range}(Z(y_\star))$, and let $U_\star \in \mathbf{R}^{n \times (n - r_\star)}$ be a fixed orthonormal basis for $\mathcal{U}_\star$. We have the decomposition $\mathcal{V}_\star + \mathcal{U}_\star = \mathbf{R}^n$.

For a matrix $V \in \mathbf{R}^{n \times r}$, define the compressed cost matrix and constraint map

$$C_V := V^* C V \quad \text{and} \quad \mathcal{A}_V(S) := \mathcal{A}(VSV^*) \quad \text{for } S \in \mathbf{S}^r. \tag{3.1}$$

In particular, $\mathcal{A}_{V_\star}$ is the compression of the constraint map onto the range of $X_\star$.

3.3. Conditioning of the SDP. Our analysis depends on conditioning properties of the pair of primal (P) and dual (D) SDPs.

First, we measure the strength of the complementarity condition (1.2) using the spectral gaps of the primal solution $X_\star$ and dual slack matrix $Z(y_\star)$:

$$\lambda_{\min>0}(X_\star) \quad \text{and} \quad \lambda_{\min>0}(Z(y_\star))$$

These two numbers capture how far we can perturb the solutions before the complementarity condition fails.

Second, we measure the robustness of the primal solution to perturbations of the problem data b using the quantity

$$\kappa := \frac{\sigma_{\max}(\mathcal{A})}{\sigma_{\min}(\mathcal{A}_{V^\star})}. \tag{3.2}$$

This term arises because we have to understand the conditioning of the system $\mathcal{A}_{V_\star}(S) = b$ of linear equations in the variable $S \in \mathbf{S}^{r_\star}$.

3.4. Quality of Solutions. We measure the quality of a primal matrix variable $X \in \mathbf{S}_+^n$ and a dual vector $y \in \mathbf{R}_m$ in terms of their suboptimality, their infeasibility, and their distance to the true solutions. Table 2 gives formulas for these quantities.

We say that a matrix X is an (ϵ, δ)-solution of (P) if its suboptimality $\epsilon_p(X)$ is at most ϵ and its infeasibility $\delta_p(X)$ is at most δ.

The primal suboptimality $\epsilon_p(X)$ and infeasibility $\delta_p(X)$ are both controlled by the distance $d_p(X)$ to the primal solution set:

$$\epsilon_p(X) \le \|C\|_{\mathrm{F}}\, d_p(X) \quad \text{and} \quad \delta_p(X) \le \max\{1, \|\mathcal{A}\|_{\mathrm{op}}\} d_p(X). \tag{3.3}$$

We can also control the distance of a dual vector y and its slack matrix $Z(y)$ from their optima using the following quadratic growth lemma.

LEMMA 3.1 (Quadratic Growth). *Instate the assumptions from Subsection 1.1. For any dual feasible y with dual slack matrix $Z(y) := C - \mathcal{A}^* y$ and dual suboptimality $\epsilon_d(y) = d_\star - b^* y$, we have*

(3.4)
$$\|(Z(y), y) - (Z(y_\star), y_\star)\| \leq \frac{1}{\sigma_{\min}(\mathcal{D})}\left[\frac{\epsilon}{\lambda_{\min>0}(X_\star)} + \sqrt{\frac{2\epsilon}{\lambda_{\min>0}(X_\star)}\|Z(y)\|_{\text{op}}}\right],$$

where the linear operator $\mathcal{D} : \mathbf{S}^n \times \mathbf{R}^m \to \mathbf{S}^n \times \mathbf{S}^n$ is defined by

$$\mathcal{D}(Z, y) := (Z - (U_\star U_\star{}^*)Z(U_\star U_\star{}^*), Z + \mathcal{A}^* y).$$

The orthonormal matrix $U_\star$ is defined in Subsection 3.2.

We defer the proof of Lemma 3.1 to Appendix A. The name *quadratic growth* arises from a limit of inequality 3.4: when ϵ is small, the second term in the bracket dominates the first term, so $\|y - y_\star\|_2^2 = \mathcal{O}(\epsilon)$ [26].

4. Reduced SDPs and Approximate Complementarity. In this section, we describe two reduced SDP formulations, and we explain when their solutions are nearly optimal for the original SDP (P). We can interpret these results as constructive proofs of the approximate complementarity principle.

4.1. Reduced SDPs. Suppose that we have obtained a dual approximate solution y and its associated dual slack matrix $Z(y) := C - \mathcal{A}^* y$. Let r be a rank parameter, which we will discuss later. Construct an orthonormal matrix $V \in \mathbf{R}^{n \times r}$ whose range is an r-dimensional invariant subspace associated with the r smallest eigenvalues of the dual slack matrix $Z(y)$. Our goal is to compute a matrix X with range V that approximately solves the primal SDP (P).

Our first approach minimizes infeasibility over all psd matrices with range V:

$$\text{(MinFeasSDP)} \qquad \begin{array}{ll} \text{minimize} & \frac{1}{2}\|\mathcal{A}_V(S) - b\|^2 \\ \text{subject to} & S \succeq 0, \end{array}$$

with variable $S \in \mathbf{S}^r$. Given a solution $\hat{S}$, we can form an approximate solution $X_{\text{infeas}} = V\hat{S}V^*$ for the primal SDP (P). This is the same method from Subsection 1.4.

Our second approach minimizes the objective value over all psd matrices with range V, subject to a specified limit δ on infeasibility:

$$\text{(MinObjSDP)} \qquad \begin{array}{ll} \text{minimize} & \mathbf{tr}(C_V S) \\ \text{subject to} & \|\mathcal{A}_V(S) - b\| \leq \delta \quad \text{and} \quad S \succeq 0, \end{array}$$

with variable $S \in \mathbf{S}^r$. Given a solution $\tilde{S}$, we can form an approximate solution $X_{\text{obj}} = V\tilde{S}V^*$ for the primal SDP (P).

As we will see, both approaches lead to satisfactory solutions to the original SDP (P) under appropriate assumptions. Theorem 4.1 addresses the performance of (MinFeasSDP), while Theorem 4.5 addresses the performance of (MinObjSDP). Table 3 summarizes the hypotheses we impose to study each of the two problems, as well as the outcomes of the analysis.

The bounds in this section depend on the problem data and rely on assumptions that are not easy to check. Section 5 shows that there are easily verifiable conditions that allow us to calculate bounds on the quality of X_{infeas} and X_{obj}.

4.2. Analysis of (MinFeasSDP). First, we establish a result that connects the solution of (MinFeasSDP) with the solution of the original problem (P).

THEOREM 4.1 (Analysis of (MinFeasSDP)). *Instate the hypotheses of Subsection 1.1. Moreover, assume the solution rank $r_\star$ is known. Set $r = r_\star$. Let $y \in \mathbf{R}^m$ be feasible for the dual SDP (D) with suboptimality $\epsilon = \epsilon_d(y) = d_\star - b^* y < c_1$, where the constant $c_1 > 0$ depends only on $\mathcal{A}, b$ and C. Then the threshold $T := \lambda_{n-r}(Z(y))$ obeys*

$$T := \lambda_{n-r}(Z(y)) \geq \frac{1}{2}\lambda_{n-r}(Z(y_\star)) > 0,$$

and we have the bound

$$\|X_{\text{infeas}} - X_\star\|_F \leq (1+2\kappa)\left(\frac{\epsilon}{T} + \sqrt{2\frac{\epsilon}{T}\|X_\star\|_{\text{op}}}\right). \tag{4.1}$$

This bound shows that $\|X_{\text{infeas}} - X_\star\|_{\text{F}}^2 = \mathcal{O}(\epsilon)$ when the dual vector y is ϵ suboptimal. Notice this result requires knowledge of the solution rank $r_\star$. The proof of Theorem 4.1 occupies the rest of this section.

4.2.1. Primal Optimizers and the Reduced Search Space. The first step in the argument is to prove that $X_\star$ is near the search space $\{VSV^* : S \in \mathbf{S}_+^r\}$ of the reduced problems. We will use this lemma again, so we state it under minimal conditions.

LEMMA 4.2. *Suppose (P) and (D) admit solutions and satisfy strong duality, Equation (1.1). Further suppose $y \in \mathbf{R}^m$ is feasible and ϵ-suboptimal for the dual SDP (D), and construct the orthonormal matrix V as in Subsection 4.1. Assume that the threshold $T := \lambda_{n-r}(C - \mathcal{A}^* y) > 0$. For any solution $X_\star$ of the primal SDP (P),*

$$\|X_\star - VV^* X_\star VV^*\|_F \leq \frac{\epsilon}{T} + \sqrt{2\frac{\epsilon}{T}\|X_\star\|_{\text{op}}},$$

and

$$\|X_\star - VV^* X_\star VV^*\|_* \leq \frac{\epsilon}{T} + 2\sqrt{r\frac{\epsilon}{T}\|X_\star\|_{\text{op}}}.$$

Proof. Complete V to form a basis $W = [U\, V]$ for $\mathbf{R}^n$, where $U = [v_{r+1}, \ldots, v_n] \in \mathbf{R}^{n\times(n-r)}$ and where v_i is the eigenvector of $Z = C - \mathcal{A}^* y$ associated with the i-th smallest eigenvalue. Rotating into this coordinate system, let's compare X and its projection into the space spanned by V,

$$W^* X_\star W = \begin{bmatrix} U^* X_\star U & U^* X_\star V \\ V^* X_\star U & V^* X_\star V \end{bmatrix}, \quad \text{and} \quad W^* VV^* X_\star VV^* W = \begin{bmatrix} 0 & 0 \\ 0 & V^* X_\star V \end{bmatrix}.$$

Let $X_1 = U^* X_\star U$, $B = U^* X_\star V$ and $X_2 = V^* X_\star V$. Using the unitary invariance of $\|\cdot\|_{\text{F}}$, we have

$$\|X_\star - VV^* X_\star VV^*\|_{\text{F}} = \|W^* X W - W^* VV^* X_\star VV^* W\|_{\text{F}} = \left\|\begin{bmatrix} X_1 & B \\ B & 0 \end{bmatrix}\right\|_{\text{F}}. \tag{4.2}$$

A similar equality holds for $\|\cdot\|_*$. Thus we need only bound the terms X_1 and B. Applying Lemma B.1 to $W X_\star W^* = \begin{bmatrix} X_1 & B \\ B^* & X_2 \end{bmatrix}$, we have

$$\|X_2\|_{\text{op}}\, \mathbf{tr}(X_1) \quad \geq \|BB^*\|_*. \tag{4.3}$$

Using strong duality and feasibility of $X_\star$, we rewrite the suboptimality as

$$\epsilon = b^* y_\star - b^* y = \mathbf{tr}(CX_\star) - (\mathcal{A}X_\star)^* y = \mathbf{tr}(ZX_\star). \tag{4.4}$$

Since all the vectors in U have corresponding eigenvalues at least as large as the threshold $T > 0$, and $Z \succeq 0$ as y is feasible, we have

$$\epsilon = \mathbf{tr}(ZX_\star) = \sum_{i=1}^{n} \lambda_{n-i+1}(Z) v_i^* X_\star v_i \geq \lambda_{n-r}(Z) \sum_{i=r+1}^{n} v_i^* X_\star v_i. \tag{4.5}$$

This inequality allows us to bound $\|X_1\|_{\mathrm{F}}$ as

$$\frac{\epsilon}{T} \geq \sum_{i=r+1}^{n} v_i^* X_\star v_i = \mathbf{tr}(UX_\star U^*) = \mathbf{tr}(X_1) = \|X_1\|_* \geq \|X\|_{\mathrm{F}}, \tag{4.6}$$

where we recall $X_1 \succeq 0$ to obtain the second to last equality. Combining (4.3), (4.6), and $\|X_2\|_{\mathrm{op}} \leq \|X_\star\|_{\mathrm{op}}$, we have

$$\|BB^*\|_* \leq \frac{\epsilon}{T} \|X_2\|_{\mathrm{op}} \leq \frac{\epsilon}{T} \|X_\star\|_{\mathrm{op}}. \tag{4.7}$$

Basic linear algebra shows

$$\left\| \begin{bmatrix} 0 & B \\ B^* & 0 \end{bmatrix} \right\|_{\mathrm{F}}^2 = \mathbf{tr}\left(\begin{bmatrix} BB^* & 0 \\ 0 & B^*B \end{bmatrix} \right) \leq 2\,\mathbf{tr}(BB^*) = 2\,\|BB^*\|_*. \tag{4.8}$$

Combining pieces, we bound the error in the Frobenius norm:

$$\begin{aligned} \|X_\star - VV^*X_\star VV^*\|_{\mathrm{F}} &\overset{(a)}{\leq} \|X_1\|_{\mathrm{F}} + \left\| \begin{bmatrix} 0 & B \\ B^* & 0 \end{bmatrix} \right\|_{\mathrm{F}} \\ &\overset{(b)}{\leq} \frac{\epsilon}{T} + \sqrt{2\,\|BB^*\|_*} \\ &\overset{(c)}{\leq} \frac{\epsilon}{T} + \sqrt{\frac{2\epsilon}{T} \|X_\star\|_{\mathrm{op}}}, \end{aligned}$$

where step (a) uses (4.2) and the triangle inequality; step (b) uses (4.6) and (4.8); and step (c) uses (4.7).

Similarly, we may bound the error in the nuclear norm:

$$\begin{aligned} \|X_\star - VV^*X_\star VV^*\|_* &\overset{(a)}{\leq} \|X_1\|_* + \left\| \begin{bmatrix} 0 & B \\ B^* & 0 \end{bmatrix} \right\|_* \\ &\overset{(b)}{\leq} \mathbf{tr}(X_1) + \sqrt{2r} \left\| \begin{bmatrix} 0 & B \\ B^* & 0 \end{bmatrix} \right\|_{\mathrm{F}} \\ &\overset{(c)}{\leq} \frac{\epsilon}{T} + 2\sqrt{\frac{r\epsilon}{T} \|X_\star\|_{\mathrm{op}}} \end{aligned}$$

by the same reasoning. Step (b) uses the fact that $\begin{bmatrix} 0 & B \\ B^* & 0 \end{bmatrix}$ has rank at most $2r$. ∎

4.2.2. Relationship between the Solutions of (MinFeasSDP) and (P). Lemma 4.2 shows that any solution $X_\star$ of (P) is close to its compression $VV^*X_\star VV^*$ onto the range of V. Next, we show that X_{infeas} is also close to $VV^*X_\star VV^*$. We can invoke strong convexity of the objective of (MinFeasSDP) to achieve this goal.

LEMMA 4.3. *Instate the assumptions from Lemma 4.2. Assume $\sigma_{\min}(\mathcal{A}_V) > 0$. and that the threshold $T = \lambda_{n-r}(Z(y)) > 0$. Then*

$$\|X_{\text{infeas}} - X_\star\|_F \le \left(1 + \frac{\sigma_{\max}(\mathcal{A})}{\sigma_{\min}(\mathcal{A}_V)}\right)\left(\frac{\epsilon}{T} + \sqrt{2\frac{\epsilon}{T}\|X_\star\|_{\text{op}}}\right), \tag{4.9}$$

where $X_\star$ is any solution of the primal SDP (P).

Proof. Since we assume that $\sigma_{\min}(\mathcal{A}_V) > 0$, we know the objective of (MinFeasSDP), $f(S) = \frac{1}{2}\|\mathcal{A}_V(S) - b\|_2^2$, is $\sigma^2_{\min}(\mathcal{A}_V)$-strongly convex, and so the solution $S_\star$ is unique. We then have for any $S \in \mathbf{S}^r$

$$\begin{aligned} f(S) - f(S^\star) &\overset{(a)}{\ge} \mathbf{tr}(\nabla f(S^\star)^*(S - S^\star)) + \frac{\sigma_{\min}(\mathcal{A}_V)}{2}\|S - S^\star\|_{\text{F}}^2 \\ &\overset{(b)}{\ge} \frac{\sigma^2_{\min}(\mathcal{A}_V)}{2}\|S - S^\star\|_{\text{F}}^2, \end{aligned} \tag{4.10}$$

where step (a) uses strong convexity and step (b) is due to the optimality of $S_\star$.

Since $\mathcal{A}X_\star = b$, we can bound the objective of (MinFeasSDP) by $\|VV^*XVV^* - X_\star\|_{\text{F}}$:

$$\begin{aligned} \|\mathcal{A}_V(V^*XV) - b\|_2 &= \|\mathcal{A}(VV^*XVV^* - X_\star)\|_2 \\ &\le \sigma_{\max}(\mathcal{A})\|VV^*XVV^* - X_\star\|_{\text{F}}. \end{aligned} \tag{4.11}$$

Combining pieces, we know that $S_\star$ satisfies

$$\begin{aligned} \|S^\star - V^*X_\star V\|_{\text{F}}^2 &\overset{(a)}{\le} \frac{2}{\sigma^2_{\min}(\mathcal{A}_V)}(f(V^*X_\star V) - f(S^\star)) \\ &\overset{(b)}{\le} \frac{\sigma^2_{\max}(\mathcal{A})}{\sigma^2_{\min}(\mathcal{A}_V)}\|X_\star - VV^*X_\star VV^*\|_{\text{F}}^2 \\ &\overset{(c)}{\le} \frac{\sigma^2_{\max}(\mathcal{A})}{\sigma^2_{\min}(\mathcal{A}_V)}\left(\frac{\epsilon}{T} + \sqrt{2\frac{\epsilon}{T}\|X_\star\|_{\text{op}}}\right)^2, \end{aligned}$$

where step (a) uses (4.10), step (b) uses (4.11), and step (c) uses Lemma 4.2. Lifting to the larger space $\mathbf{R}^{n\times n}$, we see

$$\begin{aligned} \|VS^\star V^* - X_\star\|_{\text{F}} &\le \|VS^\star V^* - VV^*X_\star VV^*\|_{\text{F}} + \|X_\star - VV^*X_\star VV^*\|_{\text{F}} \\ &\overset{(a)}{=} \|S^\star - V^*X_\star V\|_{\text{F}} + \|X_\star - VV^*X_\star VV^*\|_{\text{F}} \\ &\overset{(b)}{\le} \left(1 + \frac{\sigma_{\max}(\mathcal{A})}{\sigma_{\min}(\mathcal{A}_V)}\right)\left(\frac{\epsilon}{T} + \sqrt{2\frac{\epsilon}{T}\|X_\star\|_{\text{op}}}\right). \end{aligned}$$

where we use the unitary invariance of $\|\cdot\|_{\text{F}}$ in (a). The inequality (b) uses our bound above for $S_\star$ and Lemma 4.2. ∎

4.2.3. Lower Bounds for the Threshold and Minimum Singular Value. Finally, we must confirm that the extra hypotheses of Lemma 4.3 hold, viz., $T > 0$ and $\sigma_{\min}(\mathcal{A}_V) > 0$.

Here is the intuition. Strict complementarity forces $\lambda_{n-r}(Z(y_\star)) > 0$. If Z is close to $Z(y_\star)$, then we expect that $T > 0$ by continuity. When $X_\star$ is unique, Lemma A.1 implies that $\mathbf{null}(\mathcal{A}_{V_\star}) = \{0\}$. As a consequence, $\sigma_{\min}(\mathcal{A}_{V_\star}) > 0$. If V is close to $V_\star$, then we expect that $\sigma_{\min}(\mathcal{A}_V) > 0$ as well. We have the following rigorous statement.

LEMMA 4.4. *Instate the hypotheses of Theorem 4.1. Then*

$$T = \lambda_{n-r}(Z(y)) \geq \frac{1}{2}\lambda_{n-r}(Z(y_\star));$$
$$\sigma_{\min}(\mathcal{A}_V) \geq \frac{1}{2}\sigma_{\min}(\mathcal{A}_{V^\star}) > 0.$$

Proof. We first prove the lower bound on the threshold T. Using $\|(Z,y) - (Z(y_\star), y_\star)\| \geq \|Z - Z(y_\star)\|_{\text{op}} \geq \|Z\|_{\text{op}} - \|Z(y_\star)\|_{\text{op}}$ and quadratic growth (Lemma 3.1), we have

$$\|Z\|_{\text{op}} - \|Z(y_\star)\|_{\text{op}} \leq \frac{1}{\sigma_{\min}(\mathcal{D})}\left(\frac{\epsilon}{\lambda_{\min>0}(X_\star)} + \sqrt{\frac{2\epsilon}{\lambda_{\min>0}(X_\star)}\|Z\|_{\text{op}}}\right).$$

Thus for sufficiently small ϵ, we have $\|Z\|_{\text{op}} \leq 2\|Z(y_\star)\|_{\text{op}}$. Substituting this bound into previous inequality gives,

(4.12)
$$\|(Z,y) - (Z(y_\star), y_\star)\| \leq \frac{1}{\sigma_{\min}(\mathcal{D})}\left(\frac{\epsilon}{\lambda_{\min>0}(X_\star)} + \sqrt{\frac{4\epsilon}{\lambda_{\min>0}(X_\star)}\|Z(y_\star)\|_{\text{op}}}\right).$$

Weyl's inequality tells us that $\lambda_{n-r}(Z(y_\star)) - T \leq \|Z - Z(y_\star)\|_{\text{op}}$. Using (4.12), we see that for all sufficiently small ϵ, $T := \lambda_{n-r}(C - \mathcal{A}^* y) \geq \frac{1}{2}\lambda_{n-r}(Z(y_\star))$.

Next we prove the lower bound on $\mathcal{A}_V$. We have $\sigma_{\min}(\mathcal{A}_{V^\star}) > 0$ by Lemma A.1. It will be convenient to align the columns of V with those of $V^\star$ for our analysis. Consider the solution O to the orthogonal Procrustes problem $O = \operatorname{argmin}_{OO^*=I, O\in\mathbf{R}^{r\times r}} \|VO - V^\star\|_{\text{F}}$. Since $\sigma_{\min}(\mathcal{A}_V) = \sigma_{\min}(\mathcal{A}_{VO})$ for orthonormal O, without loss of generality, we suppose we have already performed the alginment and $V = VO$ in the following.

Let $S_1 = \operatorname{argmin}_{\|S\|_{\text{F}}=1} \|\mathcal{A}_V(S)\|_2$. Then we have

$$\begin{aligned}\sigma_{\min}(\mathcal{A}_{V_\star}) - \sigma_{\min}(\mathcal{A}_V) &\leq \|\mathcal{A}_{V_\star}(S_1)\|_2 - \|\mathcal{A}_V(S_1)\|_2 \\ &\leq \|\mathcal{A}(V^\star S_1 (V^\star)^*) - \mathcal{A}(V S_1 V^*)\|_2 \\ &\leq \|\mathcal{A}\|_{\text{op}} \|V^\star S_1 (V^\star)^* - (V S_1 V^*)\|_{\text{F}}.\end{aligned} \tag{4.13}$$

Defining $E = V - V^\star$, we bound the term $\|V^\star S_1 (V^\star)^* - (V S_1 V^*)\|_{\text{F}}$ as

$$\begin{aligned}\|V^\star S_1 (V^\star)^* - (V S_1 V^*)\|_{\text{F}} &= \|E S_1 (V_\star)^* + V_\star S_1 E^* + E S_1 E^*\|_{\text{F}} \\ &\overset{(a)}{\leq} 2\|E\|_{\text{F}}\|V_\star S_1\|_{\text{F}} + \|E\|_{\text{F}}^2 \|S_1\|_{\text{F}} \\ &\overset{(b)}{=} 2\|E\|_{\text{F}} + \|E\|_{\text{F}}^2,\end{aligned} \tag{4.14}$$

where (a) uses the triangle inequality and the submultiplicativity of the Frobenius norm. We use the orthogonality of the columns of V and of $V^\star$ and the fact that $\|S_1\|_{\text{F}} = 1$ in step (b).

A variant of the Davis–Kahan inequality [68, Theorem 2] asserts that $\|E\|_{\text{F}} \leq 4\|Z - Z(y_\star)\|_{\text{F}} / \lambda_{\min>0}(Z(y_\star))$. Combining this fact with inequality (4.12), we see $\|E\|_{\text{op}} \to 0$ as $\epsilon \to 0$. Now using (4.14) and (4.13), we see that for all sufficiently small ϵ, $\sigma_{\min}(\mathcal{A}_V) \geq \frac{1}{2}\sigma_{\min}(\mathcal{A}_{V^\star}) > 0$. ∎

4.2.4. Proof of Theorem 4.1. Instate the hypotheses of Theorem 4.1. Now, Lemma 4.4 implies that $\sigma_{\min}(\mathcal{A}_V) > 0$ and that $T > 0$. Therefore, we can invoke Lemma 4.3 to obtain the stated bound on $\|X_{\text{infeas}} - X_\star\|_{\text{F}}$.

4.3. Analysis of (MinObjSDP). Next, we establish a result that connects the solution to (MinObjSDP) with the solution to the original problem (P).

THEOREM 4.5 (Analysis of (MinObjSDP)). *Instate the hypotheses of Subsection 1.1. Moreover, assume $r \geq r_\star$. Let $y \in \mathbf{R}^m$ be feasible for the dual SDP (D) with suboptimality $\epsilon = \epsilon_d(y) = d_\star - b^* y < c_2$, where the constant $c_2 > 0$ depends only on $\mathcal{A}, b$ and C. The threshold $T := \lambda_{n-r}(Z(y))$ obeys*

$$T := \lambda_{n-r}(Z(y)) \geq \frac{1}{2}\lambda_{n-r}(Z(y_\star)) > 0.$$

Introduce the quantities

$$\delta_0 := \sigma_{\max}(\mathcal{A})\left(\frac{\epsilon}{T} + \sqrt{2\frac{\epsilon}{T}\|X_\star\|_{\text{op}}}\right);$$

$$\epsilon_0 := \min\left\{\|C\|_F\left(\frac{\epsilon}{T} + \sqrt{2\frac{\epsilon}{T}\|X_\star\|_{\text{op}}}\right), \|C\|_{\text{op}}\left(\frac{\epsilon}{T} + \sqrt{2\frac{r\epsilon}{T}\|X_\star\|_{\text{op}}}\right)\right\}.$$

If we solve (MinObjSDP) with the infeasibility parameter $\delta = \delta_0$, then the resulting matrix X_{obj} is an (ϵ_0, δ_0) solution to (P).

If in addition $C = I$, then X_{obj} is superoptimal with $0 \geq \epsilon_0 \geq -\frac{\epsilon}{T}$.

The analysis Theorem 4.1 of (MinFeasSDP) requires knowledge of the solution rank $r_\star$, and the bounds depend on the conditioning κ. In contrast, Theorem 4.5 does not require knowledge of $r_\star$, and the bounds do not depend on κ. Table 3 compares our findings for the two optimization problems Theorem 4.1 and Theorem 4.5.

Remark 4.6. The quality of the primal reconstruction depends on the ratio between the threshold T and the suboptimality ϵ. The quality improves as the suboptimality ϵ decreases, so the primal reconstruction approaches optimality as the dual estimate y approaches optimality. The threshold T is increasing in the rank estimate r, and so the primal reconstruction improves as r increases. Since r controls the storage required for the primal reconstruction, we see that the quality of the primal reconstruction improves as our storage budget increases.

Remark 4.7. Using the concluding remarks of [62], the above bound on suboptimality and infeasibility shows that the distance between X_{obj} and $\mathcal{X}^\star$ is at most $\mathcal{O}(\epsilon^{1/4})$. Here, the $\mathcal{O}(\cdot)$ notation omits constants depending on $\mathcal{A}$, b, and C.

The proof of Theorem 4.5 occupies the rest of this subsection.

4.3.1. Bound on the Threshold via Quadratic Growth. We first bound T when the suboptimality of y is bounded. This bound is a simple consequence of quadratic growth (Lemma 3.1).

LEMMA 4.8. *Instate the hypotheses of Theorem 4.5. Then*

$$T := \lambda_{n-r}(Z(y)) \geq \frac{1}{2}\lambda_{n-r}(Z(y_\star)) > 0.$$

Proof. The proof is similar to the proof of Lemma 4.4, by noting $\lambda_{n-r}(Z(y_\star)) > 0$ whenever $r \geq n - \mathbf{rank}(Z(y_\star))$. We omit the details. ∎

Table 3: Comparison of (MinFeasSDP) and (MinObjSDP) given a feasible ϵ-suboptimal dual vector y.

Assumption and Quality	(MinFeasSDP)	(MinObjSDP)
Require assumptions in Subsection 1.1 ?	Yes	Yes
Require $r = r_\star$?	Yes	No
Suboptimality	$\mathcal{O}(\kappa\sqrt{\epsilon})$	$\mathcal{O}(\sqrt{\epsilon})$
Infeasibility	$\mathcal{O}(\kappa\sqrt{\epsilon})$	$\mathcal{O}(\sqrt{\epsilon})$
Distance to the solution	$\mathcal{O}(\kappa\sqrt{\epsilon})$	Remark 4.7

4.3.2. Proof of Theorem 4.5. Lemma 4.2 shows that any primal solution $X_\star$, is close to $VV^*X_\star VV^*$. We must ensure that $VV^*X_\star VV^*$ is feasible for (MinObjSDP). This is achieved by setting the infeasibility parameter in (MinObjSDP) as

$$\delta := \sigma_{\max}(\mathcal{A})\left(\frac{\epsilon}{T} + \sqrt{2\frac{\epsilon\,\|X_\star\|_{\mathrm{op}}}{T}}\right)$$

This choice also guarantees all solutions to (MinObjSDP) are δ-feasible.

The solution to (MinObjSDP) is δ_0-feasible by construction. It remains to show the solution is ϵ_0-suboptimal. We can bound the suboptimality of the feasible point $VV^*X_\star VV^*$ to produce a bound on the suboptimality of the solution to (MinObjSDP). We use Hölder's inequality to translate the bound on the distance between $VV^*X_\star VV^*$ and $X_\star$, from Lemma 4.2, into a bound on the suboptimality:

$$\begin{aligned}&\mathbf{tr}(C(VV^*X_\star VV^* - X_\star))\\ &\qquad\le \epsilon_0 := \min\left\{\|C\|_{\mathrm{F}}\left(\frac{\epsilon}{T} + \sqrt{2\frac{\epsilon}{T}\|X_\star\|_{\mathrm{op}}}\right), \|C\|_{\mathrm{op}}\left(\frac{\epsilon}{T} + \sqrt{2\frac{r\epsilon}{T}\|X_\star\|_{\mathrm{op}}}\right)\right\}.\end{aligned}$$

This argument shows that $VV^*X_\star VV^*$, and hence the solution to (MinObjSDP), is at most ϵ_0 suboptimal.

To establish the improved bound when $C = I$, using the notation in the proof of Lemma 4.2, we have

$$\mathbf{tr}(X_\star) = \mathbf{tr}(X_1) + \mathbf{tr}(X_2), \quad \text{and} \quad \mathbf{tr}(X_2) = \mathbf{tr}(VV^*X_\star VV^*). \tag{4.15}$$

Now using the inequality (4.6), $\mathbf{tr}(X_1) \le \frac{\epsilon}{T}$, and the fact $X_1 \succeq 0$, we see

$$\mathbf{tr}(X_\star) \ge \mathbf{tr}(VV^*X_\star VV^*) \ge \mathbf{tr}(X_\star) - \frac{\epsilon}{T}.$$

This completes the argument.

5. Computational Aspects of Primal Recovery. The previous section introduced two methods, (MinFeasSDP) and (MinObjSDP), to recover an approximate primal from an approximate dual solution y. It contains theoretical bounds on suboptimality, infeasibility, and distance to the solution set of the primal SDP (P). We summarize this approach as Algorithm 5.1.

In this section, we turn this approach into a practical optimal storage algorithm, by answering the following questions:

Algorithm 5.1 Primal recovery via (MinFeasSDP) or (MinObjSDP)

Require: Problem data $\mathcal{A}$, C and b; dual vector y and positive integer r
1: Compute an orthonormal matrix $V \in \mathbf{R}^{n \times r}$ whose range is an invariant subspace of $C - \mathcal{A}^* y$ associated with the r smallest eigenvalues.
2: Solve (MinFeasSDP) or (MinObjSDP) to obtain a matrix $\hat{S} \in \mathbf{S}^r_+$.
3: **return** $\hat{S}$ and V

1. How should we solve (MinFeasSDP) and (MinObjSDP)?
2. How should we choose δ in (MinObjSDP)?
3. How should we choose the rank parameter r?
4. How can we estimate the suboptimality, infeasibility, and (possibly) the distance to the solution to use as stopping conditions?

In particular, our choices for algorithmic parameters should not depend on any quantities that are unknown or difficult to compute. We address each question in turn.

For this discussion, let us quantify the cost of the three data access oracles (1.4). We use the mnemonic notation L_C, $L_{\mathcal{A}}$, and $L_{\mathcal{A}^*}$ for the respective running time (denominated in flops) of the three operations.

5.1. Solving MinFeasSDP and MinObjSDP. Suppose that we have a dual estimate $y \in \mathbf{R}^m$, and that we have chosen $r = \mathcal{O}(r_\star)$ and δ. The recovery problems (MinFeasSDP) and (MinObjSDP) are SDPs with an $r \times r$ decision variable and m linear constraints, and so can be solved in several ways that all guarantee optimal storage $\mathcal{O}(m + nr)$:

- We can apply matrix-free convex programming solvers, as described in [25, 52]. These methods require access to the operators

$$S \mapsto \mathcal{A}(VSV^*), \quad S \mapsto \mathbf{tr}(C_V S), \quad \text{and} \quad y \mapsto V^*(\mathcal{A}^*(y))V.$$

  These operators can be evaluated by repeated calls to the data access oracles, and their outputs have size m, 1, and r^2 respectively, so the resulting algorithm uses optimal storage. Evaluating these operators requires $r^2 L_{\mathcal{A}}$, $r^2 L_C$ and $r L_{\mathcal{A}^*} + nr^2$ flops, respectively.
- We can solve (MinFeasSDP) directly using the projected gradient method. The gradient can be computed efficiently if the quantities $L_{\mathcal{A}}$ and $L_{\mathcal{A}^*}$ are modest, and the projection step is not too difficult because the feasible region is the (low dimensional) PSD cone $\mathbf{S}^r_+$. More precisely, the gradient of the objective $f(S) := \frac{1}{2}\|\mathcal{A}(VSV^*) - b\|_2^2$ is

$$\nabla f(S) = V^* \left[\mathcal{A}^*(\mathcal{A}(VSV^*) - b)\right] V.$$

  Evaluating the gradient requires $r^2 L_{\mathcal{A}} + m + r L_{\mathcal{A}^*} + r^2 n$ flops: $r^2 L_{\mathcal{A}} + m$ flops to evaluate $\mathcal{A}(VSV^\top) - b$, $r L_{\mathcal{A}^*}$ flops to compute $\left[\mathcal{A}^*(\mathcal{A}(VSV^\top) - b)\right] V$ once $\mathcal{A}(VSV^\top) - b$ is available, and $r^2 n$ more to form $V^\top \left[\mathcal{A}^*(\mathcal{A}(VSV^\top) - b)\right] V$. The projection step requires $\mathcal{O}(r^3)$ flops for the eigenvalue decomposition [64]. Thus, the flop count for one step of the projected gradient method is

$$\underbrace{r^2 L_{\mathcal{A}} + m + r L_{\mathcal{A}^*} + r^2 n}_{\text{form the gradient}} + \underbrace{r^2}_{\text{take step}} + \underbrace{\mathcal{O}(r^3)}_{\text{project}} = \mathcal{O}(r^2 L_{\mathcal{A}} + r L_{\mathcal{A}^*} + m + r^2 n).$$

  We must store the primal iterate S, the infeasibility $\mathcal{A}(VSV^\top) - b$, and the gradient $\nabla f(S)$ ($r^2 + m + r^2$ storage). Hence the projected gradient method indeed uses optimal storage.

- We can solve (MinObjSDP) directly by materializing the operator $\mathcal{A}_V$ and the projection of the matrix C onto the subspace V. This procedure is easiest to implement, but uses slightly higher storage than is optimal. Form the matrices $M \in \mathbf{R}^{m \times r^2}$ and $C_V \in \mathbf{R}^{r \times r}$ so that

$$M\,\mathbf{vec}(S) = \mathcal{A}_V(S) \qquad \text{and} \qquad \mathbf{tr}(C_V S) = \mathbf{tr}(CVSV^*),$$

for any $S \in \mathbf{R}^{r \times r}$. Then solve the following small SDP using any SDP solver:

$$\begin{array}{ll} \text{minimize} & \mathbf{tr}(\tilde{C}S) \\ \text{subject to} & \|M\,\mathbf{vec}(S) - b\| \le \delta, \\ & S \succeq 0. \end{array} \tag{5.1}$$

To materialize $\mathcal{A}_V$, we can form each of the r^2 standard basis elements in $\mathbf{R}^{r^2}$ and apply the operator $\mathcal{A}_V$ to each to compute each element of M. The procedure requires r^2 calls of the operator $\mathcal{A}$ and hence $r^2 L_{\mathcal{A}}$ flops. The storage required for this procedure is mr^2. A similar procedure can be used to form the matrix $\tilde{C} \in \mathbf{R}^{r^2}$, which requires $r^2 L_C$ flops. The total storage required (in addition to the data access oracles) to form M and $\tilde{C}$ is $mr^2 + r^2$: about a factor of r^2 larger than the optimal storage ($\mathcal{O}(m + nr)$). The computation requires $r^2 L_{\mathcal{A}} + r^2 L_C$ flops. After forming M and $\tilde{C}$, we may use a first order solver, e.g., on based on ADMM [53], to achieve a total storage requirement of $\mathcal{O}(mr^2 + nr)$. We can also directly employ the matrix free method described earlier to achieve optimal storage with a somewhat more complex implementation. Alternatively, we may use an interior point solver, which requires at least m^2 storage (to form the Hessian matrix for the dual variable corresponding to $M\,\mathbf{vec}(S) - b$): not storage optimal, but fast.

5.2. Choosing the Infeasibility Parameter δ. One safe way to choose δ is to solve (MinFeasSDP) to obtain a solution X_{infeas}. Then set $\delta = \gamma \left\|\mathcal{A}_V(X_{\text{infeas}}) - b\right\|_2$ for some $\gamma \ge 1$, which guarantees that (MinObjSDP) is feasible. Using $\gamma > 1$ increases the feasible region of (MinObjSDP). When a bound on $\left\|X_\star\right\|_{\text{op}}$ is available (which is often true; see Subsection 6.1 and Appendix D), we can use the explicit formula for δ from Theorem 5.1. Using this value provides theoretical guarantees on the suboptimality and infeasibility of the solution X_{obj}.

5.3. Choosing the Rank Parameter r. Theorem 4.5 shows that (MinObjSDP) provides useful results so long as the rank estimate r exceeds the true rank $r_\star$, and the quality of the solution improves as r increases. A user seeking the best possible solution to (MinObjSDP) should choose the largest rank estimate r for which the SDP (MinObjSDP) can still be solved, given computational and storage limitations. For (MinFeasSDP), even though our Theorem 4.1 requires the rank estimate r to match $r_\star$, we found in our numerics Section 7 that in fact (MinFeasSDP) also finds a good solution X_{infeas} when the rank estimate is larger than $r_\star$. In Theorem 5.1, we bound the distance to the solution even when $r \ne r_\star$, as long as an easily checkable condition holds. In practice, it is usually not too hard to choose a rank estimate r by considering the decay of the spectrum (first few smallest eigenvalues) of the dual slack matrix $C - \mathcal{A}^* y$ for an approximate dual solution y.

5.4. Bounds on Suboptimality, Infeasibility, and Distance to the Solution Set. Suppose we solve either (MinObjSDP) or (MinFeasSDP) to obtain a primal estimate $X = X_{\text{obj}}$ or $X = X_{\text{infeas}}$. How can we estimate its suboptimality, infeasibility, and distance to the solution set of (P)?

The first two are easy: we can directly compute the suboptimality $\epsilon_p = \mathbf{tr}(CX) - b^* y$ and infeasibility $\delta_p = \|\mathcal{A}X - b\|_2$.

It is harder to bound the distance to the solution. Fortunately, when a bound on $\|X_\star\|_{\mathrm{op}}$ is available (often true in applications; see Subsection 6.1 and Appendix D), we can use Lemma 4.3 to bound the distance to the solution for (MinFeasSDP). Moreover, based on this bound, we can also estimate ϵ_p, δ_p for the solution X_{obj} of (MinObjSDP) before solving it. We state the bounds on these three quantities more precisely in Theorem 5.1. Here we use weaker assumptions than in subsection 1.1.

Theorem 5.1 (Computable Bounds). *Suppose* (P) *and* (D) *admit solutions and satisfy strong duality, Equation* (1.1). *Let $y \in \mathbf{R}^m$ be a dual feasible point with suboptimality $\epsilon = \epsilon_d(y) = b^* y_\star - b^* y$. For a positive integer r, form the orthonormal matrix $V \in \mathbf{R}^{n\times r}$, as in Algorithm 5.1, and compute the threshold $T = \lambda_{n-r}(Z(y))$.*

If $\sigma_{\min}(\mathcal{A}_V) > 0$ and $T > 0$ and $\|X_\star\|_{\mathrm{op}} \le B$ for some solution $X_\star$ to (P), *then*

$$\|X_{\mathrm{infeas}} - X_\star\|_F \le \left(1 + \frac{\sigma_{\max}(\mathcal{A})}{\sigma_{\min}(\mathcal{A}_V)}\right)\left(\frac{\epsilon}{T} + \sqrt{2\frac{\epsilon}{T}B}\right). \tag{5.2}$$

Moreover, any solution $\tilde{S}$ of (MinObjSDP) *with infeasibility parameter*

$$\delta \ge \delta_0 := \sigma_{\max}(\mathcal{A})\left(\frac{\epsilon}{T} + 2\sqrt{\frac{2\epsilon B}{T}}\right) \tag{5.3}$$

leads to an (ϵ_0, δ)-solution X_{obj} for the primal SDP (P) *with*

$$\epsilon_0 = \min\left\{ \|C\|_F\left(\frac{\epsilon}{T} + \sqrt{2\frac{\epsilon}{T}B}\right), \|C\|_{\mathrm{op}}\left(\frac{\epsilon}{T} + \sqrt{2\frac{r\epsilon}{T}B}\right)\right\}. \tag{5.4}$$

Proof. The inequality (5.2) is a direct application of Lemma 4.2 and Lemma 4.3. The bound on δ_0 and ϵ_0 follows the same proof as in Theorem 4.5. ∎

When no prior bound on $\|X_\star\|_{\mathrm{op}}$ is available, we can invoke Lemma C.1 to estimate $\|X_\star\|_{\mathrm{op}}$ using any feasible point of (MinFeasSDP). The quantities T, V, κ appearing in Theorem 5.1 can all be computed efficiently from available information. For example, we can apply a Krylov subspace method [64], since the matrix or operator associated with each can be applied efficiently to vectors or low-rank matrices.

5.5. Well-posedness. Theorem 5.1 makes no guarantee on the quality of the reconstructed primal when $\min\{\sigma(\mathcal{A}_V), T\} = 0$. In fact, this failure signals either that y is far from optimality, or that the primal (P) or dual (D) is degenerate (violating the assumptions from Subsection 1.1).

To see this, suppose for simplicity, we know the rank $r = r_\star$ of a solution to (P). If y is close to $y_\star$ and the primal (P) and dual (D) are not degenerate, then Lemma 4.4 shows that the quantities $\min\{\sigma(\mathcal{A}_V), T\}$ are close to $\min\{\sigma(\mathcal{A}_{V_\star}), \lambda_{n-r_\star}(Z(y_\star))\}$. Furthermore, Lemma 4.4 shows that our assumptions (from Subsection 1.1) guarantee $\min\{\sigma(\mathcal{A}_{V_\star}), \lambda_{n-r_\star}(Z(y_\star))\} > 0$. Thus if $\min\{\sigma(\mathcal{A}_V), T\} = 0$, then either we need a more accurate solution to the dual problem to recover the primal, or the problem is degenerate and our assumptions fail to hold.

6. Computational Aspects of the Dual SDP (D). The previous two subsections showed how to efficiently and robustly recover an approximate primal solution from an approximate dual solution. We now discuss how to (approximately) solve the dual SDP (D) with optimal storage and with a low per-iteration computational cost. Together, the (storage-optimal) dual solver and (storage-optimal) primal recovery compose a new algorithm for solving generic SDPs with optimal storage.

Algorithm 6.1 Dual Algorithm + Primal Recovery

Require: Problem data $\mathcal{A}$, C and b
Require: Positive integer r and an iterative algorithm $\mathcal{G}$ for solving the dual SDP
1: **for** $k = 1, 2, \dots$ **do**
2: Compute the k-th dual y_k via k-th iteration of $\mathcal{G}$
3: Compute a recovered primal $\hat{X}_k = V\hat{S}V^*$ using Primal Recovery, Algorithm 5.1.
4: **end for**

6.1. Exact Penalty Formulation. It will be useful to introduce an unconstrained version of the dual SDP (D), parametrized by a real positive number α, which we call the penalized dual SDP:

$$\text{maximize} \quad b^*y + \alpha \min\{\lambda_{\min}(C - \mathcal{A}^*y), 0\}. \tag{6.1}$$

That is, we penalize vectors y that violate the dual constraint $C - \mathcal{A}^*y \succeq 0$.

Problem (6.1) is an exact penalty formulation for the dual SDP (D). Indeed, the following lemma shows that the solution of Problem (6.1) and the solution set of the dual SDP (D) are the same when α is large enough. The proof is in the same spirit as [59, Theorem 7.21] and is deferred to Appendix D.

LEMMA 6.1. *Instate the assumptions in Subsection 1.1. If $b \neq 0$ and $\alpha > \|X_\star\|_*$, then any solution $y_\star$ to the penalized dual SDP (6.1) solves the dual SDP (D) and vice versa.*

Thus, as long as we know an upper bound on the nuclear norm of the primal solution, then we can solve Problem (6.1) to find the dual optimal solution $y_\star$. It is easy to obtain a bound on $\|X^\star\|_*$ in applications when (1) the objective of (P) is $\mathbf{tr}(X)$, or (2) the constraints of (P) force $\mathbf{tr}(X)$ to be bounded. See Appendix D for more specific examples.

When no such bound is available, we may search over α numerically. For example, solve Problem (6.1) for $\alpha = 2, 4, 8, \dots, 2^d$ for some integer d (perhaps, in parallel, simultaneously). Since any feasible y for the dual SDP (D) may be used to recover the primal, using (MinFeasSDP) and (MinFeasSDP), we can use any approximate solution of the penalized dual SDP, Problem (6.1), for any α, as long as it is feasible for the dual SDP.

6.2. Computational Cost and Convergence Rate for Primal Approximation. Suppose we have an iterative algorithm $\mathcal{G}$ to solve the dual problem. Denote by y_k the kth iterate of $\mathcal{G}$. Each dual iterate y_k generates a corresponding primal iterate using either (MinFeasSDP) or (MinFeasSDP). We summarize this approach to solving the primal SDP in Algorithm 6.1.

The primal iterates X_k generated by Algorithm 6.1 converge to a solution of the primal SDP (P) by our theory.[1] However, it would be computational folly to recover the primal at every iteration: the primal recovery problem is much more computationally challenging than a single iteration of most methods for solving the dual. Hence, to determine when (or how often) to recover the primal iterate from the

[1] Iterative algorithms for solving the dual SDP (D) may not give a feasible point y. If a strictly feasible point is available, we can use the method of Lemma D.1 or Lemma D.2 to obtain a sequence of feasible points from a sequence of (possibly infeasible) iterates without affecting the convergence rate. Alternatively, our theory can be extended to handle the infeasible case; we omit this analysis for simplicity.

dual, we would like to understand how quickly the recovered primal iterates converge to the solution of the primal problem.

To simplify exposition as we discuss algorithms for solving the dual, we reformulate the penalized dual SDP as a convex minimization problem,

$$\text{minimize} \quad g_\alpha(y) := -b^* y + \alpha \max\{\lambda_{\max}(-C + \mathcal{A}^* y), 0\}, \tag{6.2}$$

which has the same solution set as the penalized dual SDP (6.1).

We focus on the convergence of suboptimality and infeasibility, as these two quantities are easier to measure than distance to the solution set. Recall from Table 3 that

$$\epsilon\text{-optimal dual feasible } y \xrightarrow[\text{or MinFeasSDP}]{\text{MinObjSDP}} (\mathcal{O}(\sqrt{\epsilon}), \mathcal{O}(\sqrt{\epsilon}))\text{-primal solution } X \tag{6.3}$$

if $\kappa = \mathcal{O}(1)$. Thus the convergence rate of the primal sequence depends strongly on the convergence rate of the algorithm we use to solve the penalized dual SDP.

6.2.1. Subgradient Methods, Storage Cost, and Per-Iteration Time Cost. We focus on subgradient-type methods for solving the penalized dual SDP (6.1), because the objective g_α is nonsmooth but has an efficiently computable subgradient. Any subgradient method follows a recurrence of the form

$$y_0 \in \mathbf{R}^m \quad \text{and} \quad y_{k+1} = y_k - \eta_k g_k, \tag{6.4}$$

where g_k is a subgradient of g_α at y_k and $\eta_k \geq 0$ is the step size. Subgradient methods differ in the methods for choosing the step size η_k and in their use of parallelism. However, they are all easy to run for our problem because it is easy to compute a subgradient of the dual objective with penalty g_α:

LEMMA 6.2. *Let $Z = C - \mathcal{A}^* y$. The subdifferential of the function g_α is*

$$\partial g_\alpha(y) = \begin{cases} -b + \mathbf{conv}\{\alpha \mathcal{A}(vv^*) \mid Zv = \lambda_{\min}(Z) v\}, & \lambda_{\min}(Z) < 0 \\ -b, & \lambda_{\min}(Z) > 0 \\ -b + \beta\, \mathbf{conv}\{\alpha \mathcal{A}(vv^*) \mid Zv = \lambda_{\min}(Z) v, \beta \in [0,1]\}, & \lambda_{\min}(Z) = 0 \end{cases}.$$

This result follows directly via standard subdifferential calculus from the subdifferential of the maximum eigenvalue $\lambda_{\max}(\cdot)$. Thus our storage cost is simply $\mathcal{O}(m+n)$ where m is due to storing the decision variable y and the gradient g_k, and n is due to the intermediate eigenvector $v \in \mathbf{R}^n$. The main computational cost in computing a subgradient of the objective in (6.2) is computing the smallest eigenvalue $\lambda_{\min}(C - \mathcal{A}^* y)$ and the corresponding eigenvector v of the matrix $C - \mathcal{A}^* y$. Since $C - \mathcal{A}^* y$ can be efficiently applied to vectors (using the data access oracles (1.4)), we can compute this eigenpair efficiently using the randomized Lanczos method [41].

6.2.2. Convergence Rate of the Dual and Primal. The best available subgradient method [38] has convergence rate $\mathcal{O}(1/\epsilon)$ when the quadratic growth condition is satisfied. (This result does not seem to appear in the literature for SDP; however, it is a simple consequence of [38, Table 1] together with the quadratic growth condition proved in Lemma 3.1.) Thus, our primal recovery algorithm has convergence rate $\mathcal{O}(1/\sqrt{\epsilon})$, using the relation between dual convergence and primal convergence in (6.3). Unfortunately, the algorithm in [38] involves many unknown constants. In practice, we recommend using dual solvers that require less tuning. For example, in our numerical experiments, we use AdaGrad [27], AdaNGD [44], and AcceleGrad [45].

7. Numerical Experiments. In this section, we give a numerical demonstration of our approach to solving (P) via approximate complementarity. We first show that Algorithm 5.1 (Primal Recovery) recovers an approximate primal given an approximate dual solution. Next, we show that Algorithm 6.1 (Dual Algorithm + Primal Recovery) effectively solves the primal SDP (P) with a variety of dual solvers.

We test our methods on the Max-Cut and Matrix Completion SDPs, defined in Table 4. For Max-Cut, L is the Laplacian of a given graph. For Matrix Completion,

Table 4: Problems for numerics

Max-Cut	**Matrix Completion**
$\begin{array}{ll} \text{minimize} & \mathbf{tr}(-LX) \\ \text{subject to} & \mathbf{diag}(X) = \mathbf{1} \\ & X \succeq 0 \end{array}$	$\begin{array}{ll} \text{minimize} & \mathbf{tr}(W_1) + \mathbf{tr}(W_2) \\ \text{subject to} & X_{ij} = \bar{X}_{ij},\ (i,j) \in \Omega \\ & \begin{bmatrix} W_1 & X \\ X^* & W_2 \end{bmatrix} \succeq 0 \end{array}$

Ω is the set of indices of the observed entries of the underlying matrix $\bar{X} \in \mathbf{R}^{n_1 \times n_2}$. We show results for Max-Cut using the G1 dataset from [1], which has $L \in \mathbf{R}^{800 \times 800}$. (Many other datasets gave similar results.) To evaluate our method, we compare the recovered primal with the solution $X_\star$, $y_\star$ and $r_\star = 13$ obtained via the MOSEK interior point solver [48]. For matrix completion, we generate a random rank 5 matrix $\bar{X} \in \mathbf{R}^{1000 \times 1500}$. We generate the set Ω by observing each entry of $\bar{X}$ with probability 0.025 independently. To evaluate our method, we compare the recovered primal with $\bar{X}$, which (with high probability) solves the Matrix Completion problem [21].

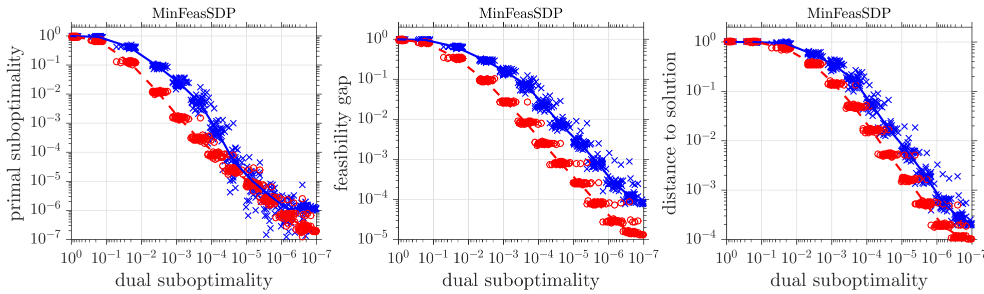
7.1. Primal Recovery. Our first experiment confirms numerically that Algorithm 5.1 (Primal Recovery) recovers an approximate primal from an approximate dual solution, validating our theoretical results. We test recovery using both (MinObjSDP) and (MinFeasSDP) with two different ranks: $r = r_\star$ and $r = 3r_\star$. To obtain approximate dual solutions, we perturb the true dual solution $y_\star$ to generate

$$y = y_\star + \varepsilon s \left\| y_\star \right\|_2 ,$$

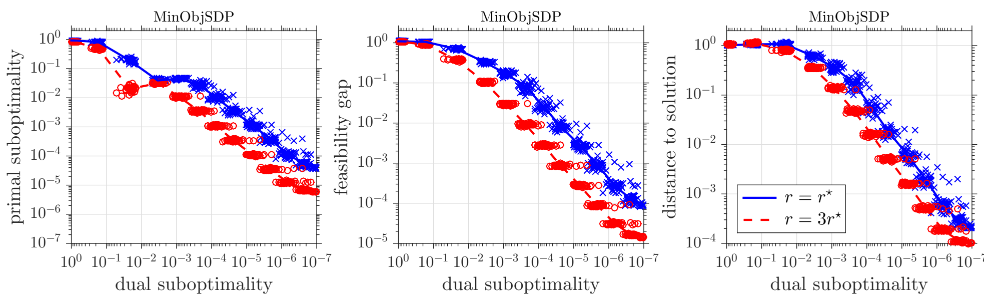
where ε is the noise level, which we vary from 1 to 10^{-5}, and s is a uniformly random vector on the unit sphere in $\mathbf{R}^m$. For each y, we first solve (MinFeasSDP) to obtain a solution X_{infeas}, and then solve (MinObjSDP) with $\delta = 1.1 \left\| \mathcal{A} X_{\text{infeas}} - b \right\|_2$. Figure 1 shows distance of the recovered primal to the solution (as measured by suboptimality, infeasibility, and distance to the solution set) as a function of the dual suboptimality of y: $p^\star + g_\alpha(y)$. (We measure suboptimality with g_α, defined in equation (6.2), since y may not be feasible for the dual SDP (D). In our experiments we set $\alpha = 1.1\,\mathbf{tr}(X_\star)$.) The blue dots represent the primal recovered using $r = r_\star$, while the red dots represent the primal recovered using $r = 3r_\star$. The blue and red curves are fit to the dots of the same color to provide a visual guide. The red line ($r = 3r_\star$) generally lies below the blue line ($r = r_\star$), which confirms that using a higher rank (even higher than the true solution) produces a more accurate primal reconstruction.

These plots show that the recovered primal approaches the true primal solution as the dual suboptimality approaches zero, as expected from our theory.[2] From Table 3,

[2]To be precise, the theory we present above in Theorem 4.1 and Theorem 4.5 requires the approximate dual solution to be feasible, while y may be infeasible in our experiments. A straightforward extension of our results can show similar bounds when y is infeasible but $g_\alpha(y)$ is close to $-d_\star$.



(a) (MinFeasSDP)



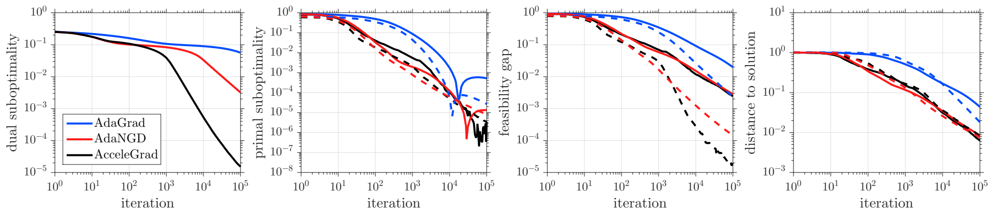
(b) (MinObjSDP)

Fig. 1: The plots shows the primal recovery performance of (MinFeasSDP) (upper) and (MinObjSDP) (lower) in terms of primal suboptimality, infeasibility and the distance to solution. The horizontal axis is the dual suboptimality. The blue dots corresponds to the choice $r = r_\star$ and the red dots corresponds to the choice $r = 3r_\star$ in Algorithm 6.1.

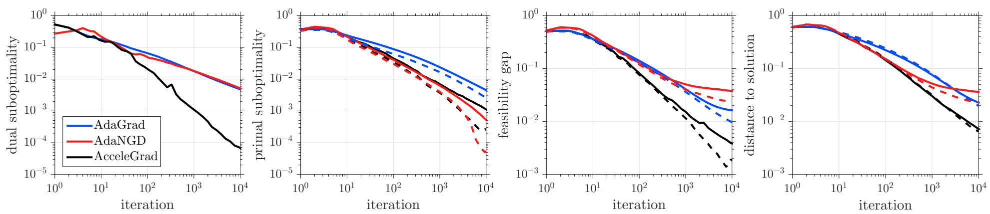
recall that the primal solution recovered from an ϵ suboptimal dual solution should be expected to have a suboptimality, infeasibility, and distance to the true primal solution that all scale as $\mathcal{O}(\sqrt{\epsilon})$. The plots confirm this scaling for distance to solution and infeasibility, while suboptimality decays even faster than predicted by our theory. By construction, the primal suboptimality of (MinObjSDP) is smaller than that of (MinFeasSDP); however, in plots we measure primal suboptimality by its absolute value $|\,\mathbf{tr}(CX) - \mathbf{tr}(CX_\star)|$. Since (MinObjSDP) actually recovers a *superoptimal* X_{obj} ($\mathbf{tr}(CX_{\text{obj}}) - \mathbf{tr}(CX_\star) < 0$) its "suboptimality" appears larger in the figure.

7.2. Solving the primal SDP. Our next experiment shows that Algorithm 6.1 (Dual Algorithm + Primal Recovery) solves the primal SDP, using the dual solvers AdaGrad [27], AdaNGD [44], and AcceleGrad [45]. Here we use (MinFeasSDP) to recover the primal. The numerical results are shown in Figure 2. We plot the dual suboptimality, primal suboptimality, infeasibility and distance to solution for each iteration of the dual method. The solid lines show recovery with $r = r_\star$ while the dotted lines use the higher rank $r = 3r_\star$.

We observe convergence in each of these metrics, as expected from theory. Primal and dual suboptimality converge faster than the other two quantities, as in Figure 1. Interestingly, while AccelGrad converges much faster than the other algorithms on



(a) **Max-Cut**



(b) **Matrix Completion**

Fig. 2: Plots from left to right columns show convergence of penalized dual objective g_α, primal suboptimality, infeasibility, and distance to solution. The solid lines show recovery with $r = r_\star$ while the dotted lines use the higher rank $r = 3r_\star$.

the dual side, its advantage on the primal side is more modest. We again see that the primal recovered using the larger rank $r = 3r_\star$ converges more quickly, though interestingly using the higher rank confers less of an advantage in reducing distance to the solution than in reducing primal suboptimality and infeasibility.

8. Conclusions. This paper presents a new theoretically justified method to recover an approximate solution to a primal SDP from an approximate solution to a dual SDP, using complementarity between the primal and dual optimal solutions. We present two concrete algorithms for primal recovery, which offer guarantees on the suboptimality, infeasibility, and distance to the solution set of the recovered primal under the generic conditions on the SDP, and we demonstrate that this primal recovery method works well in practice.

We use this primal recovery method to develop the first storage-optimal algorithm to solve generic SDP: use any first-order algorithm to solve a penalized version of the dual problem, and recover a primal solution from the dual. This method requires $O(m + nr)$ storage: the storage is linear in the number of constraints m and in the side length n of the SDP variable, when the target rank r of the solution is fixed. These storage requirements improve on the storage requirements that guarantee convergence for nonconvex factored (Burer-Monteiro) methods to solve the SDP, which scale as $O(\sqrt{m}n)$. Furthermore, we show that no method can use less storage without a more restrictive data access model or a more restrictive representation of the solution. We demonstrate numerically that our algorithm is able to solve SDP of practical interest including Max-Cut and Matrix Completion.

The ideas illustrated in this paper can be extended to solve problems with inequality constraints. Further, some of the analytical assumptions can be weakened. We leave these extensions for future work.

Appendix A. Lemmas for Section 1. To establish Lemma 3.1, we prove a lemma concerning the operator $\mathcal{A}_{V_\star}$.

LEMMA A.1. *Instate the hypothesis of Theorem 4.1. Then* $\mathbf{null}(\mathcal{A}_{V^\star}) = \{0\}$.

Proof of Lemma A.1. Suppose by way of contradiction that $\ker(\mathcal{A}_{V_\star}) \neq \{0\}$. Let $S \in \ker(\mathcal{A}_{V_\star})$, so $\mathcal{A}_{V_\star}(S) = 0$. Recall $X_\star = V_\star S_\star (V_\star)^*$ for some unique $S_\star \succ 0$. Hence for some $\alpha_0 > 0$, $S_\star + \alpha S \succeq 0$ for all $|\alpha| \leq \alpha_0$. Now pick any α with $|\alpha| \leq \alpha_0$ to see

$$\mathcal{A}(X_\alpha) = \mathcal{A}_{V_\star}(S_\star + \alpha S) = \mathcal{A}_{V_\star}(S_\star) + 0 = b.$$

This shows X_α is feasible for all $|\alpha| \leq \alpha_0$. But we can always find some $|\alpha| \leq \alpha_0$, $\alpha \neq 0$, so that $\mathbf{tr}(CX_\alpha) = p_\star + \alpha\, \mathbf{tr}(CV_\star(SV_\star)^\top) \leq p_\star$. This contradicts the assumption that $X_\star$ is unique. Hence we must have $\mathbf{null}(\mathcal{A}_{V_\star}) = \{0\}$. ∎

Proof of Lemma 3.1. We write Z to mean $Z(y)$ for arbitrary $y \neq y_\star$ to save some notation. Consider the linear operator $\mathcal{D}$ defined in Lemma 3.1. An argument similar to the proof of Lemma A.1 shows $\ker(\mathcal{D}) = \{0\}$. Hence

$$\|(Z,y) - (Z(y_\star), y_\star)\| \leq \frac{1}{\sigma_{\min}(\mathcal{D})} \|\mathcal{D}(Z - Z(y_\star), y - y_\star)\|.$$

By utilizing the argument in Lemma 4.2, we see that

$$\|Z - (U^\star)(U^\star)^* Z (U^\star)(U^\star)^*\|_{\mathrm{F}} \leq \left(\frac{\epsilon}{\lambda_{\min>0}(X_\star)} + \sqrt{\frac{2\epsilon}{\lambda_{\min>0}(X_\star)} \|Z\|_{\mathrm{op}}} \right).$$

We also have

$$\mathcal{D}(Z - Z(y_\star), y - y_\star) = (\mathcal{D}Z, 0) = Z - (U^\star)(U^\star)^* Z (U^\star)(U^\star)^*.$$

Combining the above pieces, we get the results in Lemma 3.1. ∎

Appendix B. Lemmas from Section 4.

LEMMA B.1. *Suppose* $Y = \begin{bmatrix} A & B \\ B^* & D \end{bmatrix} \succeq 0$. *Then* $\|A\|_{\mathrm{op}} \mathbf{tr}(D) \geq \|BB^*\|_*$.

Proof. For any $\epsilon > 0$, denote $A_\epsilon = A + \varepsilon I$ and $Y_\epsilon = \begin{bmatrix} A_\epsilon & B \\ B^* & D \end{bmatrix}$. We know Y_ϵ is psd, as is its Schur complement $D - B^* A_\epsilon^{-1} B \succeq 0$ with trace $\mathbf{tr}(D) - \mathbf{tr}(A_\epsilon^{-1} BB^*) \geq 0$.

Von Neumann's lemma for A_ϵ, $BB^\top \succeq 0$ shows $\mathbf{tr}(A_\epsilon^{-1}BB^*) \geq \dfrac{1}{\|A_\epsilon\|_{\mathrm{op}}} \|BB^*\|_*$. Use this with the previous inequality to see $\mathbf{tr}(D) \geq \frac{1}{\|A_\epsilon\|_{\mathrm{op}}} \|BB^*\|_*$. Multiply by $\|A_\epsilon\|_{\mathrm{op}}$ and let $\varepsilon \to 0$ to complete the proof. ∎

Appendix C. Lemmas from Section 5. Here we estimate a bound on $\|X_\star\|_{\mathrm{op}}$ when no prior information is available.

LEMMA C.1. *Suppose* (P) *and* (D) *admit solutions and satisfy strong duality.Let* S *be feasible for* (MinFeasSDP). *Define* ϵ, T, *and* κ *as in Theorem 5.1. Define the scaled distance bound* $\phi = (1+\kappa)\sqrt{\frac{\epsilon}{T}}$ *and the infeasibility* $\delta_0 = \|\mathcal{A}_V(S) - b\|_2$. *If* $\sigma_{\min}(\mathcal{A}_V) > 0$ $T > 0$. *Then* $\|X_\star\|_{\mathrm{op}} \leq B$ *for some constant* B, *where*

$$B = \frac{1}{4}\left[\sqrt{\phi^2 + 4\left(\frac{\delta_0}{\sigma_{\min}(\mathcal{A}_V)} + \|S\|_{\mathrm{op}} \right) + 4\frac{\phi}{1+\kappa}} + \phi \right]^2. \tag{C.1}$$

Proof. Use inequality (4.10) in Lemma 4.3 to see $\|S - S_\star\|_F \leq \frac{\delta_0}{\sigma_{\min}(\mathcal{A}_V)}$ for a minimizer $S_\star$ of (MinFeasSDP). Combine this with (4.9) in Lemma 4.3 to obtain

$$\|VSV^* - X_\star\|_F \leq (1+\kappa)\left(\frac{\epsilon}{T} + \sqrt{2\frac{\epsilon}{T}\|X_\star\|_{op}}\right) + \frac{\delta_0}{\sigma_{\min}(\mathcal{A}_V)}. \tag{C.2}$$

Because $\|VSV^* - X_\star\|_{op} \geq \|X\|_{op} - \|S\|_{op}$, we further have

$$\|X_\star\|_{op} - \|S\|_{op} \leq (1+\kappa)\left(\frac{\epsilon}{T} + \sqrt{2\frac{\epsilon}{T}\|X_\star\|_{op}}\right) + \frac{\delta_0}{\sigma_{\min}(\mathcal{A}_V)}.$$

Solve the above inequality for $\|X_\star\|_{op}$ to find a formula for the bound B. ∎

Appendix D. Bounding $\|X_\star\|_*$ and Lemmas from Section 6. We can bound $\|X_\star\|_{op}$ by $\|X_\star\|_*$ via $\|X_\star\|_{op} \leq \|X_\star\|_* \leq r_\star \|X_\star\|_{op}$. It is often easy to find a bound on $\|X_\star\|_*$ in applications, for example:

1. *Nuclear norm objective.* Suppose the objective in (P) is $\|X\|_* = \mathbf{tr}(X)$. Problems using this objective include matrix completion [20], phase retrieval [22], and covariance estimation [23]. In these settings, it is generally easy to find a feasible solution or to bound the objective via a spectral method. (See [40] for matrix completion and [19] for phase retrieval.)
2. *Constant trace constraints.* Suppose the constraint $\mathcal{A}X = b$ enforces $\mathbf{tr}(X) = \alpha$ for some constant α. Problems with this constraint include Max-Cut [32], Community Detection [47], and PhaseCut in [66]. In each of these, the constraint is simply that the diagonal of X is constant: $X_{ii} = \beta_0$ for all i and some constant $\beta_0 > 0$. Then any $\alpha > n\beta_0$ serves as an upper bound. More generally, suppose the constraint is $X_{ii} \leq \beta_i, \forall i$, as in the Powerflow [6, 46] problem. Then any $\alpha > \sum_{i=1}^n \beta_i$ serves as an upper bound. (The Powerflow problem does not directly fit into our standard form (P), but a small modification of our framework can handle the problem.)

Proof of Lemma 6.1. In fact, the lemma holds as long as (P) and (D) attain their solutions, strong duality (1.1) holds, and $\alpha > \sup_{X_\star \in \mathcal{X}_\star} \|X\|_*$. Hence we adopt these weaker assumptions for the proof. Let the optimal value of the penalized dual SDP (6.1) be $d_p^\star$. We have $d_p^\star \leq p_\star$ since

$$\begin{aligned}\max_y b^*y + \alpha \min\{\lambda_{\min}(C - \mathcal{A}^*y), 0\} &= \max_y \min_{X \succeq 0, \mathbf{tr}(X) \leq \alpha} \mathbf{tr}(CX) + (b - \mathcal{A}X)^* y \\ &\leq \min_{X \succeq 0, \mathbf{tr}(X) \leq \alpha} \max_y \mathbf{tr}(CX) + (b - \mathcal{A}X)^* y \\ &= \min_{X \succeq 0, \mathbf{tr}(X) \leq \alpha, \mathcal{A}X = b} \mathbf{tr}(CX).\end{aligned}$$

The problem $\min_{X \succeq 0, \mathbf{tr}(X) \leq \alpha, \mathcal{A}X = b} \mathbf{tr}(CX)$ in the last line is the same as the primal SDP (P) as we assume $\alpha > \sup_{X \in \mathcal{X}_\star}$. Thus if $y \in \mathcal{Y}_\star$, y is also a solution of the dual SDP with penalty (6.1) by strong duality $d_\star = p_\star$ and $d_\star \leq d_p^\star \leq p_\star$. We are left to show that if y solves the penalized dual SDP (6.1), it is also feasible for the dual SDP (D). If y is feasible for (D), it is also optimal, since $d_\star = d_p^\star$.

Suppose $y \in \mathcal{Y}$ and let $Z = C - \mathcal{A}^*y$. We claim that $\lambda_{\min}(Z) = 0$. Otherwise, $\lambda_{\min}(Z) < 0$, and by optimality of y,

$$0 \in -b + \alpha \mathcal{A}\big[\partial(\lambda_{\max})(-Z)\big]. \tag{D.1}$$

Using $(\partial(\lambda_{\max})(-Z)) = \mathbf{conv}(\{vv^T \mid (-Z)v = \lambda_{\max}(-Z)v\})$ and the previous inclusion, there exists an $W \in \partial(\lambda_{\max})(-Z))$ with $W \succeq 0, \mathbf{tr}(W) = 1$ such that $\mathcal{A}(\alpha W) = b$. Hence αW is feasible for the primal SDP (P). It is also optimal to the primal SDP (P) since the duality gap between αW and y is

$$\begin{aligned} \mathbf{tr}(C\alpha W) - (b^T y + \alpha\lambda_{\min}(Z)) &= \mathbf{tr}(C\alpha W) - (\mathcal{A}(\alpha W))^T y - \alpha\lambda_{\min}(Z) \\ &= \mathbf{tr}((C - \mathcal{A}^* y)\alpha W) - \alpha\lambda_{\min}(Z) \\ &= \alpha\lambda_{\min}(Z) - \alpha\lambda_{\min}(Z) \\ &= 0 \end{aligned} \tag{D.2}$$

But $\|W\|_* = \mathbf{tr}(W) = \alpha > \sup_{X\in\mathcal{X}_\star} \|X\|_*$ as v_i are orthonormal. This contradicts our assumption on α. Hence the claim is proved. □

We present one lemma to bound the infeasibility of a dual vector y, and another to show how to construct a feasible y from an infeasible one.

Lemma D.1. *Suppose* (P) *and* (D) *admit solutions and satisfy strong duality, Equation (1.1). Let* $\underset{\sim}{\alpha} := \inf_{X\in\mathcal{X}_\star} \mathbf{tr}(X)$. *For any dual vector* y *with suboptimality* $\mathbf{tr}(CX_\star) - g_\alpha(y) \le \epsilon$ *with* $\alpha > \underset{\sim}{\alpha}$, *we have* $\lambda_{\min}(Z(y)) \ge -\dfrac{\epsilon}{\alpha - \underset{\sim}{\alpha}}$.

This lemma shows infeasibility decreases at the same speed as suboptimality.

Proof. Let $Z = C - \mathcal{A}^* y$. Assume $\lambda_{\min}(Z) < 0$. (Otherwise, we are done.) Then

$$\begin{aligned} \mathbf{tr}(CX_\star) - g_\alpha(y) &= \mathbf{tr}(CX_\star) - b^* y - \alpha\lambda_{\min}(Z) \\ &= \mathbf{tr}(CX_\star) - (\mathcal{A}X_\star)^\top y - \alpha\lambda_{\min}(Z) \\ &= \mathbf{tr}(ZX_\star) - \alpha\lambda_{\min}(Z). \end{aligned} \tag{D.3}$$

Using the suboptimality assumption and Von Neumann's inequality, we further have

$$\epsilon \ge \mathbf{tr}(ZX_\star) - \alpha\lambda_{\min}(Z) \ge \mathbf{tr}(X_\star)\lambda_{\min}(Z) - \alpha\lambda_{\min}(Z). \tag{D.4}$$

Rearrange to see $\lambda_{\min}(Z) \ge -\dfrac{\epsilon}{\alpha - \mathbf{tr}(X_\star)}$. Let $\mathbf{tr}(X_\star) \to \underset{\sim}{\alpha}$ to obtain the result. □

We next show how to construct an ϵ-suboptimal and feasible dual vector from an ϵ-suboptimal and potentially infeasible dual vector.

Lemma D.2. *Suppose* (P) *and* (D) *admit solutions and satisfy strong duality, Equation (1.1). Further suppose a dual vector* y_1 *with* $Z_1 = C - \mathcal{A}^* y_1$ *is infeasible with* $-\epsilon \le \lambda_{\min}(Z_1) < 0$ *and* y_2 *with* $Z_2 = C - \mathcal{A}^* y_2$ *is strictly feasible in the sense that* $\lambda_{\min}(Z_2) > 0$, *then the dual vector*

$$y_\gamma = \gamma y_1 + (1-\gamma) y_2$$

is feasible for $\gamma = \dfrac{\lambda_{\min}(Z_2)}{\epsilon + \lambda_{\min}(Z_2)}$. *The objective value of* y_γ *is*

$$g_\alpha(y_\gamma) = \frac{\lambda_{\min}(Z_2)}{\epsilon + \lambda_{\min}(Z_2)} b^* y_1 + \frac{\epsilon}{\epsilon + \lambda_{\min}(Z_2)} b^* y_2$$

Proof. The results follow from the linearity of $C - \mathcal{A}^* y$ and the concavity of $\lambda_{\min}$. □

Acknowledgments. MU and LD was supported in part by DARPA Award FA8750-17-2-0101. JAT was supported in part by ONR Awards No. N-00014-11-1002, N-00014-17-12146, and N-00014-18-12363. VC and AY have received funding for this project from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement no 725594-time-data), and from the Swiss National Science Foundation (SNSF) under grant number 200021_178865/1. Parts of this research were conducted while MU was in residence at the Simons Institute.

REFERENCES

[1] *The university of florida sparse matrix collection: Gset group*, https://www.cise.ufl.edu/research/sparse/matrices/Gset/index.html (accessed 2015-06-10).

[2] F. Alizadeh, *Combinatorial optimization with interior point methods and semi-definite matrices*, Ph. D. thesis, University of Minnesota, (1991).

[3] F. Alizadeh, *Interior point methods in semidefinite programming with applications to combinatorial optimization*, SIAM journal on Optimization, 5 (1995), pp. 13–51.

[4] F. Alizadeh, J.-P. A. Haeberly, and M. L. Overton, *Complementarity and nondegeneracy in semidefinite programming*, Mathematical programming, 77 (1997), pp. 111–128.

[5] F. Alizadeh, J.-P. A. Haeberly, and M. L. Overton, *Primal-dual interior-point methods for semidefinite programming: convergence rates, stability and numerical results*, SIAM Journal on Optimization, 8 (1998), pp. 746–768.

[6] X. Bai, H. Wei, K. Fujisawa, and Y. Wang, *Semidefinite programming for optimal power flow problems*, International Journal of Electrical Power & Energy Systems, 30 (2008), pp. 383–392.

[7] A. Barvinok, *A course in convexity*, vol. 54, American Mathematical Soc., 2002.

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

[9] A. Beck and M. Teboulle, *A fast iterative shrinkage-thresholding algorithm for linear inverse problems*, SIAM journal on imaging sciences, 2 (2009), pp. 183–202.

[10] R. Bellman and K. Fan, *On systems of linear inequalities in hermitian matrix variables*, Convexity, 7 (1963), pp. 1–11.

[11] A. Ben-Tal and A. Nemirovski, *Lectures on modern convex optimization: analysis, algorithms, and engineering applications*, vol. 2, Siam, 2001.

[12] S. J. Benson, Y. Ye, and X. Zhang, *Solving large-scale sparse semidefinite programs for combinatorial optimization*, SIAM Journal on Optimization, 10 (2000), pp. 443–461.

[13] N. Boumal, V. Voroninski, and A. Bandeira, *The non-convex burer-monteiro approach works on smooth semidefinite programs*, in Advances in Neural Information Processing Systems, 2016, pp. 2757–2765.

[14] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, *Linear matrix inequalities in system and control theory*, vol. 15, Siam, 1994.

[15] S. Boyd, N. Parikh, E. Chu, B. Peleato, J. Eckstein, et al., *Distributed optimization and statistical learning via the alternating direction method of multipliers*, Foundations and Trends® in Machine learning, 3 (2011), pp. 1–122.

[16] S. Boyd and L. Vandenberghe, *Convex optimization*, Cambridge university press, 2004.

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

[18] S. Burer and R. D. Monteiro, *Local minima and convergence in low-rank semidefinite programming*, Mathematical Programming, 103 (2005), pp. 427–444.

[19] E. J. Candes, Y. C. Eldar, T. Strohmer, and V. Voroninski, *Phase retrieval via matrix completion*, SIAM review, 57 (2015), pp. 225–251.

[20] E. J. Candès and B. Recht, *Exact matrix completion via convex optimization*, Foundations of Computational mathematics, 9 (2009), p. 717.

[21] E. J. Candès and T. Tao, *The power of convex relaxation: Near-optimal matrix completion*, IEEE Transactions on Information Theory, 56 (2010), pp. 2053–2080.

[22] A. Chai, M. Moscoso, and G. Papanicolaou, *Array imaging using intensity-only measurements*, Inverse Problems, 27 (2010), p. 015005.

[23] Y. Chen, Y. Chi, and A. J. Goldsmith, *Exact and stable covariance estimation from quadratic sampling via convex programming*, IEEE Transactions on Information Theory, 61 (2015), pp. 4034–4059.

[24] K. L. Clarkson, *Coresets, sparse greedy approximation, and the frank-wolfe algorithm*, ACM Transactions on Algorithms (TALG), 6 (2010), p. 63.

[25] S. Diamond and S. Boyd, *Matrix-free convex optimization modeling*, in Optimization and its

applications in control and data sciences, Springer, 2016, pp. 221–264.
[26] D. Drusvyatskiy and A. S. Lewis, *Error bounds, quadratic growth, and linear convergence of proximal methods*, Mathematics of Operations Research, (2018).
[27] J. Duchi, E. Hazan, and Y. Singer, *Adaptive subgradient methods for online learning and stochastic optimization*, Journal of Machine Learning Research, 12 (2011), pp. 2121–2159.
[28] M. Frank and P. Wolfe, *An algorithm for quadratic programming*, Naval research logistics quarterly, 3 (1956), pp. 95–110.
[29] M. P. Friedlander and I. Macedo, *Low-rank spectral optimization via gauge duality*, SIAM Journal on Scientific Computing, 38 (2016), pp. A1616–A1638.
[30] D. Gabay and B. Mercier, *A dual algorithm for the solution of non linear variational problems via finite element approximation*, Institut de recherche d'informatique et d'automatique, 1975.
[31] R. Glowinski and A. Marroco, *Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problèmes de dirichlet non linéaires*, Revue française d'automatique, informatique, recherche opérationnelle. Analyse numérique, 9 (1975), pp. 41–76.
[32] 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 (1995), pp. 1115–1145.
[33] N. Halko, P. G. Martinsson, and J. A. Tropp, *Finding structure with randomness: probabilistic algorithms for constructing approximate matrix decompositions*, SIAM Rev., 53 (2011), pp. 217–288.
[34] E. Hazan, *Sparse approximate solutions to semidefinite programs*, in Latin American Symposium on Theoretical Informatics, Springer, 2008, pp. 306–316.
[35] C. Helmberg and F. Rendl, *A spectral bundle method for semidefinite programming*, SIAM Journal on Optimization, 10 (2000), pp. 673–696.
[36] S. Homer and M. Peinado, *Design and performance of parallel and distributed approximation algorithms for maxcut*, Journal of Parallel and Distributed Computing, 46 (1997), pp. 48–61.
[37] M. Jaggi, *Revisiting frank-wolfe: Projection-free sparse convex optimization.*, in ICML (1), 2013, pp. 427–435.
[38] P. R. Johnstone and P. Moulin, *Faster subgradient methods for functions with h\" olderian growth*, arXiv preprint arXiv:1704.00196, (2017).
[39] L. K. Jones et al., *A simple lemma on greedy approximation in hilbert space and convergence rates for projection pursuit regression and neural network training*, The annals of Statistics, 20 (1992), pp. 608–613.
[40] R. H. Keshavan, A. Montanari, and S. Oh, *Matrix completion from a few entries*, IEEE transactions on information theory, 56 (2010), pp. 2980–2998.
[41] J. Kuczyński and H. Woźniakowski, *Estimating the largest eigenvalue by the power and lanczos algorithms with a random start*, SIAM journal on matrix analysis and applications, 13 (1992), pp. 1094–1122.
[42] A. Lemon, A. M.-C. So, Y. Ye, et al., *Low-rank semidefinite programming: Theory and applications*, Foundations and Trends® in Optimization, 2 (2016), pp. 1–156.
[43] E. S. Levitin and B. T. Polyak, *Constrained minimization methods*, Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki, 6 (1966), pp. 787–823.
[44] K. Levy, *Online to offline conversions, universality and adaptive minibatch sizes*, in Advances in Neural Information Processing Systems, 2017, pp. 1613–1622.
[45] K. Y. Levy, A. Yurtsever, and V. Cevher, *Online adaptive methods, universality and acceleration*, arXiv preprint arXiv:1809.02864, (2018).
[46] R. Madani, S. Sojoudi, and J. Lavaei, *Convex relaxation for optimal power flow problem: Mesh networks*, IEEE Transactions on Power Systems, 30 (2015), pp. 199–211.
[47] C. Mathieu and W. Schudy, *Correlation clustering with noisy input*, in Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms, Society for Industrial and Applied Mathematics, 2010, pp. 712–728.
[48] A. Mosek, *The mosek optimization software*, Online at http://www. mosek. com, 54 (2010), p. 5.
[49] Y. Nesterov, *Introductory lectures on convex optimization: A basic course*, vol. 87, Springer Science & Business Media, 2013.
[50] Y. Nesterov and A. Nemirovski, *Self-concordant functions and polynomial time methods in convex programming, ussr acad*, Sci., Central Economic&Mathematical Institute, Moscow, (1989).
[51] Y. Nesterov and A. Nemirovskii, *Interior-point polynomial algorithms in convex program-*

ming, vol. 13, Siam, 1994.
[52] B. O'Donoghue, E. Chu, N. Parikh, and S. Boyd, *Conic optimization via operator splitting and homogeneous self-dual embedding*, Journal of Optimization Theory and Applications, 169 (2016), pp. 1042–1068, http://stanford.edu/~boyd/papers/scs.html.
[53] B. O'Donoghue, E. Chu, N. Parikh, and S. Boyd, *SCS: Splitting conic solver, version 2.0.2*. https://github.com/cvxgrp/scs, Nov. 2017.
[54] G. Pataki, *On the rank of extreme matrices in semidefinite programs and the multiplicity of optimal eigenvalues*, Mathematics of operations research, 23 (1998), pp. 339–358.
[55] N. Rao, P. Shah, and S. Wright, *Conditional gradient with enhancement and truncation for atomic-norm regularization*, in NIPS workshop on Greedy Algorithms, 2013.
[56] B. Recht, M. Fazel, and P. A. Parrilo, *Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization*, SIAM review, 52 (2010), pp. 471–501.
[57] J. Renegar, *Efficient first-order methods for linear programming and semidefinite programming*, arXiv preprint arXiv:1409.5832, (2014).
[58] R. T. Rockafellar, *Convex Analysis*, Citeseer, 1970.
[59] A. P. Ruszczyński and A. Ruszczynski, *Nonlinear optimization*, vol. 13, Princeton university press, 2006.
[60] A. M.-C. So, Y. Ye, and J. Zhang, *A unified theorem on sdp rank reduction*, Mathematics of Operations Research, 33 (2008), pp. 910–920.
[61] N. Srebro and A. Shraibman, *Rank, trace-norm and max-norm*, in International Conference on Computational Learning Theory, Springer, 2005, pp. 545–560.
[62] J. F. Sturm, *Error bounds for linear matrix inequalities*, SIAM Journal on Optimization, 10 (2000), pp. 1228–1248.
[63] M. J. Todd, *Semidefinite optimization*, Acta Numerica, 10 (2001), pp. 515–560.
[64] L. N. Trefethen and D. Bau III, *Numerical linear algebra*, vol. 50, Siam, 1997.
[65] L. Vandenberghe and S. Boyd, *Semidefinite programming*, SIAM review, 38 (1996), pp. 49–95.
[66] I. Waldspurger, A. dAspremont, and S. Mallat, *Phase recovery, maxcut and complex semidefinite programming*, Mathematical Programming, 149 (2015), pp. 47–81.
[67] I. Waldspurger and A. Waters, *Rank optimality for the burer-monteiro factorization*, arXiv preprint arXiv:1812.03046, (2018).
[68] Y. Yu, T. Wang, and R. J. Samworth, *A useful variant of the davis–kahan theorem for statisticians*, Biometrika, 102 (2014), pp. 315–323.
[69] A. Yurtsever, O. Fercoq, and V. Cevher, *A conditional gradient-based augmented lagrangian framework*, arXiv preprint arXiv:1901.04013, (2019).
[70] A. Yurtsever, O. Fercoq, F. Locatello, and V. Cevher, *A conditional gradient framework for composite convex minimization with applications to semidefinite programming*, arXiv preprint arXiv:1804.08544, (2018).
[71] A. Yurtsever, Y.-P. Hsieh, and V. Cevher, *Scalable convex methods for phase retrieval*, in Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP), 2015 IEEE 6th International Workshop on, IEEE, 2015, pp. 381–384.
[72] A. Yurtsever, M. Udell, J. Tropp, and V. Cevher, *Sketchy decisions: Convex low-rank matrix optimization with optimal storage*, in Artificial Intelligence and Statistics, 2017, pp. 1188–1196.