Accelerating Certifiable Estimation with Preconditioned Eigensolvers

David M. Rosen

Abstract—Convex (specifically semidefinite) relaxation provides a powerful approach to constructing robust machine perception systems, enabling the recovery of certifiably globally optimal solutions of challenging estimation problems in many practical settings. However, solving the large-scale semidefinite relaxations underpinning this approach remains a formidable computational challenge. A dominant cost in many state-of-the-art (Burer-Monteiro factorization-based) certifiable estimation methods is solution verification (testing the global optimality of a given candidate solution), which entails computing a minimum eigenpair of a certain symmetric certificate matrix. In this letter, we show how to significantly accelerate this verification step, and thereby the overall speed of certifiable estimation methods. First, we show that the certificate matrices arising in the Burer-Monteiro approach generically possess spectra that make the verification problem expensive to solve using standard iterative eigenvalue methods. We then show how to address this challenge using preconditioned eigensolvers; specifically, we design a specialized solution verification algorithm based upon the locally optimal block preconditioned conjugate gradient (LOBPCG) method together with a simple yet highly effective algebraic preconditioner. Experimental evaluation on a variety of simulated and real-world examples shows that our proposed verification scheme is very effective in practice, accelerating solution verification by up to 280x, and the overall Burer-Monteiro method by up to 16x, versus the standard Lanczos method when applied to relaxations derived from large-scale SLAM benchmarks.

Index Terms—Optimization and optimal control, probabilistic inference, SLAM.

I. INTRODUCTION

MANY fundamental machine perception tasks require the solution of a high-dimensional nonconvex estimation problem; this class includes (for example) the fundamental problems of simultaneous localization and mapping (SLAM) in robotics, 3D reconstruction (in computer vision), and sensor network localization (in distributed sensing), among others. Such problems are known to be computationally hard to solve in general [1], with many local minima that can entrap the smooth local optimization methods commonly applied to solve them [2]. The result is that traditional machine perception algorithms (based upon local optimization) can be surprisingly brittle, often returning egregiously wrong answers even when the problem to which they are applied is well-posed.

Recent work has shown that convex (specifically semidefinite) relaxation provides a powerful approach to (heuristic) local search for solving strongly robust machine perception systems, enabling the recovery of provably globally optimal solutions of generally-intractable estimation problems in many practical settings (cf. [1] generally). However, despite the great promise of this approach, solving the resulting semidefinite relaxations remains a significant computational challenge [3].

One of the most successful strategies to date for solving the large-scale semidefinite relaxations arising in machine perception applications is Burer-Monteiro factorization [4]. In brief, this method searches for a solution to a (high-dimensional) semidefinite program by alternately applying fast local optimization to a surrogate nonconvex (but low-dimensional) problem, and then testing whether the recovered critical point is a global optimum for the original SDP (which entails calculating a minimum eigenpair of a certain symmetric certificate matrix [cf. Section II]). Surprisingly, verifying a candidate solution is frequently the dominant cost in the Burer-Monteiro approach, often requiring an order of magnitude more computational effort (using standard iterative eigenvalue methods) than the entire nonlinear optimization required to produce it.

In this paper we show how to significantly accelerate this rate-limiting verification step, and thereby the overall speed of current state-of-the-art certifiable estimation methods. First, we show that the certificate matrices arising in the Burer-Monteiro approach generically possess spectra that make it expensive to compute a minimum eigenpair using standard iterative eigenvalue methods. Next, we show how to address this challenge using preconditioned eigensolvers; specifically, we design a specialized verification algorithm based upon the locally optimal block preconditioned conjugate gradient (LOBPCG) [5] method together with a simple yet highly effective algebraic preconditioner. Experimental evaluation on a variety of synthetic and real-world examples demonstrates that our proposed verification approach is very effective in practice, accelerating solution verification by up to 280x, and the overall Burer-Monteiro method by up to 16x, versus a standard Lanczos method when applied to relaxations derived from large-scale SLAM benchmarks.

II. SEMIDEFINITE OPTIMIZATION USING THE BURER-MONTEIRO METHOD

In this section we briefly review the Burer-Monteiro method for semidefinite optimization [4], which forms the algorithmic foundation of many state-of-the-art certifiable estimation methods (cf. [1] and the references therein). We begin with
a brief review of semidefinite programs and a discussion of their computational cost. We then describe the Burer-Monteiro method, which provides a scalable approach to solving SDPs with low-rank solutions. Finally, we discuss the verification problem (Problem 1) arising in the Burer-Monteiro approach, and identify a key feature that makes this problem expensive to solve using standard iterative eigenvalue methods.

A. Semidefinite Programs

Recall that a semidefinite program (SDP) is a convex optimization problem of the form [6]:

\[
\min_{X \in S_p^n} \text{tr}(CX) \quad \text{s.t. } \mathcal{A}(X) = b,
\]

where \( C \in S^n \), \( b \in \mathbb{R}^m \), and \( \mathcal{A} : S^n \rightarrow \mathbb{R}^m \) is a linear map:

\[
\mathcal{A}(X) \triangleq \text{tr}(A_iX) \quad \forall i \in [m]
\]

parameterized by coefficient matrices \( \{A_i\}_{i=1}^m \subseteq S^n \). Problem (1) can in principle be solved efficiently (i.e. in polynomial time) using general-purpose (e.g. interior point) optimization methods [6], [7]. In practice, however, the high computational cost of storing and manipulating the \( O(n^2) \) elements of the decision variable \( X \) prevents such straightforward approaches from scaling effectively to problems in which \( n \) is larger than a few thousand [3]. Unfortunately, typical instances of many important machine perception problems (e.g. SLAM) are often several orders of magnitude larger, placing them well beyond the reach of such general-purpose techniques.

B. Burer-Monteiro Factorization

While problem (1) is expensive to solve in general, the specific instances of (1) arising as relaxations in certifiable estimation often admit low-rank solutions. Any such solution \( X^* \in S_+^p \) can be represented concisely as \( X^* = Y^* Y^* \), where \( Y^* \in \mathbb{R}^{n \times r} \) and \( r = \text{rank } X^* \ll n \).

In their seminal work, Burer and Monteiro [4] proposed to exploit the existence of such low-rank solutions by replacing the high-dimensional decision variable \( X \) in (1) with its rank-\( r \) symmetric factorization \( YY^\top \), producing the following (nonconvex) rank-restricted version of the problem:

\[
\min_{Y \in \mathbb{R}^{n \times r}} \text{tr}(CYY^\top) \quad \text{s.t. } \mathcal{A}(YY^\top) = b.
\]

This substitution has the effect of dramatically reducing the dimensionality of the search space of (3) versus (1), as well as rendering the positive-semidefiniteness constraint in (1) redundant (since \( YY^\top \succeq 0 \) by construction). Burer and Monteiro [4] thus proposed to search for minimizers \( X^* \) of (1) by attempting to recover their low-rank factors \( Y^* \) from (3) using fast local nonlinear programming methods [7].

C. Ensuring Global Optimality

Since the Burer-Monteiro approach entails searching for a global minimizer \( Y^* \) of the nonconvex problem (3), one might naturally wonder whether it is possible to find such global optima in practice. Remarkably, recent work has shown that this can in fact be done under quite general conditions, despite problem (3)’s nonconvexity [8], [9].

Suppose that we have applied a numerical optimization method to an instance of problem (3), and recovered a first-order stationary point \( Y \in \mathbb{R}^{n \times r} \). Then either \( X = YY^\top \) is a global minimizer of (1) (which we can determine by checking the necessary and sufficient KKT conditions for the convex program (1)), or \( X \) is suboptimal, in which case we would like to identify a direction of improvement for the low-rank factor \( Y \). The following result establishes that checking the optimality of \( X = YY^\top \) and constructing a direction of improvement (if necessary), can both be accomplished with a single minimum-eigenpair calculation:

**Theorem 1 (Theorem 4 of [10]):** Assume that the semidefinite program (1) satisfies Slater’s condition. Let \( Y \in \mathbb{R}^{n \times r} \) be a KKT point of (3) that satisfies the linear independence constraint qualification, \( \lambda \in \mathbb{R}^m \) the corresponding Lagrange multiplier, and define:

\[
S \triangleq C + \sum_{i=1}^m \lambda_i A_i.
\]

Then exactly one of the following two cases holds:

i) \( S \succeq 0 \), and \( X = YY^\top \) is a global minimizer of (1).

ii) There exists \( v \in \mathbb{R}^n \) such that \( v^\top S v < 0 \), and in that case, \( Y_+ = \begin{pmatrix} Y & 0 \end{pmatrix} \in \mathbb{R}^{n \times (r+1)} \) is a KKT point of (3) attaining the same objective value as \( Y \), and \( Y_+ = \begin{pmatrix} 0 & v \end{pmatrix} \in \mathbb{R}^{n \times (r+1)} \) is a feasible second-order direction of descent from \( Y_+ \).

In brief, Theorem 1 establishes that the global optimality of \( X = YY^\top \) as a solution of (1) can be verified by checking whether the certificate matrix \( S \) defined in (4) is positive semidefinite, and in the event that it is not, that any direction of negative curvature \( v \in \mathbb{R}^n \) of \( S \) provides a direction of improvement \( Y_+ \) from the embedding \( Y_+ \) of the current critical point \( Y \) into a higher-dimensional instance of (3).

This theorem of the alternative suggests a natural algorithm for recovering minimizers \( X^* \) of (1) via a sequence of local optimizations of (3) [10], [11], [12]. Starting at some (small) initial rank \( r \), apply local optimization to (3) to obtain a stationary point \( Y \). If \( X = YY^\top \) is optimal for (1), then return \( Y \); otherwise, increment the rank parameter \( r \), and restart the local optimization for the next instance of (3) using the direction of improvement \( Y_+ \) provided by Theorem 1.

D. The Solution Verification Problem

Surprisingly, simply verifying the optimality of a candidate solution \( X = YY^\top \) (by calculating a minimum eigenpair of the certificate matrix \( S \) in (4) and applying Theorem 1) is frequently the dominant cost in the Burer-Monteiro approach, often requiring an order of magnitude more computational effort than the entire nonlinear optimization (3) required to produce \( Y \) itself (cf. Section V). This turns out to be a consequence of a particular feature of the spectra of the certificate matrices \( S \) arising in the Burer-Monteiro method.
Remark 1 (The spectrum of the certificate matrix): A straightforward calculation shows that the first-order stationarity conditions for (3) can be expressed as (cf. [10, Th. 2]):

\[ SY = 0. \]  

(5)

This implies that \( S \) will have 0 as an eigenvalue of multiplicity at least \( \text{rank} \ Y \); that is, \( S \) always has a cluster of eigenvalues at 0. It is this eigenvalue clustering that makes calculating a minimum eigenpair of \( S \) expensive. In detail, if \( S \gtrsim 0 \), then \( \lambda_{\text{min}}(S) = 0 \), so the target eigenvalue is embedded in a tight cluster, while if \( \lambda_{\text{min}}(S) < 0 \) but has small magnitude, it is difficult to estimate accurately because the cluster of 0 eigenvalues nearby ensures that the relevant eigenvalue gap \( \gamma \leq |\lambda_{\text{min}}(S) - 0| = |\lambda_{\text{min}}(S)| \) is small. In effect, (5) guarantees that the minimum-eigenpair calculation is almost always poorly-conditioned, and therefore expensive to solve using standard iterative eigenvalue methods (cf. Section III).

In light of Theorem 1 and Remark 1, we are thus interested in developing more computationally efficient methods for solving the following (solution) verification problem:

**Problem 1 (Verification problem):** Given a matrix \( S \in S^n \), determine whether \( S \gtrsim 0 \), and if it is not, calculate a vector \( x \in \mathbb{R}^n \) such that \( x^T S x < 0 \).

III. RELATED WORK

In this section we review prior work pertaining to the solution of the verification problem (Problem 1) from two areas: algorithms for the symmetric eigenvalue problem (Section III-A), and certifiable estimation (Section III-B).

A. Algorithms for the Symmetric Eigenvalue Problem

Standard large-scale eigenvalue algorithms are iterative projection methods, meaning that they recover approximate eigenpairs of a matrix \( A \in S^n \) by searching for approximate eigenvectors within a (computationally tractable) low-dimensional subspace of \( \mathbb{R}^n \) [13, Ch. 3]. This includes the power and subspace iteration methods (for computing dominant eigenpairs), and the Lanczos method, which is the preferred technique for computing a small number of extremal eigenpairs of \( A \) [13, Ch. 4]. An important property of these methods is that their convergence rates depend crucially upon the distribution of \( A \)'s eigenvalues, and in particular how well-separated a target eigenvalue is from the remainder of \( A \)'s spectrum (relative to \( A \)'s spectral norm).

For example, letting \( \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_K \) denote \( A \)'s distinct eigenvalues, the number of iterations required to estimate a minimum eigenvector \( x_1 \) of \( A \) using the Lanczos method is approximately \( O(\sqrt{\|A\|/\gamma}) \), where \( \gamma \triangleq \lambda_2 - \lambda_1 \) is the spectral gap between \( A \)'s minimum eigenvalue \( \lambda_1 \) and the next-largest eigenvalue \( \lambda_2 \) (cf. [14, Th. 6.3] and the discussion in [15, Section 6.1]). Consequently, computing a minimum eigenpair of \( A \) using the Lanczos method is expensive when \( \gamma \ll \|A\| \).

Unfortunately, as shown in Section II-D, this case generically holds for the verification problems arising in the Burer-Monteiro approach.

Given the sensitivity of iterative eigenvalue methods to the distribution of \( A \)'s spectrum, a natural idea for improving computational performance is to precondition the eigenvalue problem by applying a spectral transformation that sends the desired eigenvalue \( \lambda \) of \( A \) to a well-separated extremal eigenvalue [13, Sec. 3.3]. The most common such method is the shift-and-invert (SI) spectral transformation, in which the matrix \( A \) is replaced by the resolvent \( R_\sigma \triangleq (A - \sigma I)^{-1} \) for \( \sigma \notin \lambda(A) \). This transformation maps each eigenpair \( (\lambda, x) \) of \( A \) to the eigenpair \( (\nu, x) \) of \( R_\sigma \), where \( \nu = (\lambda - \sigma)^{-1} \); in particular, it sends the eigenvalues of \( A \) nearest to the spectral shift \( \sigma \) to well-separated extremal eigenvalues of \( R_\sigma \), thus enabling their efficient recovery. Unfortunately, basic SI is insufficient to precondition the verification problem, since we do not know a priori where the desired minimum eigenvalue \( \lambda_{\text{min}}(S) \) is (indeed, determining this is part of the problem!)”

Some iterative eigenvalue methods employ what can be interpreted as adaptive SI preconditioning schemes. In brief, these methods leverage the updated eigenvalue estimates \( \theta_k \) calculated in each iteration to construct an improved SI preconditioner \( R_{\theta_k} \) for the next iteration; well-known examples of this approach include Rayleigh quotient iteration (RQI) [13, Sec. 4.3.3] and the Jacobi-Davidson (JD) method [13, Section 4.7], which can be interpreted as a kind of inexact subspace-accelerated RQI method. While these techniques promote rapid convergence, they require solving sequences of linear systems with varying coefficient matrices \( A - \theta_k I \), which in practice entails either repeated matrix factorizations or the repeated construction of preconditioners to use with iterative linear system solvers (e.g. MINRES), both of which can be dominant costs in large-scale calculations.

In light of these considerations, we propose to adopt the locally optimal block preconditioned conjugate gradient (LOBPCG) method [5] as the basis of our fast verification approach. In brief, LOBPCG is a preconditioned subspace-accelerated simultaneous iteration scheme for recovering a few algebraically-smallest eigenpairs of \( A \). An important distinguishing feature of this method is that it employs a single (constant) preconditioner, which enables preconditioning the verification problem with modest computational overhead.

B. Certifiable Estimation

In consequence of the increasing interest in certifiable estimation methods (cf. [1] and the references therein), several recent works have made use of (specific instances of) Theorem 1 to certify the global optimality of candidate solutions to machine perception problems, whether as part of a Burer-Monteiro method [10], [15], [16], [17], [18], [19] or as a standalone procedure applied to estimates recovered from more traditional heuristic local search [20], [21], [22]. These prior works solve the verification problem by applying Krylov subspace methods (typically the Lanczos method) directly to the certificate matrix \( S \) to recover a minimum eigenpair. Reference [23] proposed a simple spectral-shifting strategy to convert the verification problem to the calculation of a dominant eigenpair (for which the Lanczos method exhibited better empirical performance),
while [15] proposed to employ accelerated power iterations, which are more amenable to a distributed implementation.

To the best of our knowledge, this letter is the first to investigate the design of specialized algorithms for solving Problem 1, and in particular the first to propose the use of preconditioned eigensolvers to ameliorate the structural ill conditioning identified in Section II-D whenever the calculation of a vector \( x \in \mathbb{R}^n \) satisfying \( x^T S x < 0 \) is required.

IV. ACCELERATING SOLUTION VERIFICATION WITH PRECONDITIONED EIGENSOlVERS

In this section we present our proposed fast verification scheme. We begin in Section IV-A with a review of the LOBPCG method, which forms the basis of our approach, and highlight some of the features that make it especially well suited to solving the verification problem. We then describe our preconditioning strategy for use with LOBPCG in Section IV-B. Finally, we present our complete algorithm in Section IV-C.

A. The LOBPCG Eigensolver

The locally optimal block preconditioned conjugate gradient (LOBPCG) method [5], [24] is an iterative eigenvalue method for calculating the \( k \) algebraically-smallest eigenpairs \((\lambda, x)\) of the generalized symmetric eigenvalue problem:

\[
Ax = \lambda Bx
\]

where \( A, B \in \mathbb{S}^n \) and \( B \succ 0 \). This method exploits the Courant-Fischer variational characterization of a minimum eigenpair \((\lambda, x)\) of (6) as a solution of the following minimization problem [14, Sec. 9.2.6]:

\[
\lambda = \min_{x \in \mathbb{R}^n} x^T Ax \quad \text{s.t.} \quad x^T Bx = 1
\]

(7)

to search for \( x \) by applying first-order optimization to (7).

In detail, the Lagrangian \( L : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R} \) of (7) is:

\[
L(x, \lambda) \triangleq x^T Ax - \lambda (x^T Bx - 1)
\]

(8)

and a straightforward calculation shows that the gradient of \( L(x, \lambda) \) with respect to \( x \) is:

\[
\nabla_x L(x, \lambda) = 2Ax - 2\lambda Bx.
\]

(9)

Comparing (6) and (9) reveals that the KKT points \((x, \lambda)\) of (7) are precisely the eigenpairs of (6); furthermore, given any estimate \((\theta, x)\) for a minimum eigenpair, we see that the residual \( r \) of the generalized eigenvalue equation (6):

\[
r \triangleq Ax - \theta Bx
\]

(10)

is parallel to the gradient \( \nabla_x L(x, \theta) \) in (9). A basic first-order optimization method would thus take \(-r\) as a search direction to calculate an improved minimum eigenvector estimate [7].

LOBPCG improves upon a basic first-order optimization approach in three important ways. First and foremost, it exploits the fact that the objective in (7) is a quadratic parameterized by \( A \) by replacing the residual (gradient) search direction \( r \) in (10) with the preconditioned gradient search direction:

\[
w \triangleq Tr,
\]

(11)

where \( T \in \mathbb{S}^n_{++} \) is a user-defined preconditioning operator. This should be chosen to approximate \( A^{-1} \) in the sense that \( \kappa(TA) \ll \kappa(A) \); this has the effect of “undistorting” the contours of the quadratic objective in (7), producing significantly higher-quality search directions. This strategy has two major advantages versus the alternative SI preconditioning strategy used in e.g. Rayleigh quotient iteration and the Jacobi-Davidson method. First, the figure of merit \( \kappa(TA) \) for the preconditioner \( T \) is the same one appearing in the design of preconditioners for iterative linear system solvers (such as MINRES), which enables the direct application of this large body of prior work to the generalized eigenvalue problem (6). Second, in contrast to the varying resolvents \( R_{\theta_k} \triangleq (A - \theta_k I)^{-1} \) used in adaptive SI preconditioning, the preconditioner \( T \) used in LOBPCG is designed for the parameter matrix \( A \) appearing in (7), and is therefore constant. Since constructing a preconditioner is often done via incomplete matrix factorization (which is expensive at large scale), enabling the use of a constant preconditioner \( T \) can provide substantial computational savings.

Second, LOBPCG employs subspace acceleration: instead of searching solely over the 1-dimensional subspace spanned by \( w \) (as in basic preconditioned gradient descent), at each iteration it calculates the next eigenvector estimate \( x_{i+1} \) as a linear combination of the preconditioned gradient direction \( w_i \), the current iterate \( x_i \), and the previous iterate \( x_{i-1} \):

\[
x_{i+1} = \alpha_i w_i + \beta_i x_i + \gamma_i x_{i-1},
\]

(12)

where \( \alpha_i, \beta_i, \gamma_i \in \mathbb{R} \) are coefficients. The use of a larger (3-dimensional) search space in (12) provides more flexibility in the choice of update \( x_{i+1} \), and therefore the opportunity for enhanced performance. In particular, we remark that standard accelerated gradient methods (such as Nesterov, heavy ball, and the accelerated power method) all correspond to special cases of (12) for specific choices of \( \alpha_i, \beta_i, \) and \( \gamma_i \). Moreover, LOBPCG further improves upon standard accelerated gradient schemes by exploiting the specific form of (7) to dynamically compute optimal choices of \( \alpha_i, \beta_i, \) and \( \gamma_i \) in (12) in each iteration. Specifically, the updated estimate \( x_{i+1} \) is chosen to minimize the Rayleigh quotient over the 3-dimensional subspace spanned by \( w_i, x_i, \) and \( x_{i-1} \):

\[
x_{i+1} \in \underset{x \in \text{span}\{w_i, x_i, x_{i-1}\}}{\arg\min} x^T Ax \quad \text{s.t.} \quad x^T Bx = 1.
\]

(13)

In practice, the minimization (13) amounts to solving a projected generalized eigenvalue problem on the 3-dimensional subspace \( S = \text{span}\{w_i, x_i, x_{i-1}\} \), which can be done simply and efficiently using the Rayleigh-Ritz procedure (Algorithm 1).

Finally, LOBPCG extends the single-vector iteration described in (10)–(13) to simultaneous iteration on a block \( X \in \mathbb{R}^{n \times m} \) of \( m \) eigenvector estimates. This provides two major advantages. First, it permits the simultaneous recovery of the \( k \leq m \) algebraically-smallest eigenpairs of (6). Second, a rough analysis suggests (and numerical experience confirms) that the convergence rate of the \( k \) smallest eigenpairs in a block of size \( m \) depends upon the eigengap \( \gamma_{m,k} \triangleq \lambda_{m+1} - \lambda_k \) between the

\(^{1}\text{This is the sense in which LOBPCG is "locally optimal".}\)
Algorithm 1: The Rayleigh-Ritz Procedure.

Input: Symmetric linear operators $A, B \in \mathbb{S}^n$ with $B > 0$, full-rank basis matrix $S \in \mathbb{R}^{n \times k}$.
Output: Diagonal matrix $\Theta = \text{Diag}(\theta_1, \ldots, \theta_k) \in \mathbb{S}^k$ of Ritz values with $\theta_1 \leq \cdots \leq \theta_k$, matrix $C \in \text{GL}(k)$ satisfying $C^T (S^T B S) C = I_k$ and $C^T (S^T A S) C = \Theta$.

1: function RayleighRitz(A, B, S)
2: \hspace{1em} $D = \text{Diag}(S^T B S)^{-1/2}$.
3: \hspace{1em} $X = D W R$ \hspace{1em} \triangleright orthonormalize initial $X$
4: \hspace{1em} $R = A X - B X \Theta$ \hspace{1em} \triangleright Initial residuals
5: \hspace{1em} $P = \emptyset$ \hspace{1em} \triangleright Initialize $P$
6: repeat
7: \hspace{1em} $W = TR$ \hspace{1em} \triangleright Precondition residuals
8: \hspace{1em} $S = (X \ W \ P)$ \hspace{1em} \triangleright Search space basis
9: \hspace{1em} $(\Theta, C) = \text{RayleighRitz}(A, B, X)$
10: \hspace{1em} $X = SC_{1:m}$ \hspace{1em} \triangleright Update eigenvector estimates
11: \hspace{1em} $R = A X - B X \Theta_{1:m}$ \hspace{1em} \triangleright Compute residuals
12: \hspace{1em} $P = S_{m+1:end} C_{m+1:end, 1:m}$ \hspace{1em} \triangleright Compute residuals
13: \hspace{1em} Determine the number $n_c$ of the $k$ algebraically-smallest eigenpairs $(\theta_i, x_i)$ that have converged.
14: until $n_c = k$
15: return $(\Theta_{1:k, 1:k}, X_{1:1:k})$
16: end function

Algorithm 2: The Locally Optimal Block Preconditioned Conjugate Gradient (LOBPCG) Method.

Input: Symmetric linear operators $A, B, T$ with $B, T > 0$, initial eigenvector estimates $X \in \mathbb{R}^{n \times m}$, number of desired eigenpairs $k \leq m$.
Output: Estimates of the $k$ algebraically-smallest eigenpairs of the generalized symmetric eigenvalue problem (6).

1: function LOBPCG(A, B, T, X, k)
2: \hspace{1em} $N = \text{Diag}(A, B, X)$ \hspace{1em} \triangleright Initial orthonormalization
3: \hspace{1em} $X = X \ N$ \hspace{1em} \triangleright $B$-orthonormalize initial $X$
4: \hspace{1em} $R = A X - B X \Theta$ \hspace{1em} \triangleright Initial residuals
5: \hspace{1em} $P = \emptyset$ \hspace{1em} \triangleright Initialize $P$
6: repeat
7: \hspace{1em} $W = TR$ \hspace{1em} \triangleright Precondition residuals
8: \hspace{1em} $S = (X \ W \ P)$ \hspace{1em} \triangleright Search space basis
9: \hspace{1em} $(\Theta, C) = \text{RayleighRitz}(A, B, X)$
10: \hspace{1em} $X = SC_{1:m}$ \hspace{1em} \triangleright Update eigenvector estimates
11: \hspace{1em} $R = A X - B X \Theta_{1:m}$ \hspace{1em} \triangleright Compute residuals
12: \hspace{1em} $P = S_{m+1:end} C_{m+1:end, 1:m}$ \hspace{1em} \triangleright Compute residuals
13: \hspace{1em} Determine the number $n_c$ of the $k$ algebraically-smallest eigenpairs $(\theta_i, x_i)$ that have converged.
14: until $n_c = k$
15: return $(\Theta_{1:k, 1:k}, X_{1:1:k})$
16: end function

B. Preconditioner Design

In this section, we propose a simple yet highly effective algebraic preconditioning strategy for use with LOBPCG, based upon a modified incomplete symmetric indefinite factorization.

Our approach is based upon the following simple argument. Recall that every nonsingular matrix $A \in \mathbb{S}^n$ admits a Bunch-Kaufman factorization of the form \cite[Sec. 4.4]{26}:

$$P A P^T = L \text{Diag}(D_1, \ldots, D_K) L^T,$$

where $P$ is a permutation matrix, $L$ is unit lower-triangular, and $D_k$ is a $1 \times 1$ or $2 \times 2$ symmetric nonsingular matrix for all $k \in [K]$. Given the factorization (14), consider the matrix:

$$T \triangleq P^T L^{-T} \text{Diag} (D_1^+, \ldots, D_K^+) L^{-1} P,$$

where for any nonsingular matrix $M \in \mathbb{S}^p$ with symmetric eigendecomposition $M = Q \text{Diag}(\lambda_1, \ldots, \lambda_p) Q^T$, we define:

$$M^+ \triangleq Q \text{Diag} (|\lambda_1|^{-1}, \ldots, |\lambda_p|^{-1}) Q^T \in \mathbb{S}^p_{++}$$

(16)

to be the positive-definite matrix obtained from $M$ by replacing each of $M$’s eigenvalues with the reciprocal of its absolute value. Then $T > 0$, and a straightforward calculation yields:

$$T A = P^{-1} L^{-T} \text{Diag} (D_1^+, \ldots, D_K^+) L^{-1} P T P^T P = T^{-1} A T$$

Equation (17) shows that the product $T A$ is similar to a block-diagonal matrix whose blocks $D_k^+ D_k$ all have eigenvalues contained in $\{ \pm 1 \}$. This proves that $\Lambda(TA) \subseteq \{ \pm 1 \}$, and therefore $\kappa(TA) = 1$: that is, the matrix $T$ defined in (15) attains the smallest possible condition number $\kappa(TA)$, and is therefore an “ideal” preconditioner for use with LOBPCG.

Since computing a matrix decomposition can be expensive when $A$ is large, in practice we propose to construct our preconditioners by replacing the exact factorization (14) with a tractable approximation $P A P^T \approx \hat{L} \text{Diag}(D_1, \ldots, D_K) L^T$. Specifically, we propose to use a limited-memory incomplete symmetric indefinite LDL$^T$ factorization \cite{27}; this provides direct control over the number of nonzero elements appearing in the unit lower-triangular factors $\hat{L}$, and therefore scales gracefully (in both memory and computation) to high-dimensional problems. We construct each diagonal block $\hat{D}_k^+$ by directly computing and modifying the symmetric eigendecomposition of $\hat{D}_k$ as in (16); this is again efficient, as each block is at most 2-dimensional. Finally, we avoid explicitly instantiating $T$ as a (generically-dense) matrix. Instead, whenever we must compute a product with $T$, we sequentially apply each of the linear operators on the right-hand side of (15); this requires only 2 permutations, 2 sparse triangular solves, and a set of parallel multiplications with the (very small) diagonal blocks $\hat{D}_k^+$ per application of $T$, and is thus again efficient.

C. An Accelerated Solution Verification Method

Our complete proposed method for solving the verification problem (Problem 1) is shown as Algorithm 3. Our method first tests the positive-semidefiniteness of the regularized certificate matrix $M \triangleq S + \eta I$ (where $\eta > 0$ is a user-selectable regularization parameter) by attempting to compute its Cholesky decomposition. Note that the use of regularization here is in

largest desired eigenvalue $\lambda_k$ and the smallest eigenvalue $\lambda_{m+1}$ not included in the block (cf. the discussion around eq. (5.5) in \cite{3}). It is therefore advantageous to choose the block size $m$ greater than the number of desired eigenpairs $k$ if doing so will significantly enhance the eigengap $\gamma_{m,k}$. This capability is especially useful for addressing clustered eigenvalues, which we have seen are the primary challenge in solving the verification problem.

The complete LOBPCG algorithm is shown as Algorithm 2 (cf. Algorithms 1 and 2 of \cite{25}).
Algorithm 3: Fast Solution Verification.

Input: Certificate matrix $S \in \mathbb{S}^n$, minimum eigenvalue numerical nonnegativity tolerance $\eta > 0$, blocksize $m \geq 1$.
Output: Certificate matrix $S + \eta I = LL^T$ certifying $S \succeq -\eta I$, or a Ritz pair $(\lambda, x)$ satisfying $\lambda = x^T S x$ and $\lambda < 0$.

1: function FAST
2: \hspace{1em} Construct regularized certificate matrix $M = S + \eta I$.
3: \hspace{1em} Attempt Cholesky factorization of $M$.
4: \hspace{1em} if $M = LL^T$ then
5: \hspace{2em} return Cholesky factor $L$. \hspace{1em} $\triangleright$ $S \succeq -\eta I$
6: \hspace{1em} end if
7: Incomplete symmetric indefinite factorization:
\[ PMP^T \approx L \text{Diag}(D_1, \ldots, D_K) L^T. \]
8: Construct preconditioner $T$ as in (15).
9: Randomly sample initial block $X \in \mathbb{R}^{n \times m}$.
10: \hspace{1em} $(\theta, x) = \text{LOBPCG}(M, I, T, X, 1)$.
11: \hspace{1em} return $(\theta - \eta, x)$
end function

V. EXPERIMENTAL RESULTS

In this section we evaluate the performance of our fast verification method (Algorithm 3) on a variety of synthetic and real-world examples. All experiments are performed on a Lenovo T480 laptop with an Intel Core i7-8650U 1.90 GHz processor and 16 Gb of RAM running Ubuntu 22.04. Our experimental implementation of Algorithm 3 is written in C++, using CHOLMOD [28] to perform the Cholesky factorization in line 7.

Algorithm 4: Test Matrix Sampler.

1: function SAMPLE
2: \hspace{1em} Sample $N$ vertices distributed uniformly randomly over the unit square: $V \triangleq \{ x_i \sim U([0, 1]^2) \mid i \in [N] \}$.
3: Construct edge set by joining vertices closer than a distance $r$: $E \triangleq \{ (i, j) \mid i \neq j \text{ and } d(x_i, x_j) < r \}$.
4: Sample edge weights uniformly randomly in $[0, W_{\text{max}}]: W \triangleq \{ w_{ij} \sim U([0, W_{\text{max}}]) \mid (i, j) \in E \}$.
5: Construct weighted graph $G \triangleq (V, E, W)$.
6: Construct Laplacian matrix $L(G)$ of $G$, and set:
\[ S \triangleq \begin{pmatrix} L(G) & 0 \\ 0 & -\gamma \end{pmatrix}. \] (18)
7: return $S$
8: end function

A. Simulation Studies

Our first set of experiments is designed to study how the performance of Algorithm 3 depends upon (i) the eigenvalue gap $\gamma$ associated with the minimum eigenvalue $\lambda_{\text{min}}(S)$ (cf. Section III-A) and (ii) the dimension of $S$.

To that end, we generate a set of suitable test instances using Algorithm 4. In brief, this method constructs a matrix $S$ by randomly sampling a weighted geometric random graph $G$ on the unit square $[0, 1]^2 \subset \mathbb{R}^2$ [29], forming its graph Laplacian $L(G)$ [30], and then extending this matrix by a single row and column whose only nonzero value is the final element $-\gamma$ on the diagonal. This generative procedure is designed to capture several important properties of the certificate matrices arising in real-world instances of Problem 1, including:

- The sampled matrices are of the form $S = L(G) + E$, where $L(G)$ is the Laplacian of a weighted graph $G$ and $E$ is a sparse matrix;
- The topology of the graph $G$ captures the (spatial) locality-based connectivity typical of real-world measurement networks in robotics and computer vision [1];
- There is variation in the weights of $G$’s edges (corresponding to the variation of measurement precisions in real-world measurement networks).

Finally, since $L(G) \succeq 0$ and $0 \in \Lambda(L(G))$ (as $L(G)$ is a graph Laplacian), it follows from (18) that $\lambda_{\text{min}}(S) = -\gamma$ and the eigenvalue gap is $|\gamma - 0| = \gamma$; that is, we can directly and deterministically control the relevant eigenvalue gap $\gamma$ of the random test matrices $S$ sampled from Algorithm 4.

For the following experiments we implemented Algorithm 4 in Python, using the random_geometric_graph method in the NetworkX library to sample the graphs $G$, with default
parameters $r = 1.29 \sqrt{\log(\gamma)/\pi N}$ and $W_{\max} = 10^3$, while varying $N$ and $\gamma$. As our baseline for comparison, we apply the Lanczos method to compute a minimum eigenpair of $S$.

Both methods employ the relative-error convergence criterion:

$$\|S x - \theta x\| \leq \tau |\theta|$$

for eigenpair estimates $(\theta, x)$, with $\tau = 10^{-2}$.

In our first set of experiments, we study the effect of varying the eigenvalue gap $\gamma$. We sample 50 matrices $S$ from Algorithm 4 for each value of $\gamma = 10^k$, where $k \in \{-6, \ldots, 1\}$ (holding $N = 25000$), and then solve the verification problem (Problem 1) using Algorithm 3 (both with and without preconditioning) and the Lanczos method. We record the elapsed computation time for each method, as well as the number of iterations required by LOBPCG in each version of Algorithm 3. Results from this experiment are shown in Figs. 1(a) and (b).

These results demonstrate that using LOBPCG together with the preconditioner proposed in Section IV-B renders Algorithm 3 essentially insensitive to the eigenvalue gap $\gamma$ (note that the blue curves in Figs. 1(a) and (b) are effectively flat). In fact, we observe that Algorithm 3 actually achieves a significant gain in performance for $\gamma \lesssim \eta$, since in that case we can immediately detect $S$’s (numerical) positive-semidefiniteness using the test in lines 3–6. This small-$\gamma$ regime is especially important because (as discussed in Section III) it is the most challenging for standard iterative eigenvalue methods, and in the context of the

Burer-Monteiro approach is precisely the case that corresponds to certifying an optimal solution.

In our second set of experiments, we study the effect of varying the problem size $N$. As before, we sample 50 realizations of $S$ from Algorithm 4 for each value of $N = 5000k$, where $k \in \{1, \ldots, 10\}$, and then solve the verification problem (Problem 1) using Algorithm 3 (again both with and without preconditioning) and the Lanczos method. Results from this experiment are shown in Figs. 1(c) and (d).

These results show that the computational cost of Algorithm 3 (as measured in both elapsed time and LOBPCG iterations) scales gracefully with the size of the verification problem. In particular, Fig. 1(c) shows that Algorithm 3’s running time increases approximately linearly with $N$ (note the logarithmic axis), which is consistent with the (approximately linearly) increasing cost of performing sparse matrix-vector multiplications and triangular solves with the operators $A$ and $T$.

### B. SLAM Benchmarks

In this experiment we evaluate the impact of our fast verification approach (Algorithm 3) on the total computational cost of the Burer-Monteiro method. To do so, we solve a sequence of large-scale semidefinite relaxations obtained from a standard suite of SLAM benchmarks (see [16] for details). For each problem instance, we record the total time required to perform the local nonlinear optimizations (3), as well as the total time required to solve the subsequent verification problems (Problem 1) using (i) our proposed fast verification method (Algorithm 3) and (ii) the spectrally-shifted Lanczos approach proposed in [23]. Results of this experiment are shown in Fig. 2.

These results clearly show that the verification step is the dominant cost of the Burer-Monteiro approach when using an unpreconditioned Krylov-subspace method, often requiring an order of magnitude more computational effort than nonlinear optimization. Our fast verification procedure significantly accelerates this rate-limiting step in all tested cases, achieving a
relative gain in speed of between 4.08 (on torus) and 280 (on ais2klklinik) for the verification problems, and a corresponding gain of between 1.21 (on sphere) and 16.9 (on csail) for the end-to-end Burer-Monteiro method. Crucially, solution verification is no longer the dominant cost when using our fast verification scheme [note that the blue bars are uniformly taller than the green in Fig. 2(a)].

VI. CONCLUSION

In this letter we showed how to significantly accelerate solution verification in the Burer-Monteiro method, the rate-limiting step in many state-of-the-art certifiable perception algorithms. We first established that the certificate matrices arising in the verification problem generically possess spectra that make it expensive to solve using standard iterative eigenvalue methods. Next, we showed how to overcome this challenge using preconditioned eigensolvers; specifically, we proposed a specialized solution verification algorithm based upon applying LOBPCG together with a simple yet highly effective incomplete factorization-based preconditioner. Experimental evaluation confirms that our proposed verification scheme is very effective in practice, accelerating solution verification by up to 280x, and the overall Burer-Monteiro method by up to 16x, versus the Lanczos method when applied to relaxations derived from large-scale SLAM benchmarks.

REFERENCES

[1] D. Rosen, K. Doherty, A. Terán Espinoza, and J. Leonard, “Advances in inference and representation for simultaneous localization and mapping,” Ann. Rev. Control Robot. Auton. Syst., vol. 4, pp. 215–242, 2021.
[2] F. Dellaert and M. Kaess, “Factor graphs for robot perception,” Found. Trends Robot., vol. 6, no. 1/2, pp. i–139, 2017.
[3] A. Majumdar, G. Hall, and A. Ahmadi, “Recent scalability improvements for semidefinite programming with applications in machine learning, control and robotics,” Ann. Rev. Control Robot. Auton. Syst., vol. 3, pp. 331–360, 2020.
[4] S. Burer and R. Monteiro, “A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization,” Math. Prog., vol. 95, pp. 329–357, 2003.
[5] A. Knyazev, “Toward the optimal preconditioned eigensolver: Locally optimal block preconditioned conjugate gradient method,” SIAM J. Sci. Comput., vol. 25, no. 2, pp. 517–541, 2001.
[6] S. Boyd and L. Vandenberghe, Convex Optimization. Cambridge, U.K.: Cambridge Univ. Press, 2004.
[7] J. Nocedal and S. Wright, Numerical Optimization, 2nd ed. New York, NY, USA: Springer, 2006.
[8] N. Boumal, V. Voroninski, and A. Bandeira, “The non-convex Burer-Monteiro approach works on smooth semidefinite programs,” in Proc. Adv. Neural Inf. Process. Syst., 2016, pp. 2757–2765.
[9] D. Cifuentes and A. Moitra, “Polynomial time guarantees for the Burer-Monteiro method,” 2019, arXiv:1912.01745.
[10] D. Rosen, “Scalable low-rank semidefinite programming for certifiably correct machine perception,” in Proc. Intl. Workshop Algorithmic Found. Robot., 2020, pp. 551–566.
[11] N. Boumal, “A riemannian low-rank method for optimization over semidefinite matrices with block-diagonal constraints,” 2015, arXiv:1506.00575v2.
[12] M. Journé, F. Bach, P.-A. Absil, and R. Sepulchre, “Low-rank optimization on the cone of positive semidefinite matrices,” SIAM J. Optim., vol. 20, no. 5, pp. 2327–2351, 2010.
[13] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst, Eds., Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide. Philadelphia, PA, USA: Soc. Ind. Appl. Math. (SIAM), 2000.
[14] Y. Saad, Numerical Methods for Large Eigenvalue Problems, 2nd ed. Philadelphia, PA, USA: Soc. Ind. Appl. Math. (SIAM), 2011.
[15] Y. Tian, K. Khosoussi, D. Rosen, and J. How, “Distributed certifiably correct pose-graph optimization,” IEEE Trans. Robot., vol. 37, no. 6, pp. 2137–2156, Dec. 2021.
[16] D. Rosen, L. Carlone, A. Bandeira, and J. Leonard, “SE-Sync: A certifiably correct algorithm for synchronization over the special euclidean group,” Intl. J. Robot. Res., vol. 38, no. 2/3, pp. 95–125, Mar. 2019.
[17] J. Brialles and J. Gonzalez-Jimenez, “Cartan-sync: Fast and global SE(d)-synchronization,” Robot. Auton. Lett., vol. 2, no. 4, pp. 2127–2134, Oct. 2017.
[18] T. Fan, H. Wang, M. Rubenstei[in, and T. Murphey, “Efficient and guaranteed planar pose graph optimization using the complex number representation,” in Proc. IEEE/RSJ Intl. Conf. Intell. Robots Syst., Macau, China, 2019, pp. 1904–1911.
[19] F. Dellaert, D. Rosen, J. Wu, R. Mahony, and L. Carlone, “Shonan rotation averaging: Global optimality by surfing SO(p)×2,” in Proc. Eur. Conf. Comput. Vis., 2020, pp. 292–308.
[20] L. Carlone, D. Rosen, G. Calafiore, J. Leonard, and F. Dellaert, “La-grangian duality in 3D SLAM: Verification techniques and optimal solutions,” in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., Hamburg, Germany, 2015, pp. 125–132.
[21] L. Carlone and F. Dellaert, “Duality-based verification techniques for 2D SLAM,” in Proc. IEEE Intl. Conf. Robot. Automat., Seattle, WA, USA, 2015, pp. 4589–4596.
[22] J. Brialles and J. Gonzalez-Jimenez, “Fast global optimality verification in 3D SLAM,” in Proc. IEEE/RSJ Intl. Conf. Intell. Robots Syst., Daejon, Korea, 2016, pp. 4630–4636.
[23] D. Rosen and L. Carlone, “Computational enhancements for certifiably correct SLAM,” 2017, presented at the International Conference on Intelligent Robots and Systems (IROS) in the workshop “Introspective Methods for Reliable Autonomy”.
[24] A. Knyazev, M. Argentati, I. Lashuk, and E. Ovtchinnikov, “Block locally optimal preconditioned eigenvectors (BLOPEX) in hypre and PETSc,” SIAM J. Sci. Comput., vol. 29, no. 5, pp. 2224–2239, 2007.
[25] J. Duersch, M. Shao, C. Yang, and M. Gu, “A robust and efficient implementation of LOBPCG,” SIAM J. Sci. Comput., vol. 40, no. 5, pp. C655–C676, Oct. 2018.
[26] G. Golub and C. V. Loan, Matrix Computations, 3rd ed. Baltimore, MD, USA: Johns Hopkins Univ. Press, 1996.
[27] C. Greif, M. Penrose, “SYM-ILDL: Incomplete LU factorization of symmetric indefinite and skew-symmetric matrices,” ACM Trans. Math. Softw., vol. 44, no. 1, pp. 1–21, Apr. 2017.
[28] Y. Chen, T. Davis, W. Hager, and S. Rajamanickam, “Algorithm 887: CHOLMOD, supernodal sparse Cholesky factorization and update/downdate,” ACM Trans. Math. Softw., vol. 35, no. 3, pp. 1–14, Oct. 2008.
[29] M. Penrose, Random Geometric Graphs, Ser. Oxford Studies in Probability, Oxford, U.K.: Oxford Univ. Press, 2003.
[30] F. Chung, Spectral Graph Theory, Providence, Rhode Island: Amer. Math. Soc, 1997.