SIAM J. OPTIM.
Vol. 20, No. 5, pp. 2327–2351
© 2010 Society for Industrial and Applied Mathematics

LOW-RANK OPTIMIZATION ON THE CONE OF POSITIVE SEMIDEFINITE MATRICES*

M. JOURNÉE†, F. BACH‡, P.-A. ABSIL§, AND R. SEPULCHRE†

Abstract. We propose an algorithm for solving optimization problems defined on a subset of the cone of symmetric positive semidefinite matrices. This algorithm relies on the factorization $X = YY^T$, where the number of columns of Y fixes an upper bound on the rank of the positive semidefinite matrix X. It is thus very effective for solving problems that have a low-rank solution. The factorization $X = YY^T$ leads to a reformulation of the original problem as an optimization on a particular quotient manifold. The present paper discusses the geometry of that manifold and derives a second-order optimization method with guaranteed quadratic convergence. It furthermore provides some conditions on the rank of the factorization to ensure equivalence with the original problem. In contrast to existing methods, the proposed algorithm converges monotonically to the sought solution. Its numerical efficiency is evaluated on two applications: the maximal cut of a graph and the problem of sparse principal component analysis.

Key words. low-rank constraints, cone of symmetric positive definite matrices, Riemannian quotient manifold, sparse principal component analysis, maximum-cut algorithms, large-scale algorithms

AMS subject classifications. 65K05, 90C30, 90C25, 90C22, 90C27, 62H25, 58C05, 49M15

DOI. 10.1137/080731359

1. Introduction.

Many combinatorial optimization problems can be relaxed into a convex program. These relaxations are mainly introduced as a tool to obtain lower and upper bounds on the problem of interest. The relaxed solutions provide approximate solutions to the original program. Even when the relaxation is convex, computing its solution might be a demanding task in the case of large-scale problems. In fact, some convex relaxations of combinatorial problems consist in expanding the dimension of the search space by optimizing over a symmetric positive semidefinite matrix variable of the size of the original problem. Fortunately, in many cases, the relaxation is tight once its solution is rank one, and it is expected that the convex relaxation, defined in terms of a matrix variable that is likely to be very large, presents a low-rank solution. This property can be exploited to make a direct solution of the convex problem feasible in large-scale problems.

The present paper focuses on the optimization problem,

$$\begin{array}{lll} & \min\limits_{X\in\mathbb{S}^n} & f(X) \\ (1) & \text{s.t.} & \operatorname{Tr}(A_iX) = b_i, \quad A_i \in \mathbb{S}^n, \quad b_i \in \mathbb{R}, \quad i = 1,\dots,m, \\ & & X \succeq 0, \end{array}$$

*Received by the editors July 28, 2008; accepted for publication (in revised form) March 31, 2010; published electronically May 28, 2010. This paper presents research results of the Belgian Network DYSCO (Dynamical Systems, Control, and Optimization), funded by the Interuniversity Attraction Poles Programme, initiated by the Belgian State, Science Policy Office. The scientific responsibility rests with its authors.

http://www.siam.org/journals/siopt/20-5/73135.html

†Department of Electrical Engineering and Computer Science, University of Liège, 4000 Liège, Belgium (M.Journee@ulg.ac.be, R.Sepulchre@ulg.ac.be). Michel Journée is a research fellow of the Belgian National Fund for Scientific Research (FNRS).

‡INRIA—Willow project, Département d'Informatique, Ecole Normale Supérieure, 45, rue d'Ulm, 75230 Paris, France (Francis.Bach@mines.org).

§Department of Mathematical Engineering, Université Catholique de Louvain, B-1348 Louvain-la-Neuve, Belgium (http://www.inma.ucl.ac.be/~absil/).

where f is a smooth function and $\mathbb{S}^n = \{X \in \mathbb{R}^{n\times n} | X^T = X\}$ is the set of symmetric matrices of $\mathbb{R}^{n\times n}$. Although the objective function f will be convex in most of the applications discussed in this paper—in which case (1) is a convex program—this assumption is not required by the proposed optimization algorithm, which computes then a local solution of (1).

In general, the solution of the program (1) has to be searched in a space of dimension $O(n^2)$. Solving such a problem becomes rapidly untractable for large n. In this paper, we propose an approach for solving (1) at a reduced computational cost once the following assumptions hold.

Assumption 1. The program (1) presents a low-rank solution X^*, i.e.,

$$\operatorname{rank}(X^*) = r \ll n.$$

Assumption 2. Either the number m of equality constraints is one, or the symmetric matrices A_i satisfy

$$A_i A_j = 0$$

for any $i, j \in \{1, \dots, m\}$ such that $i \neq j$.

By considering the compact eigenvalue decomposition $A_i = V_i D_i V_i^T$ with the rectangular matrix V_i having orthonormal columns and the full-rank diagonal matrix D_i, Assumption 2 implies the V_i's to be mutually orthogonal, i.e., $V_i^T V_j = 0$ for all $i \neq j$. As a consequence, there can be at most n equality constraints in (1), i.e., $m \leq n$, in which case all matrices A_i are rank one. Assumption 2 is, for instance, fulfilled by the *spectahedron*,

$$\mathcal{S} = \{X \in \mathbb{S}^n | X \succeq 0, \operatorname{Tr}(X) = 1\},$$

and the *elliptope* (also known as the set of correlation matrices),

$$\mathcal{E} = \{X \in \mathbb{S}^n | X \succeq 0, \operatorname{diag}(X) = \mathbf{1}\}. \tag{2}$$

Assumption 1 suggests to factor the matrix variable X as the product

$$X = YY^T, \tag{3}$$

where the number of independent columns of $Y \in \mathbb{R}^{n\times p}$ fixes the rank of X. Solving the nonlinear optimization program

$$\begin{array}{lll} \min\limits_{Y\in\mathbb{R}^{n\times p}} & f(YY^T) & \\ \text{s.t.} & \operatorname{Tr}(Y^T A_i Y) = b_i, \quad A_i \in \mathbb{S}^n, \quad b_i \in \mathbb{R}, \quad i = 1, \dots, m \end{array} \tag{4}$$

in terms of the new variable Y amounts to searching a space of dimension np, which can be much lower than the dimension of the positive semidefinite matrices X. The parameter p should ideally equal the rank r, which is usually unknown. The proposed algorithm for solving (1) combines thus a method that finds a local minimizer Y of (4) with an approach that increments p until a sufficient condition is satisfied for Y to provide a solution YY^T of (1). Note that even when the original program (1) is convex, the low-rank reformulation (4) is not.

A further potential difficulty of (4) is that the solutions are not isolated. For any solution Y and any orthogonal matrix Q of $\mathbb{R}^{p\times p}$, i.e., such that $Q^TQ = I$, the matrix YQ is also a solution. In other words, the program (4) is invariant by right multiplication of the unknown with an orthogonal matrix. This issue is not harmful for simple gradient schemes but it greatly affects the convergence of second-order methods that we would like to use here (see, e.g., [4] and [3]). In order to take into account the inherent symmetry of the solution, the proposed method conceptually considers a search space whose points are the equivalence classes $\{YQ | Q \in \mathbb{R}^{p\times p}, Q^TQ = I\}$. The minimizers of (4) can be isolated in that *quotient* space.

The idea of reformulating a convex program into a nonconvex one by factorization of the matrix unknown is not new and was investigated in [5] for solving semidefinite programs (SDP). While the setup considered in [5] is general but restricted to gradient methods, the present paper further exploits the particular structure of the equality constraints (Assumption 2) and proposes second-order methods (i.e., which exploit second-order derivative information) that lead to a descent algorithm with guaranteed quadratic convergence. More recently, the authors of [9] have proposed manifold-based optimization algorithms to exploit the factorization (3) to efficiently solve optimization problems that are defined on the elliptope (2), very much in the same spirit as in the present paper. However, the quotient structure studied here is fundamentally different from the embedded structure used in [9] and leads to very distinct algorithms and numerical results.

Besides being based on quadratic second-order methods, the proposed algorithm presents two further important features. First, in contrast to the algorithm of [5], it converges *monotonically* toward the solution of (1). Second, it is provided with an indicator of convergence able to control the accuracy of the results. This tool is particularly convenient to find a reasonable trade-off between fidelity in the initial program and computational efficiency.

The paper is organized as follows. In section 3, we derive conditions for an optimizer of (4) to represent a solution of the original problem (1). A meta-algorithm for solving (1) based on the factorization (3) is built upon these theoretical results. In section 4, we describe the geometry of the underlying quotient manifold and propose an algorithm for solving (4) based on second-order derivative information. In sections 5 and 6, we evaluate the efficiency of the proposed algorithm on two applications: the maximal cut of a graph and three different formulations of the problem of sparse principal component analysis, including a nonsmooth and nonconvex program.

2. Notations.

Given a function $f : \mathbb{S}^n \to \mathbb{R} : X \mapsto f(X)$, we define the function

$$\bar{f} : \mathbb{R}^{n\times p} \to \mathbb{R} : Y \mapsto \bar{f}(Y) = f(YY^T).$$

For a differentiable function f, the notation $\nabla_X f(X_0)$ refers to the gradient of f at X_0 with respect to the variable X,

$$[\nabla_X f(X_0)]_{i,j} = \frac{\partial f}{\partial X_{i,j}}(X_0).$$

The derivative of f at X_0 in a direction Z is written

$$\mathrm{D}_X f(X_0)[Z] = \lim_{t\to 0} \frac{f(X_0 + tZ) - f(X_0)}{t}.$$

It holds that

$$\mathrm{D}_X f(X_0)[Z] = \langle \nabla_X f(X_0), Z\rangle,$$

where $\langle \cdot, \cdot \rangle$ denotes the Frobenius inner product $\langle Z_1, Z_2\rangle = \mathrm{Tr}(Z_1^T Z_2)$.

3. Optimality conditions.

In this section, we derive and analyze the optimality conditions of both problems (1) and (4). These conditions provide theoretical insight about the rank p at which (4) should be solved as well as conditions for an optimizer of (4) to represent a solution of the original problem (1). A meta-algorithm for solving (1) is then derived from these results.

3.1. First-order optimality conditions.

DEFINITION 1. *A stationary point of* (1) *is a symmetric matrix $X \in \mathbb{S}^n$ for which there exists a vector $\sigma \in \mathbb{R}^m$ and a symmetric matrix $S \in \mathbb{S}^n$ such that the first-order optimality conditions hold:*

$$\begin{aligned} \operatorname{Tr}(A_i X) &= b_i, \\ X &\succeq 0, \\ S &\succeq 0, \\ SX &= 0, \\ S &= \nabla_X f(X) - \sum_{i=1}^m \sigma_i A_i. \end{aligned} \tag{5}$$

The optimality conditions (5) are necessary and sufficient for convex optimization problems [6]. In the case of a nonconvex objective function f, we consider any stationary point of (1) as a solution of the problem. The local minimizers will, in fact, be the only stable accumulation points of the optimization method proposed in the sequel, which is a descent algorithm for f.

LEMMA 2. *If Y is a local optimum of* (4), *then there exists a vector $\lambda \in \mathbb{R}^m$ such that*

$$\begin{aligned} \operatorname{Tr}(Y^T A_i Y) &= b_i, \\ \left(\nabla_X f(YY^T) - \sum_{i=1}^m \lambda_i A_i\right) Y &= 0. \end{aligned} \tag{6}$$

If the $\{A_i Y\}_{i=1,\dots,m}$ are linearly independent, the vector λ is unique.

Proof. These are the first-order KKT conditions of (4) (see, e.g., [14]). ▯

LEMMA 3. *Under Assumption* 2, *the matrices $\{A_i Y\}_{i=1,\dots,m}$ are linearly independent at any $Y \in \mathbb{R}^{n\times p}$ provided that all $b_i \neq 0$.*

Proof. The matrices $\{A_i Y\}_{i=1,\dots,m}$ are linearly independent if and only if the equality

$$\sum_{i=1}^m \gamma_i A_i Y = 0$$

implies that $\gamma_i = 0$ for all $i = 1, \dots, m$. By virtue of Assumption 2,

$$\operatorname{Tr}\left(Y^T A_j \sum_{i=1}^m \gamma_i A_i Y\right) = \gamma_j \operatorname{Tr}(Y^T A_j^2 Y) = \gamma_j \|A_j Y\|_F^2 = 0$$

for $j = 1, \dots, m$. If $b_j \neq 0$ in the equality constraints of (4), $A_j Y$ cannot be a zero matrix; i.e., $\|A_j Y\|_F \neq 0$ and thus $\gamma_j = 0$. ▯

Given a local minimizer Y of (4), one readily notices that all but one condition of Definition 1 hold for the symmetric positive semidefinite matrix YY^T. Comparison of Definition 1 and Lemma 2 therefore provides the following relationship between problems (4) and (1).

Theorem 4. *A local minimizer Y of the nonconvex problem* (4) *provides the stationary point YY^T of problem* (1) *if the matrix*

$$S_Y = \nabla_X f(YY^T) - \sum_{i=1}^{m} \lambda_i A_i \tag{7}$$

is positive semidefinite for the Lagrangian multipliers λ_i that satisfy (6). *This sufficient condition is furthermore necessary if $b_i \neq 0$ for all $i = 1, \ldots, m$.*

Proof. For the sufficient part, check the conditions of Definition 1 for the tuple $\{X, S, \sigma\} = \{YY^T, S_Y, \lambda\}$. For the necessary part, assume that $X = YY^T$ satisfies the conditions (5) for some σ and S. By virtue of Lemmas 2 and 3, the vector λ is unique and thus necessarily equal to σ; i.e., $S_Y = S \succeq 0$. ◻

Theorem 4 is a generalization of Proposition 3 in [5] to nonlinear objective functions. Under Assumption 2, the Lagrangian multipliers in (6) have the closed-form expression

$$\lambda_i = \frac{\mathrm{Tr}(Y^T A_i \nabla_X f(YY^T) Y)}{\mathrm{Tr}(Y^T A_i^2 Y)} \tag{8}$$

that is readily obtained from the identity

$$\mathrm{Tr}\left(Y^T A_i \left(\nabla_X f(YY^T) - \sum_{j=1}^{m} \lambda_j A_j \right) Y \right) = 0,$$

where the second condition in (6) is used. Hence, a closed-form expression is available for the dual matrix S_Y in (7) at an optimizer Y of (4).

3.2. Second-order optimality conditions.

Let $\mathcal{L}(Y, \lambda)$ denote the Lagrangian of the nonconvex problem (4),

$$\mathcal{L}(Y, \lambda) = f(YY^T) - \sum_{i=1}^{m} \lambda_i (\mathrm{trace}(Y^T A_i Y) - b_i).$$

The optimality conditions (6) can be rewritten in the form

$$\nabla_\lambda \mathcal{L}(Y, \lambda) = 0 \quad \text{and} \quad \nabla_Y \mathcal{L}(Y, \lambda) = 0.$$

In the following, we consider the Lagrangian multipliers λ_i to be given by (8).

Lemma 5. *For a local minimizer $Y \in \mathbb{R}^{n\times p}$ of* (4), *it holds that*

$$\mathrm{Tr}(Z^T \mathrm{D}_Y \nabla_Y \mathcal{L}(Y, \lambda)[Z]) \geq 0 \tag{9}$$

for any matrix $Z \in \mathbb{R}^{n\times p}$ that satisfies

$$\mathrm{Tr}(Z^T A_i Y) = 0,\ i = 1, \ldots, m. \tag{10}$$

Proof. These are the second-order KKT conditions of (4) (see, e.g., [14]). ◻

Lemma 6. *For any matrix $Z \in \mathbb{R}^{n\times p}$ such that $YZ^T = 0$, the following equality holds:*

$$\frac{1}{2}\mathrm{Tr}(Z^T \mathrm{D}_Y \nabla_Y \mathcal{L}(Y, \lambda)[Z]) = \mathrm{Tr}(Z^T S_Y Z).$$

Proof. By noting that $\nabla_Y \mathcal{L}(Y,\lambda) = 2S_Y Y$, one has

$$\begin{aligned}&\frac{1}{2}\mathrm{Tr}(Z^T \mathrm{D}_Y \nabla_Y \mathcal{L}(Y,\lambda)[Z])\\ &\qquad = \mathrm{Tr}(Z^T S_Y Z) + \mathrm{Tr}(Z^T \mathrm{D}_Y(\nabla f(YY^T))[Z]Y) - \sum_{i=1}^m \mathrm{D}_Y \lambda_i[Z]\mathrm{Tr}(Z^T A_i Y),\end{aligned}$$

where the two last terms cancel out by virtue of the condition $YZ^T = 0$. ◻

THEOREM 7. *A local minimizer Y of problem* (4) *provides a stationary point $X = YY^T$ of problem* (1) *if it is rank deficient.*

Proof. For the matrix $Y \in \mathbb{R}^{n\times p}$ to span an r-dimensional subspace in $\mathbb{R}^n$ (with $p > r$), the following factorization has to hold:

$$Y = \tilde{Y} M^T, \tag{11}$$

with the full-rank matrices $\tilde{Y} \in \mathbb{R}_*^{n\times r}$ and $M \in \mathbb{R}_*^{p\times r}$. Let $M_\perp \in \mathbb{R}^{p\times(p-r)}$ be an orthogonal basis for the orthogonal complement of the column space of M; i.e., $M^T M_\perp = 0$ and $M_\perp^T M_\perp = I$. For any matrix $\tilde{Z} \in \mathbb{R}^{n\times(p-r)}$, the matrix $Z = \tilde{Z} M_\perp^T$ satisfies

$$YZ^T = 0$$

such that the conditions (10) hold. By virtue of Lemmas 5 and 6,

$$\mathrm{Tr}(Z^T S_Y Z) \geq 0$$

for all the matrices $Z = \tilde{Z} M_\perp^T$; i.e., the matrix S_Y is positive semidefinite, and $X = YY^T$ is a stationary point of problem (1). ◻

Theorem 7 is a generalization of Proposition 4 in [5] to nonlinear objective functions.

COROLLARY 8. *In the case $p = n$, any local minimizer $Y \in \mathbb{R}^{n\times n}$ of problem* (4) *provides the stationary point $X = YY^T$ of problem* (1).

Proof. If Y is rank deficient, the matrix $X = YY^T$ is optimal for (1) by virtue of Theorem 7. Otherwise, the matrix S_Y is zero because of the second condition in (6) and X is optimal for (1). ◻

3.3. An algorithm for the initial problem. The proposed algorithm consists of solving a sequence of nonconvex problems (4) of increasing dimension until the resulting local minimizer Y represents a stationary point of problem (1), i.e., a solution of (1) since it is a descent algorithm. Both Theorems 4 and 7 provide sufficient conditions to check this fact. It should be noted that the conditions of Theorem 4 are furthermore necessary if all $b_i \neq 0$. When problem (4) is solved in a dimension p smaller than the unknown rank r, none of these conditions can be fulfilled. The dimension p is thus incremented after each resolution of (4). By virtue of Corollary 8, a stationary point of the initial problem (1) is obtained at the latest once $p = n$. However, for the sake of numerical efficiency, the hope is to find such a point for a dimension p that is much smaller than the problem dimension n. As we will see in sections 5 and 6, this hope is perfectly fulfilled in each of our numerical experiments.

In order to ensure a monotone decrease of the objective function through the iterations, the optimization algorithm that solves (4) is initialized with a matrix corresponding to Y with an additional zero column appended; i.e.,

$$Y_0 = [Y|0^{n\times 1}],$$

where $0^{n\times 1}$ denotes the n-by-1 zero vector. It should be noted that if $\nabla_Y \mathcal{L}(Y,\lambda) = 0$, then $\nabla_{Y_0}\mathcal{L}(Y_0,\lambda) = 0$ for the Lagrangian multipliers λ_i given by (8); i.e., Y_0 is a stationary point of problem (4) for the dimension $p+1$. Since this reinitialization occurs when the local minimizer $Y \in \mathbb{R}^{n\times p}$ of (4) does not represent the solution of (1) according to the necessary (if all $b_i \neq 0$) and sufficient conditions of Theorem 4, Y_0 is a saddle point of problem (4). This can be a critical issue for many optimization algorithms. Fortunately, in the present case, a descent direction from Y_0 can be explicitly evaluated. From Lemma 6, the matrix $Z = [0^{n\times p}|v]$, where $0^{n\times p}$ is a zero matrix of the size of Y and v is the eigenvector of S_Y related to the smallest algebraic eigenvalue, verifies

$$\frac{1}{2}\mathrm{Tr}(Z^T \mathrm{D}_Y \nabla_Y \mathcal{L}(Y_0,\lambda)[Z]) = v^T S_Y v \leq 0$$

since $Y_0 Z^T = 0$ for the Lagrangian multipliers λ given in (8). All these elements lead to the meta-algorithm displayed in Algorithm 1.[1] The parameter ε fixes a threshold on the eigenvalues of S_Y to decide about the nonnegativity of this matrix. ε is chosen to be 10^{-6} in our implementation.

Algorithm 1. Meta-algorithm for solving problem (1)

```
input  : Initial rank $p_0$, initial iterate $Y^{(0)} \in \mathbb{R}^{n\times p_0}$ and parameter $\varepsilon$.
output: The solution $X$ of problem (1).
begin
    $p \longleftarrow p_0$
    $Y_p \longleftarrow Y^{(0)}$
    stop $\longleftarrow$ 0
    while stop $\neq 1$ do
        Initialize an optimization scheme with $Y_p$ to find a local minimum $Y_p^*$
        of (4) by exploiting a descent direction $Z_p$ if available.
        if $p = p_0$ and rank$(Y_p^*) < p$ then
            stop = 1
        else
            Find the smallest eigenvalue $\lambda_{\min}$ and the related eigenvector $V_{\min}$
            of the matrix $S_Y$ (7).
            if $\lambda_{\min} \geq -\varepsilon$ then
                stop = 1
            else
                $p \longleftarrow p+1$
                $Y_p \longleftarrow [Y_p^*|0]$
                A descent direction from the saddle point $Y_p$ is given by
                $Z_p = [0|V_{\min}]$.
    $X \longleftarrow Y_p^* Y_p^{*T}$
end
```

It should be mentioned that, to check the optimality for the initial problem (1) of a local minimizer Y_p^*, the rank condition of Theorem 7 is computationally cheaper to evaluate than the nonnegativity condition of Theorem 4. Nevertheless, the rank

[1]A MATLAB implementation of Algorithm 1 with the manifold-based optimization method of section 4 can be downloaded from http://www.montefiore.ulg.ac.be/~journee.

condition does not provide a descent direction to escape saddle points. It furthermore requires to solve the program (4) at a dimension that is strictly greater than r, the rank of the solution of (1). Hence, this condition is used only at the initial rank p_0 and is likely to hold if p_0 is chosen larger than the unknown r. Numerically, the rank of $Y^*_{p_0}$ is computed as the number of singular values that are greater than a threshold fixed at 10^{-6}. The algorithm proposed in [5] exploits exclusively the rank condition of Theorem 7. For this reason, each optimization of (4) has to be randomly initialized, and the algorithm in [5] is not a descent algorithm.

As mentioned above, Algorithm 1 stops at the latest once $p = n$. The numerical experiments reported in sections 5 and 6 indicate that in practice, however, the algorithm terminates at a rank p that is much lower than the dimension n. It furthermore seems that the algorithm always terminates once p equals the rank r of the solution of (1), provided that $p_0 \leq r$. These applications also illustrate that the magnitude of the smallest eigenvalue $\lambda_{\min}$ of the matrix S_Y can be used to monitor the convergence. The value $|\lambda_{\min}|$ indicates whether the current iterate is close to satisfying the KKT conditions (5). This feature is of great interest once an approximate solution to (1) is sufficient. The threshold ε set on $\lambda_{\min}$ controls then the accuracy of the result.

A trust-region scheme based on second-order derivative information is proposed in the next section for computing a local minimum of (4). This method is provided with a convergence theory that ensures convergence of the iterates toward a local minimizer.

Hence, the proposed algorithm presents the following notable features. First, it converges toward the solution of problem (1) by ensuring a monotone decrease of the objective function. Then the magnitude of the smallest eigenvalue of S_Y provides a means to monitor the convergence. Finally the inner problem (4) is solved by second-order methods featuring quadratic local convergence.

4. Manifold-based optimization.

We now derive an optimization scheme that locally solves the nonconvex and nonlinear program,

$$\begin{array}{lll} & \min\limits_{Y\in\mathbb{R}^{n\times p}} & \bar{f}(Y) \\ (12) & \text{s.t.} & \mathrm{Tr}(Y^TA_iY) = b_i, \quad A_i \in \mathbb{S}^n, \quad b_i \in \mathbb{R}, \quad i = 1,\dots,m, \end{array}$$

where $\bar{f}(Y) = f(YY^T)$ for some $f : \mathbb{S}^n \to \mathbb{R}$.

As previously mentioned, problem (12) is invariant by right-multiplication of the variable Y by orthogonal matrices. The critical points of (12) are thus nonisolated. To get rid of this symmetry, let $\mathcal{M}$ define the set of all the equivalence classes of the form

$$[Y] = \{YQ \mid Q \in \mathbb{R}^{p\times p},\ Q^TQ = I_p\}, \tag{13}$$

where $Y \in \mathbb{R}^{n\times p}_*$ satisfies the quadratic equality constraints in (12); i.e., Y belongs to the manifold

$$\bar{\mathcal{M}} = \{Y \in \mathbb{R}^{n\times p}_* \mid \mathrm{trace}(Y^TA_iY) = b_i,\ i = 1,\dots,m\},$$

which is embedded in the noncompact Stiefel manifold $\mathbb{R}^{n\times p}_*$, i.e., the set of *full-rank* matrices in $\mathbb{R}^{n\times p}$. The full-rank condition is required to deal with differentiable manifolds. The set $\mathcal{M}$ is the *quotient* of the manifold $\bar{\mathcal{M}}$ by the *orthogonal group* $\mathcal{O}(p) = \{Q \in \mathbb{R}^{p\times p} | Q^TQ = I_p\}$,

$$\mathcal{M} = \bar{\mathcal{M}}/\mathcal{O}(p).$$

It can be furthermore proven that the quotient $\mathcal{M}$ presents the structure of a Riemannian manifold; i.e., it is a quotient manifold [4]. Let us turn problem (12) onto the quotient manifold $\mathcal{M}$, i.e.,

$$\min_{[Y]\in\mathcal{M}} \phi([Y]), \tag{14}$$

with the function $\phi : \mathcal{M} \to \mathbb{R} : [Y] \mapsto \phi([Y]) = \bar{f}(Y)$. If the minimizers of f are isolated on the feasible set of (1), then the minimizers of ϕ are isolated on the search space $\mathcal{M}$.

Isolated minimizers can also be obtained by adding a suitable set of constraints to problem (12) and optimizing on a submanifold of $\bar{\mathcal{M}}$. This is the alternative considered in [9] in the case of the elliptope: the rotational invariance is removed by imposing the matrix variable Y to be lower triangular, which defines the *Cholesky manifold* embedded in $\mathbb{R}^{n\times p}$. Choosing the appropriate set of constraints to add to (12) seems, however, somewhat arbitrary. We therefore prefer to keep the intrinsic symmetry of the problem and to optimize on the quotient manifold $\mathcal{M}$, although this might be conceptually more complex. As shown in the forthcoming numerical experiments (section 5), the quotient parametrization leads to better results.

Several unconstrained optimization methods have been generalized to search spaces that are Riemannian manifolds. This is, e.g., the case of the trust-region approach [1], which minimizes at each iteration a quadratic model of the objective on a trust-region whose radius is adaptively chosen. Since this model is meant only locally, it is defined for the elements of the tangent space to the manifold $\mathcal{M}$ at the current iterate. This tangent space is a Euclidean space, and thus the trust-region subproblem can be solved by any available method. In the numerical experiments on which this paper reports, we used the Steihaug–Toint truncated conjugate gradient method, as formulated in [1, Alg. 2]. Once the direction solving the subproblem has been identified, the trust-region radius is adapted in a similar manner as in the Euclidean case, and the update consists of moving the iterate along a curve that is tangent to this direction; see [1, Alg. 1] for a detailed algorithm description. Hence, the main difference between the classical and the Riemannian trust-region methods is that in the Riemannian version, a different tangent space is considered at each iteration. Details on this algorithm can be found in [1, 4]. It is important to mention that this algorithm has convergence properties analogous to those of trust-region methods for unconstrained optimization in $\mathbb{R}^n$. In particular, trust-region methods on manifolds converge globally to stationary points of the objective function if the inner iteration produces a model decrease that is better than a fixed fraction of the Cauchy decrease; such a property is achieved, e.g., by the Steihaug–Toint inner iteration. Since the iteration is moreover a descent method, convergence to saddle points or local maximizers is not observed in practice. It is possible to obtain guaranteed convergence to a point where the second-order necessary conditions of optimality hold by using inner iterations that exploit the model more fully (e.g., the inner iteration of Moré and Sorensen), but these inner iterations tend to be prohibitively expensive for large-scale problems. For appropriate choices of the inner iteration stopping criterion (see eq. (10) in [1]), trust-region methods converge locally superlinearly toward the nondegenerate local minimizers of the objective function. In our numerical experiments, the parameter θ in eq. (10) of [1] has been set to 1, which guarantees a quadratic convergence, and κ has been set to 10^{-1}.

To exploit the Riemannian trust-region algorithm of [1] in the context of problem (14), a few important objects need to be specified. First, every equivalence class $[Y]$

is represented by one of its elements $Y \in \bar{\mathcal{M}}$; i.e., the algorithm works conceptually on the entire quotient space $\mathcal{M}$ but numerically in $\mathbb{R}^{n\times p}$. Then the tangent space at a point Y of the manifold $\bar{\mathcal{M}}$,

$$T_Y\bar{\mathcal{M}} = \{Z \in \mathbb{R}^{n\times p} : \mathrm{Tr}(Y^T A_i Z) = 0,\ i = 1, \dots, m\},$$

has to be decomposed in two orthogonal subspaces, the *vertical space* $\mathcal{V}_Y\mathcal{M}$ and the *horizontal space* $\mathcal{H}_Y\mathcal{M}$. The vertical space $\mathcal{V}_Y\mathcal{M}$ is the tangent space to the equivalence classes

$$\mathcal{V}_Y\mathcal{M} = \{Y\Omega : \Omega \in \mathbb{R}^{p\times p},\ \Omega^T = -\Omega\}.$$

The horizontal space $\mathcal{H}_Y\mathcal{M}$ is the orthogonal complement of $\mathcal{V}_Y\mathcal{M}$ in $T_Y\bar{\mathcal{M}}$; i.e.,

$$\mathcal{H}_Y\mathcal{M} = \{Z \in T_Y\bar{\mathcal{M}} : Z^T Y = Y^T Z\} \tag{15}$$

for the Frobenius inner product $\langle Z^{(1)}, Z^{(2)}\rangle = \mathrm{Tr}(Z^{(1)T}Z^{(2)})$ for all $Z^{(1)}, Z^{(2)} \in T_Y\bar{\mathcal{M}}$. Expression (15) results from the equality $\mathrm{Tr}(S\Omega) = 0$ that holds for any symmetric matrix S and skew-symmetric matrix Ω of compatible dimension. The purpose of the horizontal space is to provide a way of representing tangent spaces to the quotient manifold $\mathcal{M}$: given $Y \in \bar{\mathcal{M}}$ and a tangent vector $Z_{[Y]}$ to $\mathcal{M}$ at $[Y]$, there exists a unique $Z_Y \in \mathcal{H}_Y\mathcal{M}$, termed the *horizontal lift* of $Z_{[Y]}$ at Y, such that, for all smooth functions h on $\mathcal{M}$, it holds that

$$\mathrm{D}h(Y)[Z_{[Y]}] = \mathrm{D}(h \circ \pi)(Y)[Z_Y],$$

where π is the quotient map $\pi : Y \mapsto [Y]$. A vector field $Y \in \bar{\mathcal{M}} \mapsto Z_Y \in \mathcal{H}_Y\mathcal{M}$ is a horizontal lift of a tangent vector to $\mathcal{M}$ if and only if it satisfies $Z_{YQ} = Z_Y Q$ for all $Q \in \mathcal{O}(p)$.

Let $N_Y\bar{\mathcal{M}}$ be the *normal space* to $\bar{\mathcal{M}}$ at Y, i.e., the orthogonal complement of $T_Y\bar{\mathcal{M}}$ in $\mathbb{R}^{n\times p}$ with respect to the Frobenius inner product,

$$N_Y\bar{\mathcal{M}} = \left\{ \sum_{i=1}^{m} \alpha_i A_i Y,\ \alpha \in \mathbb{R}^m \right\}.$$

The Euclidean space $\mathbb{R}^{n\times p}$ is then uniquely divided into three mutually orthogonal subspaces:

$$\mathbb{R}^{n\times p} = \mathcal{H}_Y\mathcal{M} \oplus \mathcal{V}_Y\mathcal{M} \oplus N_Y\bar{\mathcal{M}}.$$

The trust-region algorithm proposed in [1] requires a projection P_Y from $\mathbb{R}^{n\times p}$ to $\mathcal{H}_Y\mathcal{M}$ along $\mathcal{V}_Y\mathcal{M} \oplus N_Y\bar{\mathcal{M}}$. The following theorem provides a closed-form expression.

THEOREM 9. *Let Y be a point on $\bar{\mathcal{M}}$. For a matrix $Z \in \mathbb{R}^{n\times p}$, the projection $P_Y : \mathbb{R}^{n\times p} \to \mathcal{H}_Y\mathcal{M}$ is given by*

$$P_Y(Z) = Z - Y\Omega - \sum_{i=1}^{m} \alpha_i A_i Y,$$

where Ω is the skew-symmetric matrix that solves the Sylvester equation,

$$\Omega Y^T Y + Y^T Y \Omega = Y^T Z - Z^T Y,$$

and with the coefficients

$$\alpha_i = \frac{\mathrm{Tr}(Z^T A_i Y)}{\mathrm{Tr}(Y^T A_i^2 Y)}.$$

Proof. Any vector $Z \in \mathbb{R}^{n\times p}$ presents a unique decomposition

$$Z = Z_{\mathcal{V}_Y\mathcal{M}} + Z_{\mathcal{H}_Y\mathcal{M}} + Z_{N_Y\bar{\mathcal{M}}},$$

where each element $Z_{\mathcal{X}}$ belongs to the vector space $\mathcal{X}$. The orthogonal projection $\mathcal{P}_Y(\cdot)$ extracts the component that lies in the horizontal space,

$$P_Y(Z) = Z - Y\Omega - \sum_{i=1}^{m} \alpha_i A_i Y,$$

with Ω a skew-symmetric matrix. The parameters Ω and α are determined from the linear equations

$$\begin{aligned} &Y^T P_Y(Z) = P_Y(Z)^T Y,\\ &\mathrm{Tr}(Y^T A_i P_Y(Z)) = 0, \quad i = 1,\ldots m, \end{aligned}$$

which are satisfied by any element of the horizontal space. □

The projection P_Y provides simple formulas to compute derivatives of the function ϕ (defined on the quotient manifold) from derivatives of the function $\bar{f}$ (defined in the Euclidean space). First we consider the gradient of ϕ. For the gradient to be well defined, we need a Riemannian metric on $\mathcal{M}$, which we define by

$$\langle Z^{(1)}_{[Y]}, Z^{(2)}_{[Y]}\rangle = \mathrm{Tr}(Z_Y^{(1)T} Z_Y^{(2)}).$$

Then the horizontal lift of the gradient of ϕ is the projection on the horizontal space of the gradient of $\bar{f}$,

$$\mathrm{grad}\phi(Y) = P_Y(\nabla \bar{f}(Y)).$$

Similarly, the Riemannian Hessian of ϕ in a direction $Z \in \mathcal{H}_Y\mathcal{M}$ is represented by

$$\mathrm{Hess}\phi(Y)[Z] = P_Y(\mathrm{D}(\mathrm{grad}\phi(Y))[Z]).$$

This follows from the theory of Riemannian submersions (see sections 3.6.2, 5.3.4, and 5.5 in [4]). The directional derivative $\mathrm{D}(\zeta)[Z]$, where $\zeta = P_Y\bar{\zeta}$ and $\bar{\zeta}$ is a vector field on $\mathbb{R}_*^{n\times p}$, is performed in the Euclidean sense in $\mathbb{R}^{n\times p}$, i.e.,

$$\mathrm{D}(P_Y(\bar{\zeta}))[Z] = \mathrm{D}\bar{\zeta}[Z] - Z\Omega - Y\mathrm{D}\Omega[Z] - \sum_{i=1}^{m}\alpha_i A_i Z - \sum_{i=1}^{m}\mathrm{D}\alpha_i[Z]A_iY,$$

where $\mathrm{D}\Omega[Z]$ is the solution of the Sylvester equation,

$$\begin{aligned} &\mathrm{D}\Omega[Z]Y^TY + Y^TY\mathrm{D}\Omega[Z]\\ &\quad = Z^T\bar{\zeta} - \bar{\zeta}^T Z + Y^T\mathrm{D}\bar{\zeta}[Z] - \mathrm{D}\bar{\zeta}[Z]^T Y - \Omega(Z^TY + Y^TZ) - (Z^TY + Y^TZ)\Omega, \end{aligned}$$

and

$$\mathrm{D}\alpha_i[Z] = \frac{1}{\mathrm{trace}(Y^TA_i^2Y)}(\mathrm{D}\bar{\zeta}[Z]A_iY + \bar{\zeta}^TA_iZ) - \frac{\mathrm{trace}(Z^TA_iY)}{\mathrm{trace}(Y^TA_i^2Y)^2}(Z^TA_i^2Y + Y^TA_i^2Z).$$

The trust-region subproblem is then written in the form

$$\begin{aligned}&\min_{Z\in\mathcal{H}_Y\mathcal{M}} \phi([Y]) + \operatorname{Tr}(Z^T\operatorname{grad}\phi(Y)) + \frac{1}{2}\operatorname{Tr}(Z^T\operatorname{hess}\phi(Y)[Z])\\ &\text{s.t. } \operatorname{Tr}(Z^TZ) \le \Delta^2,\end{aligned}$$

which minimizes a quadratic model of the objective ϕ on a trust-region of radius Δ.

Finally a last ingredient needed by the Riemannian trust-region algorithm in [1] is a *retraction* $\mathcal{R}_{[Y]} : T_{[Y]}\mathcal{M} \to \mathcal{M}$, represented by a mapping

$$\mathcal{R}_Y : \mathcal{H}_Y\mathcal{M} \to \bar{\mathcal{M}},$$

satisfying the compatibility condition $[\mathcal{R}_{YQ}(ZQ)] = [\mathcal{R}_Y(Z)]$ for all $Y \in \bar{\mathcal{M}}$, all $Z \in \mathcal{H}_Y\mathcal{M}$, and all $Q \in \mathcal{O}(p)$. Such a mapping can be derived from *geodesics*, which are the curves of the shortest path on a manifold. The following theorem provides some insight on the geodesic curves on $\mathcal{M}$. Note that the additional assumptions needed in this theorem are satisfied by the spectahedron and the elliptope.

THEOREM 10. *Under the assumption that $b_i \neq 0$ for $i = 1,\ldots,m$, let $\bar{A}_i = V_iD_iV_i^T$ be the compact eigenvalue decomposition of the matrix $\bar{A}_i = \frac{1}{b_i}A_i$. If $\sum_{i=1}^m V_iV_i^T$ is the identity matrix (i.e., the matrix $[V_1|\ldots|V_m]$ is orthogonal), then the curve*

$$Y(t) = \sum_{i=1}^m V_iV_i^T \left(\cos\left(\sqrt{\operatorname{Tr}(\dot{Y}_0^T\bar{A}_i\dot{Y}_0)}t\right)Y_0 + \frac{\sin\left(\sqrt{\operatorname{Tr}(\dot{Y}_0^T\bar{A}_i\dot{Y}_0)}t\right)}{\sqrt{\operatorname{Tr}(\dot{Y}_0^T\bar{A}_i\dot{Y}_0)}}\dot{Y}_0\right), \tag{16}$$

which passes through $Y_0 \in \bar{\mathcal{M}}$ and is tangent to $\dot{Y}_0 \in \mathcal{H}_{Y_0}\mathcal{M}$ at $t = 0$, is a curve on $\bar{\mathcal{M}}$. In the specific case where $A_i = \gamma_iV_iV_i^T$ with $\gamma_i \in \mathbb{R}$ for all $i = 1,\ldots,m$, the curve $[Y(t)]$ is a geodesic on $\mathcal{M}$.

Proof. First one readily checks that $\operatorname{Tr}(\dot{Y}(t)^TA_iY(t)) = 0$, and hence

$$\begin{aligned}\operatorname{Tr}(Y(t)^TA_iY(t)) &= \operatorname{Tr}(Y_0^TA_iY_0)\cos\left(\sqrt{\operatorname{Tr}(\dot{Y}_0^T\bar{A}_i\dot{Y}_0)}t\right)^2 + b_i\sin\left(\sqrt{\operatorname{Tr}(\dot{Y}_0^T\bar{A}_i\dot{Y}_0)}t\right)^2\\ &= b_i,\end{aligned}$$

which proves that $Y(t)$ is a curve on $\bar{\mathcal{M}}$. Then

$$\begin{aligned}\ddot{Y}(t) &= -\sum_{i=1}^m V_iV_i^T\Bigg(\operatorname{Tr}(\dot{Y}_0^T\bar{A}_i\dot{Y}_0)\ \cos\left(\sqrt{\operatorname{Tr}(\dot{Y}_0^T\bar{A}_i\dot{Y}_0)}t\right)Y_0\\ &\qquad\qquad + \sqrt{\operatorname{Tr}(\dot{Y}_0^T\bar{A}_i\dot{Y}_0)}\ \sin\left(\sqrt{\operatorname{Tr}(\dot{Y}_0^T\bar{A}_i\dot{Y}_0)}t\right)\dot{Y}_0\Bigg)\\ &= -\sum_{i=1}^m V_iV_i^T\operatorname{Tr}(\dot{Y}_0^T\bar{A}_i\dot{Y}_0)Y(t)\\ &\in N_{Y(t)}\bar{\mathcal{M}}, \qquad \text{if } A_i = \gamma_iV_iV_i^T \text{ with } \gamma_i \in \mathbb{R} \text{ for all } i = 1,\ldots,m,\end{aligned}$$

in which case $P_{Y(t)}\ddot{Y}(t) = 0$; i.e., $Y(t)$ is a geodesic on $\bar{\mathcal{M}}$. Because of the condition $\dot{Y}_0 \in \mathcal{H}_{Y_0}\mathcal{M}$, one readily checks that $\dot{Y}(t) \in \mathcal{H}_{Y(t)}\mathcal{M}$; i.e., $[Y(t)]$ is a geodesic on $\mathcal{M}$. $\square$

The point at $t = 1$ of the curve (16) which passes through $Y \in \bar{\mathcal{M}}$ and that is tangent to $Z \in \mathcal{H}_Y\mathcal{M}$ provides a retraction. In the specific cases of the spectahedron $\mathcal{S}$ and the elliptope $\mathcal{E}$, this retraction is written as

$$\mathcal{R}_Y(Z) = \cos\left(\sqrt{\mathrm{Tr}(Z^TZ)}\right)Y + \frac{\sin\left(\sqrt{\mathrm{Tr}(Z^TZ)}\right)}{\sqrt{\mathrm{Tr}(Z^TZ)}}Z$$

and

$$\mathcal{R}_Y(Z) = \cos\left(\sqrt{\mathrm{Diag}(ZZ^T)}\right)Y + \mathrm{Diag}(ZZ^T)^{\frac{-1}{2}}\sin\left(\sqrt{\mathrm{Diag}(ZZ^T)}\right)Z,$$

respectively.

An alternative retraction is obtained by "projecting" the matrix $\bar{Y} = Y + Z$ onto $\bar{\mathcal{M}}$ as follows:

$$\mathcal{R}_Y(Z) = \bar{Y} + \sum_{i=1}^{m}\alpha_i A_i \bar{Y}, \tag{17}$$

where the coefficients α_i are chosen such that the quadratic equality constraints in (12) are satisfied by (17). This actually defines α_i as a set-valued function of Z, an ambiguity that is removed by considering the branch that satisfies $\alpha_i(0) = 0$. The fact that this procedure defines a retraction (and even a second-order retraction) can be deduced from [2]. Under Assumption 2, the coefficients α_i are easily computed as the solution of the quadratic polynomial,

$$\alpha_i^2\mathrm{Tr}(\bar{Y}^TA_i^3\bar{Y}) + 2\alpha_i\mathrm{Tr}(\bar{Y}^TA_i^2\bar{Y}) + \mathrm{Tr}(\bar{Y}^TA_i\bar{Y}) = b_i.$$

For the spectahedron $\mathcal{S}$, the retraction (17) is given by

$$\mathcal{R}_Y(Z) = \frac{Y+Z}{\sqrt{\mathrm{Tr}((Y+Z)^T(Y+Z))}}.$$

In the case of the elliptope $\mathcal{E}$, (17) becomes

$$\mathcal{R}_Y(Z) = \mathrm{Diag}((Y+Z)(Y+Z)^T)^{-\frac{1}{2}}(Y+Z),$$

where $\mathrm{Diag}(X)$ denotes the diagonal matrix whose diagonal elements are those of X. In the forthcoming numerical experiments, we always choose these projection-based retractions instead of those based on geodesics because of numerical stability. In fact, when moving along geodesics, the iterate deviates with time from the equality constraints because numerical errors accumulate. Retraction (17), on the other hand, enforces these constraints at each iteration.

The per-iteration complexity of the manifold-based trust-region algorithm for solving problem (12) is dominated by the computational cost required to evaluate the objective $\bar{f}(Y)$, the gradient $\nabla\bar{f}(Y)$, and the directional derivative $\mathrm{D}(\nabla\bar{f}(Y))[Z]$. Hence, the costly operations are performed in the Euclidean space $\mathbb{R}^{n\times p}$, whereas all manifold-related operations, such as evaluating a metric, a projection, and a retraction, are of linear complexity with the dimension n.

Finally we stop the optimization algorithm once the norm of the gradient falls below a small threshold, e.g.,

$$\sqrt{\mathrm{Tr}(\mathrm{grad}\phi(Y)^T\mathrm{grad}\phi(Y))} < 10^{-6}.$$

5. Optimization on the elliptope: The maximum-cut SDP relaxation.

A first application of the proposed optimization method concerns the maximal cut of a graph.

Problem statement. The maximal cut of an undirected and weighted graph corresponds to the partition of the vertices in two sets such that the sum of the weights associated to the edges crossing between these two sets is the largest. Computing the maximal cut of a graph is an NP-complete problem. Several relaxations to that problem have been proposed. The most studied one, which is the basis of a 0.878-approximation algorithm [10], is the SDP

$$\begin{array}{ll} \min\limits_{X \in \mathbb{S}^n} & \operatorname{Tr}(AX) \\ \text{s.t.} & \operatorname{diag}(X) = \mathbf{1}, \\ & X \succeq 0, \end{array} \tag{18}$$

where $A = -\frac{1}{4}L$ with L the Laplacian matrix of the graph, the dimension n is the number of vertices of that graph, and $\mathbf{1}$ is the vector of all ones. This relaxation is tight in the case of a rank one solution.

As previously mentioned, the elliptope,

$$\mathcal{E} = \{X \in \mathbb{S}^n | X \succeq 0, \operatorname{diag}(X) = \mathbf{1}\},$$

satisfies Assumption 2. Hence, program (18) is a good candidate for the proposed framework. Using the factorization $X = YY^T$, the optimization problem is defined on the quotient manifold $\mathcal{M}_{\mathcal{E}} = \bar{\mathcal{M}}_{\mathcal{E}}/\mathcal{O}_p$, where

$$\bar{\mathcal{M}}_{\mathcal{E}} = \{Y \in \mathbb{R}_*^{n \times p} : \operatorname{diag}(YY^T) = \mathbf{1}\}.$$

The per-iteration complexity of Algorithm 1 with the inner problem (4) that is solved by trust-region optimization is of order $O(n^2 p)$ in the present context. This complexity is dominated by both the manifold-based optimization, which is $O(n^2 p)$, and the eigenvalue decomposition of the dual variable S_Y, which is $O(n^2)$. The computational cost related to the manifold-based optimization is, however, reduced in the case of matrices A that are sparse.

Competing methods. The proposed algorithm is compared in the sequel against existing methods that also rest on the low-rank factorization $X = YY^T$.

First the SDPLR algorithm that was proposed in [5] for solving SDPs uses a limited memory BFGS method (L-BFGS) to solve the fixed-rank problem (4). Each optimization of (4) is furthermore randomly initialized. Consequently, the SDPLR algorithm in not a descent algorithm.

Then the conceptually simplest method for solving the fixed-rank problem (4) is probably a gradient-descent method, which consists of moving the current iterate with a certain step size in the direction of the gradient $\nabla \bar{f}(Y)$ projected onto the tangent space $T_Y\bar{\mathcal{M}}$ of the feasible set $\bar{\mathcal{M}}$. The feasibility is then recovered after each iteration by scaling the norm of the rows of the obtained matrix to one. This method corresponds to the Riemannian gradient-descent approach discussed in [4, sect. 4.2]. To ensure convergence, the step size is computed at each iteration to satisfy the Armijo condition (see, e.g., Def. 4.2.2 in [4] with parameters $\bar{\alpha} = 1$, $\beta = 0.5$, and $\sigma = 0.01$).

To sum up, six algorithms can be deduced from this discussion: the fixed-rank problem (4) can be solved by projected gradient (Grad.), trust-region (TR) or limited memory BFGS (L-BFGS), and each of such optimization problems can be initialized either randomly (Random restart) or by using the descent direction proposed in section 3.3 (Descent restart). The initial rank p_0 is chosen to be two in the forthcoming

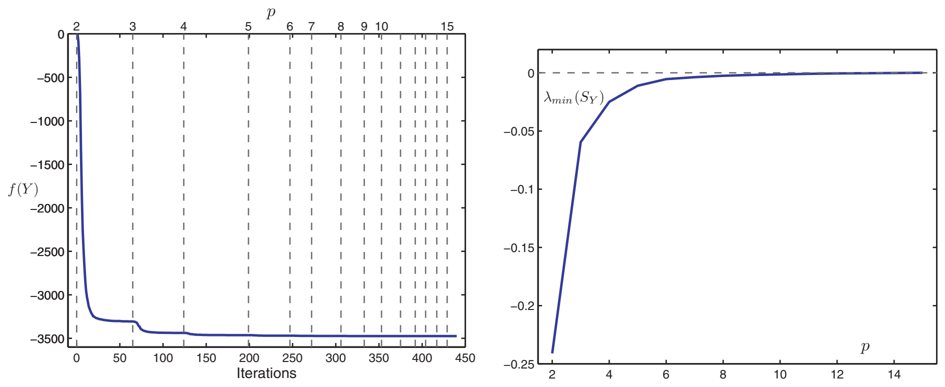

FIG. 1. *Convergence behavior of the method "Descent restart + TR" to compute the maximal cut of the graph "toruspm3-15-50." Left: Monotone decrease of the objective function $f(Y) = \mathrm{Tr}(Y^TAY)$ through the iterations (bottom abscissa) and with the rank p (top abscissa). Right: Evolution of the smallest eigenvalue of the matrix S_Y.*

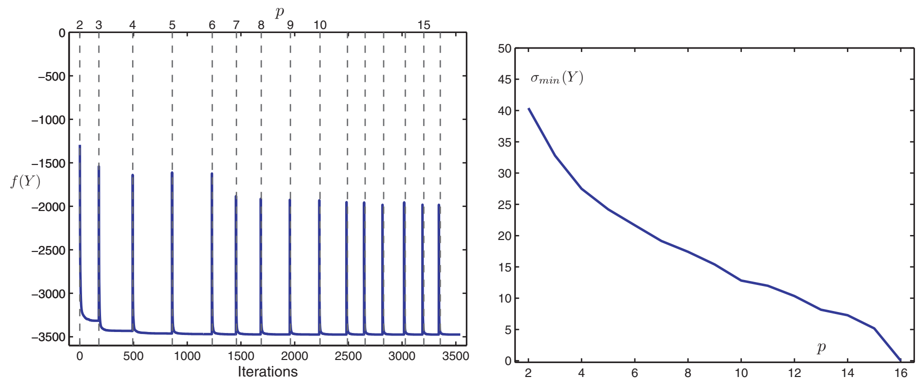

FIG. 2. *Convergence behavior of the method "Random restart + L-BFGS" to compute the maximal cut of the graph "toruspm3-15-50." Left: Evolution of the objective function $f(Y) = \mathrm{Tr}(Y^TAY)$ through the iterations (bottom abscissa) and with the rank p (top abscissa). Right: Evolution of the smallest singular value of the matrix Y.*

experiments and is incremented by one after each resolution of (4). The algorithms based on a descent restart are stopped once the dual variable S_Y is positive semidefinite (in accordance with Theorem 4), while the rank condition is used in the case of random restart (in accordance with Theorem 7). All of these algorithms have been implemented on our own in MATLAB by using the same levels of accuracies. The algorithm proposed in this paper will be denoted "Descent restart + TR," whereas the combination "Random restart + L-BFGS" is the closest fit to the SDPLR algorithm of [5].

The forthcoming computational results are obtained by using these algorithms to compute the maximal cut of a set of graphs of various sizes. More details on these graphs can be found in [5] and references therein.

Convergence plots. On Figures 1 and 2, we illustrate for both methods "Descent restart + TR" (i.e., the proposed algorithm) and "Random restart + L-BFGS" (i.e., a close approximation of the SDPLR algorithm), respectively, the evolution of the objective function as well as of the corresponding stopping criteria for comput-

TABLE 1

Computational load of various methods to compute the maximal cut of the graphs "toruspm3-8-50" ($n = 512$) and "toruspm3-15-50" ($n = 3375$). The same initialization and identical levels of accuracy are used for all methods. The results are averages for five different initializations.

		Descent restart			Random restart		
		Grad.	TR	L-BFGS	Grad.	TR	L-BFGS
toruspm3-8-50	# eval. $\bar{f}$	60535	166	9640	85524	235	14944
	# eval. grad$\bar{f}$	30266	3167	2023	42772	5243	3116
	# eval. Hess$\bar{f}$	-	3043	-	-	5029	-
	CPU time (sec)	80.9	11.0	14.2	117.0	14.0	20.6
toruspm3-15-50	# eval. $\bar{f}$	$O(10^7)$	599	62230	$O(10^7)$	900	102100
	# eval. grad$\bar{f}$	$O(10^6)$	24850	13410	$O(10^6)$	36480	21740
	# eval. Hess$\bar{f}$	-	24400	-	-	35710	-
	CPU time (sec)	44640	625	672	43860	531	713

ing the maximal cut of the graph "toruspm3-15-50," which has 3375 vertices. The monotone convergence of the proposed algorithm is depicted on the left-hand plot of Figure 1, where the number of iterations is displayed on the bottom abscissa and the top abscissa stands for the rank p. On the right-hand plot of Figure 1, the smallest eigenvalue $\lambda_{\min}$ of the dual matrix S_Y is shown to increase monotonically to zero. By comparing the right-hand plots of Figures 1 and 2, the magnitude of $\lambda_{\min}(S_Y)$ seems to provide better insight on the accuracy of the current iterate than the value of the smallest singular value of Y (i.e., $\sigma_{\min}(Y)$).

Computational speed. In Table 1, we report for both graphs "toruspm3-8-50" (512 vertices) and "toruspm3-15-50" (3375 vertices) the computational times as well as the number of evaluations of the objective function, the gradient, and the Hessian required by each of the six algorithms. These algorithms are systematically initialized with the same matrix and are stopped according to the same criterion (i.e., once the norm of the gradient gets below 10^{-6}). These results are averages for five different initializations. The conceptually simple projected gradient algorithms appear to have the worst performance. We therefore discard them from the next experiments.

The average computational times needed by the remaining algorithms are listed in Table 2 for a much larger set of graphs. The parameter n denotes the number of vertices of these graphs and corresponds thus to the size of the variable X in (18). For a given graph, the reached objective value and the rank of the computed solutions are systematically identical for all methods. These ranks indicate that the factorization $X = YY^T$ significantly reduces the size of the search space. Overall, the two methods "Descent restart + TR" and "Random restart + L-BFGS" seem to perform similarly.

Proposed geometry versus the Cholesky manifold. In Table 3, we compare the geometry discussed in section 4 of the manifold $\mathcal{M}_{\mathcal{E}}$ against the Cholesky manifold proposed in [9]. Table 3 clearly highlights that the geometry derived in section 4 provides a much more efficient representation of the quotient manifold $\mathcal{M}_{\mathcal{E}}$. The reader will notice, however, that Table 3 does not compare the algorithm proposed in this paper (trust-region on a quotient manifold) to the algorithm proposed in [9] (Newton method on an embedded manifold).

6. Optimization on the spectahedron: The sparse PCA problem.

This section presents three nonlinear programs in the context of sparse principal component

TABLE 2

Computational times of four methods based on the low-rank factorization $X = YY^T$ to compute the maximal cut of a set of graphs. The rank of the obtained solution is denoted Rank(Y). *The results are averages for five different initializations.*

			CPU time (sec)			
			Descent restart		Random restart	
Graph	n	Rank(Y)	TR	L-BFGS	TR	L-BFGS
toruspm3-8-50	512	8	11.0	14.2	14.0	20.6
torusg3-8	512	7	12.6	14.6	12	19.5
toruspm3-15-50	3375	15	625	672	531	713
torusg3-15	3375	13	882	1062	742	898
G1	800	13	55	76	138	108
G11	800	5	58	58	158	162
G14	800	13	73	93	82	68
G22	2000	18	244	337	264	385
G32	2000	6	214	276	882	844
G35	2000	17	472	590	554	303
G36	2000	19	613	874	883	308
G43	1000	13	42	66	55	85
G51	1000	14	115	166	132	87
G52	1000	15	131	182	159	82
G55	5000	19	1086	1345	984	917
G57	5000	7	1882	1573	1521	1126
G58	5000	26	6013	8066	7985	2023

TABLE 3

Computational load of the Riemannian trust-region algorithm for two different parameterizations of the manifold $\mathcal{M}_{\mathcal{E}}$ for solving the inner problem (4) *in two cases: graph "toruspm*3-8-50*" ($n = 512$) at the rank $p = 5$ and graph "toruspm*3-15-50*" ($n = 3375$) with $p = 10$. Both algorithms are initialized identically and are stopped according to the same criterion. The results are averages for five different initializations.*

		Proposed geometry	Cholesky manifold
toruspm3-8-50 for $p = 5$	# eval. $\bar{f}$	33	49
	# eval. grad$\bar{f}$	820	2181
	# eval. Hess$\bar{f}$	790	2138
	CPU time (sec)	2.0	4.2
toruspm3-15-50 for $p = 10$	# eval. $\bar{f}$	54	284
	# eval. grad$\bar{f}$	2811	161650
	# eval. Hess$\bar{f}$	2765	161420
	CPU time (sec)	55	2160

analysis (PCA) and that can be efficiently solved by means of the proposed low-rank optimization approach.

PCA is a tool that reduces multidimensional data to lower dimensions. Given a data matrix $A \in \mathbb{R}^{m \times n}$, the first principal component consists of the best rank one approximation of the matrix A in the least square sense. This decomposition is performed via estimation of the dominant eigenvector of the empirical covariance matrix $\Sigma = A^T A$. In many applications, it is of great interest to get sparse principal components, i.e., components that yield a good low-rank approximation of A while involving a limited number of nonzero elements. In the case of gene expression data, where the matrix A represents the expression of n genes through m experiments, getting factors that involve just a few genes, but still explain a great part of the variability in the data, appears to be a modeling assumption closer to the biology than the regular PCA [15]. This trade-off between variance and sparsity is the central motivation of sparse PCA methods. More details on the sparse PCA approach can be found in [17, 8] and references therein.

Sparse PCA is the problem of finding the unit-norm vector $x \in \mathbb{R}^n$ that maximizes the Rayleigh quotient of the matrix $\Sigma = A^T A$ but contains a fixed number of zeros, i.e.,

$$\begin{array}{ll} \max\limits_{x \in \mathbb{R}^n} & x^T \Sigma x \\ \text{s.t.} & x^T x = 1, \\ & \operatorname{Card}(x) \leq k, \end{array} \tag{19}$$

where k is an integer with $1 \leq k \leq n$ and $\operatorname{Card}(x)$ is the cardinality of x, i.e., the number of nonzero components. Finding the optimal sparsity pattern of the vector x is of combinatorial complexity. Several algorithms have been proposed in the literature that find an approximate solution to (19). We refer to [8] for references on these methods. Let us finally mention that the data matrix A does not necessarily have to present a sparse pattern. In the context of compressed sensing, for example, one needs to compute the sparse principal component of a matrix A that is full and sampled from a Gaussian distribution [7].

Recently, two convex relaxations have been derived that require the minimization of some nonlinear convex functions on the spectahedron $\mathcal{S} = \{X \in \mathbb{S}^m | X \succeq 0, \operatorname{Tr}(X) = 1\}$. Both of these relaxations consider a variation of (19), in which the cardinality appears as a penalty instead of a constraint, i.e., either

$$\begin{array}{ll} \max\limits_{x \in \mathbb{R}^n} & x^T \Sigma x - \rho \operatorname{Card}(x) \\ \text{s.t.} & x^T x = 1 \end{array} \tag{20}$$

or

$$\begin{array}{ll} \max\limits_{x \in \mathbb{R}^n} & x^T \Sigma x - \rho \operatorname{Card}(x)^2 \\ \text{s.t.} & x^T x = 1 \end{array} \tag{21}$$

with the parameter $\rho \geq 0$. The resolution of these two convex problems by Algorithm 1 is discussed in sections 6.1 and 6.2. A related nonconvex optimization problem, also defined on a subset of the cone of positive semidefinite matrices, is then proposed in section 6.3. We briefly report on numerical experiments for these problems; refer to [11] for more details.

6.1. A first convex relaxation to the sparse PCA problem. In [8], problem (21) is relaxed to a convex program in two steps. First a convex feasible set is obtained by lifting the unit-norm vector variable x into a matrix variable X that belongs to the spectahedron, i.e.,

$$\begin{array}{ll} \max\limits_{X \in \mathbb{S}^n} & \operatorname{Tr}(\Sigma X) - \rho \operatorname{Card}(X) \\ \text{s.t.} & \operatorname{Tr}(X) = 1, \\ & X \succeq 0. \end{array} \tag{22}$$

The relaxation (22) is tight for rank one matrices. In such cases, the vector variable x in (21) is related to the matrix variable X according to $X = xx^T$. Then for (22) to be convex, the cardinality penalty is replaced by a convex l_1 penalty, i.e.,

$$\begin{aligned} \max_{X \in \mathbb{S}^n} \quad & \operatorname{Tr}(\Sigma X) - \rho \sum_{i,j} |X_{ij}| \\ \text{s.t.} \quad & \operatorname{Tr}(X) = 1, \\ & X \succeq 0. \end{aligned} \tag{23}$$

The convex relaxation (23) can be solved by the DSPCA algorithm proposed in [8]. For Algorithm 1 to be used in this context, we still need to provide a smooth approximation to (23). This is obtained, for example, by replacing the absolute value by the differentiable function $h_\kappa(x) = \sqrt{x^2 + \kappa^2}$ with the parameter κ that is very small. A too small value of κ might, however, lead to ill-conditioned Hessians and thus to numerical problems. The convex program,

$$\begin{aligned} \max_{X \in \mathbb{S}^n} \quad & \operatorname{Tr}(\Sigma X) - \rho \sum_{i,j} h_\kappa(X_{ij}) \\ \text{s.t.} \quad & \operatorname{Tr}(X) = 1, \\ & X \succeq 0, \end{aligned} \tag{24}$$

finally fits within the framework (1). We therefore factor the variable X in the product YY^T and perform the optimization on the quotient manifold $\mathcal{M}_\mathcal{S} = \bar{\mathcal{M}}_\mathcal{S}/\mathcal{O}_p$, where

$$\bar{\mathcal{M}}_\mathcal{S} = \{Y \in \mathbb{R}_*^{n\times p} : \operatorname{Tr}(Y^T Y) = 1\}.$$

The computational complexity of Algorithm 1 with the inner problem solved by trust-region optimization (i.e., the method denoted "Descent restart + TR" in section 5) is of order $O(n^2 p)$ in the context of program (24). It should be mentioned that the DSPCA algorithm, which has been tuned to solve program (23), features a complexity of order $O(n^3)$.

Convergence plots. Figure 3 illustrates the monotone convergence of Algorithm 1 for computing a sparse principal component of a random Gaussian matrix A of size 50×50. The sparsity weight factor ρ has been chosen to 5, and the smoothing parameter κ equals 10^{-4}. The algorithm is initialized with the dominant right singular vector

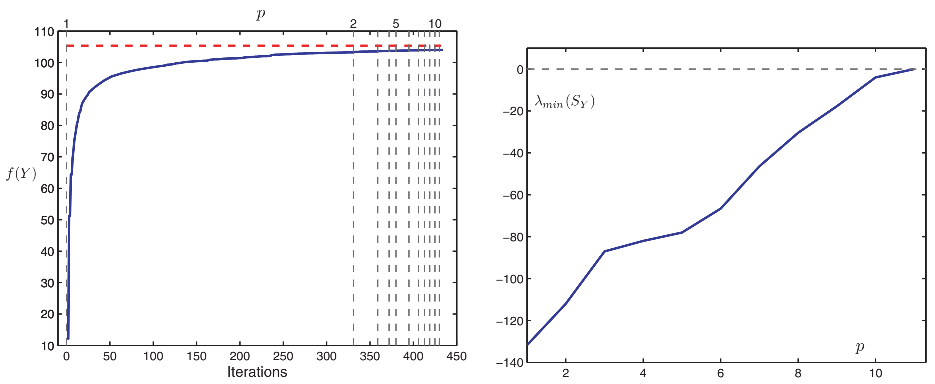

FIG. 3. *Left: Monotone increase of the objective function* $f(Y) = \operatorname{Tr}(Y^T\Sigma Y) - \rho\sum_{i,j} h_\kappa((YY^T)_{ij})$ *through the iterations (bottom abscissa) and with the rank p (top abscissa). The dashed horizontal line represents the maximum of the nonsmooth objective function in* (23). *Right: Evolution of the smallest eigenvalue of* S_Y.

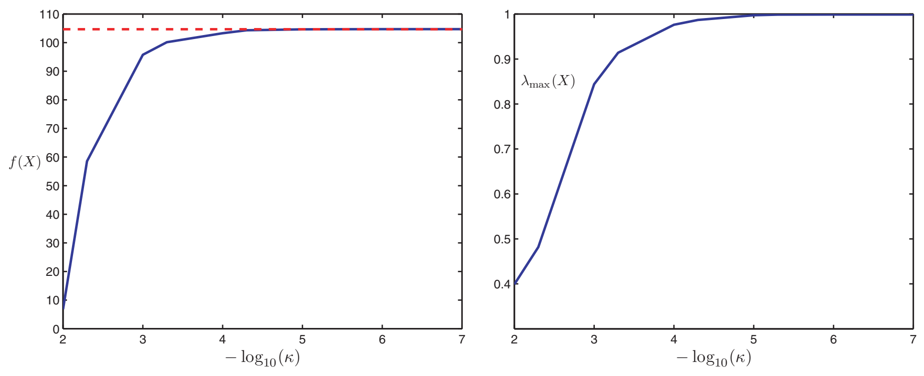

FIG. 4. *Left: Evolution of the objective value in* (24) *with the smoothing parameter κ. The dashed horizontal line represents the maximum of the nonsmooth objective function in* (23). *Right: Evolution of the largest eigenvalue of the solution X of* (24). *Since X has nonnegative eigenvalues whose sum is one, it is rank one if and only if its largest eigenvalue equals one.*

of the data matrix, i.e., the solution of (24) in the case $\rho = 0$. The maximum of the nonsmooth objective function in (23) has been computed with the DSPCA algorithm [8]. One first notices that the smooth approximation in (24) slightly underestimates the nonsmooth objective function (23). The maximizers of both (23) and (24) are, however, almost identical. It should furthermore be mentioned that all numerical experiments performed with the DSPCA algorithm for solving (23) resulted in a rank one matrix. The solution of (24) is therefore expected to be close to rank one. This intuition is confirmed by Figure 4, which shows that the solution of the smoothed problem tends to a rank one matrix once the smoothing parameter κ gets sufficiently close to zero. This explains why the improvement in terms of objective value on the left-hand plot of Figure 3 is very small for ranks larger than one. A heuristic to speed up the computations would thus consist of computing an approximate rank one solution of (24); i.e., Algorithm 1 is stopped after the iteration $p = 1$. Finally, on the right-hand plot of Figure 3, the smallest eigenvalue $\lambda_{\min}$ of the matrix S_Y appears as a way to monitor the convergence.

Computational speed. In Table 4, we compare the proposed algorithm against the above-mentioned heuristic (i.e., to compute a rank one approximation) and the

TABLE 4
Comparison of three methods for computing a sparse principal component of Gaussian data with increasing dimension. The smoothed objective function is denoted f_κ (with κ chosen to be 10^{-4}). The notation f_0 refers to the initial nonsmooth objective. Computational times are measured in seconds. The rank of the obtained solution is denoted Rank(Y). *The results are averages on five random data for each problem size.*

Dimension	Algorithm 1$_{\text{(Descent restart+TR)}}$				Heuristic			DSPCA	
of the data	Rank(Y)	f_κ	f_0	time	f_κ	f_0	time	f_0	time
10×10	2	22.6	22.7	1	22.0	22.1	0.4	22.7	4
25×25	5	46.9	47.3	3	46.3	46.8	0.8	47.3	12
50×50	10	92.2	93.1	29	79.6	81.3	2	93.2	54
100×100	15	223.0	226.1	151	204.9	212.5	6	226.7	284
200×200	25	474.0	485.1	568	384.8	414.8	22	485.9	1585
300×300	31	734.4	764.4	1279	563.4	635.2	50	766.1	4843
400×400	34	1011.9	1072.9	2725	782.0	911.5	100	1075.5	10932
500×500	38	1259.7	1368.1	4720	935.1	1138.1	155	1372.5	20652

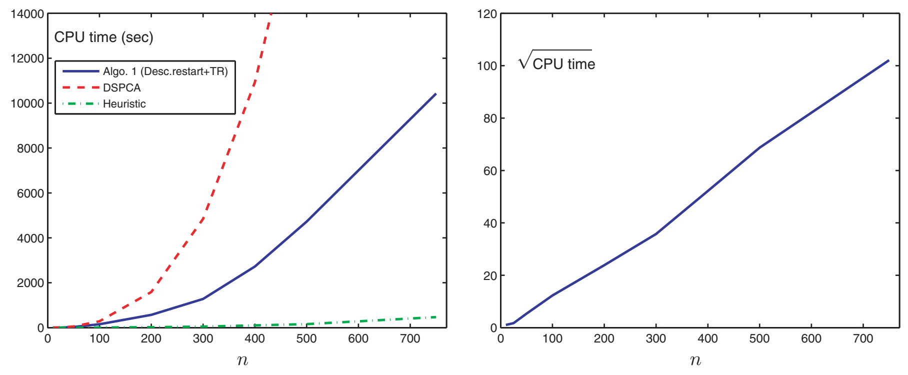

FIG. 5. *Left: Computational time of three methods for solving* (24) *(or* (23) *in the case of DSPCA) versus the problem size in the case $m = n$. Right: Square root of the computational time versus n for Algorithm* 1.

DSPCA method. The first two methods solve the smoothed problem (24), whereas the DSPCA algorithm deals with (23). The data are systematically drawn from a Gaussian distribution of zero mean and unit variance, and the choices $\rho = 5$ and $\kappa = 10^{-4}$ are made. The computational times used by these methods are also plotted on Figure 5. The quadratic complexity of Algorithm 1 with the problem size n is clearly highlighted on the right-hand plot.

6.2. A second convex relaxation to the sparse PCA problem. Problem (20) is shown in [7] to be equivalently written in the form

$$\begin{aligned} &\max_{z\in\mathbb{R}^m} && \sum_{i=1}^{n}((a_i^T z)^2 - \rho)_+ \\ &\text{s.t.} && z^T z = 1, \end{aligned} \tag{25}$$

where a_i is the i^{th} column of A and the function x_+ corresponds to $\max(0, x)$. The auxiliary variable z enables the reconstruction of the vector x: the component x_i is active if $(a_i^T z)^2 - \rho \geq 0$. As for the relaxation previously derived in section 6.1, the vector z is lifted into a matrix Z of the spectahedron,

$$\begin{aligned} &\max_{Z\in\mathbb{S}^m} && \sum_{i=1}^{n}(a_i^T Z a_i - \rho)_+ \\ &\text{s.t.} && \operatorname{Tr}(Z) = 1, \\ &&& Z \succeq 0. \end{aligned} \tag{26}$$

This program is equivalent to (25) in the case of rank one matrices $Z = zz^T$. Program (26) maximizes a convex function and is thus nonconvex. The authors of [7] have shown that, in the case of rank one matrices Z, the convex objective function in (26) equals the concave function

$$f(Z) = \sum_{i=1}^{n} \operatorname{Tr}(Z^{\frac{1}{2}}(a_i a_i^T - \rho I)Z^{\frac{1}{2}})_+,$$

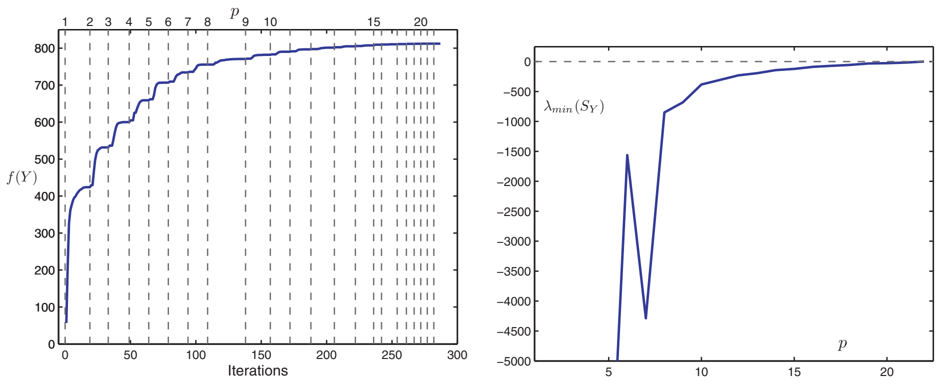

FIG. 6. *Left: Monotone increase of the objective function* (28) *through the iterations (bottom abscissa) and with the rank p (top abscissa). Right: Evolution of the smallest eigenvalue of S_Y.*

where the function $\operatorname{Tr}(X)_+$ stands for the sum of the positive eigenvalues of X. This gives the following nonsmooth convex relaxation of (20),

$$\begin{aligned} &\max_{Z\in\mathbb{S}^m} && \sum_{i=1}^{n}\operatorname{Tr}(Z^{\frac{1}{2}}(a_ia_i^T-\rho I)Z^{\frac{1}{2}})_+ \\ &\text{s.t.} && \operatorname{Tr}(Z)=1, \\ &&& Z\succeq 0, \end{aligned} \tag{27}$$

that is tight in the case of rank one solutions. This program is solved via the factorization $Z = YY^T$ and optimization on the quotient manifold $\mathcal{M}_{\mathcal{S}}$. In the case $Z = YY^T$, the objective function in (27) equals

$$f(Y)=\sum_{i=1}^{n}\operatorname{Tr}(Y^T(a_ia_i^T-\rho I)Y)_+, \tag{28}$$

which is a spectral function [7]. The evaluation of the gradient and Hessian of $f(Y)$ is based on explicit formulae derived in the papers [12, 13] to compute the first and second derivatives of a spectral function. Since we are not aware of any smoothing method that would preserve the convexity of (27), Algorithm 1 has been directly applied in this nonsmooth context. In practice, no trouble has been observed since all numerical simulations converge successfully to the solution of (27). The computational complexity of Algorithm 1 for solving (27) is of order $O(nm^2p)$. The convex relaxation (27) of the sparse PCA problem (20) appears thus well suited to treat large-scale data with $m \ll n$, such as gene expression data.

Convergence plots. Figure 6 displays the convergence of Algorithm 1 (i.e., "Descent restart + TR") for solving (27) with a random Gaussian matrix A of size $m = 100$ and $n = 500$. The sparsity parameter ρ is chosen at 5 percent of the upper bound $\bar{\rho} = \max_i a_i^T a_i$ that is derived in [7]. The smallest eigenvalue $\lambda_{\min}$ of the matrix S_Y presents a monotone decrease once it gets sufficiently close to zero.

Computational speed. Figure 7 plots the time used by our MATLAB implementation of Algorithm 1 versus the dimension n of the matrix A to compute a solution of (27). The dimension p has been fixed at 50, and A is chosen according to a Gaussian distribution. Figure 7 illustrates the linear complexity in n of the proposed sparse PCA method.

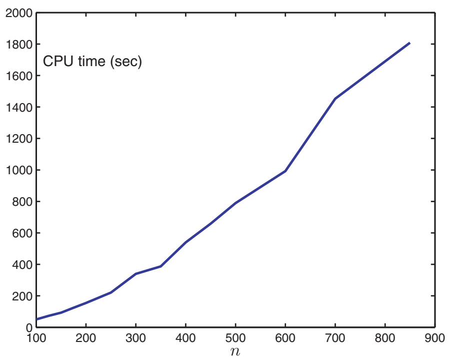

FIG. 7. *Computational time for solving* (27) *versus the problem size* n *in the case* $p = 50$.

6.3. A nonconvex relaxation to the sparse PCA problem. In this section, we use the proposed algorithm to solve the nonconvex problem,

$$\begin{aligned} &\max_{Z \in \mathbb{S}^m} && \mu f_{cvx}(Z) + (1-\mu) f_{ccv}(Z) \\ &\text{s.t.} && \operatorname{Tr}(Z) = 1, \\ &&& Z \succeq 0, \end{aligned} \tag{29}$$

with the concave function,

$$f_{ccv}(Z) = \sum_{i=1}^{n} \operatorname{Tr}(Z^{\frac{1}{2}}(a_i a_i^T - \rho I) Z^{\frac{1}{2}})_+,$$

and the convex function,

$$f_{cvx}(Z) = \sum_{i=1}^{n} (a_i^T Z a_i - \rho)_+,$$

and for the parameter $0 \le \mu \le 1$. Problem (29) might not be convex once $\mu > 0$. As previously mentioned, the functions $f_{ccv}(Z)$ and $f_{cvx}(Z)$ are identical and equal to the objective function (25) in the case of rank one matrices $Z = zz^T$.

The formulation (29) of sparse PCA is motivated by the fact that the convex relaxation (27), which is identical to (29) with $\mu = 0$, usually provides solutions with a rank larger than one, although the initial problem (25) deals with unit-norm vectors. Solving (27) at the rank $p = 1$ is, however, possible only through local maximizers. Projecting the solution Z of the relaxation (27) onto a rank one matrix zz^T might lead to better solutions. This can be done, for instance, by solving a sequence of problems of the form of (29) for an increasing value of μ from zero to one. The local solutions of (29) in the case $\mu = 1$ are, in fact, the extreme points of the spectahedron, which are rank one matrices. The proposed homotopy method projects therefore the solution of the convex relaxation (27) onto the set of rank one matrices of the spectahedron. It should be noted that the authors of [16] propose a similar approach to project doubly stochastic matrices onto permutation matrices.

Figure 8 presents computational results obtained on a random Gaussian matrix $A \in \mathbb{R}^{150 \times 50}$. The homotopy method is compared with the usual approach that projects the symmetric positive semidefinite matrix Z onto its dominant eigenvector,

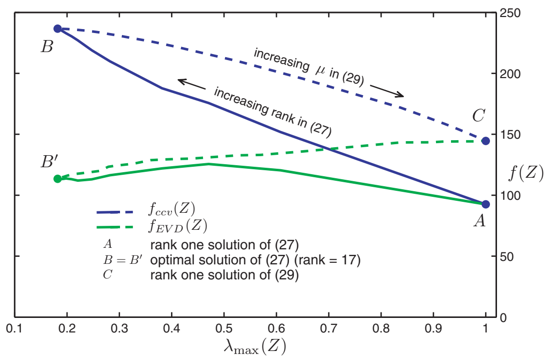

FIG. 8. *Evolution of the functions $f_{ccv}(Z)$ and $f_{EVD}(Z)$ in two situations. Continuous plots: resolution of the convex program* (27) *($\mu = 0$ in* (29)*). Dashed plots: projection of the solution of* (27) *on a rank one matrix by gradual increase of μ in* (29).

i.e., $\tilde{Z} = zz^T$, where z is the unit-norm dominant eigenvector of Z. Let $f_{EVD}(Z)$ denote the function[2]

$$f_{EVD}(Z) = f_{ccv}(\tilde{Z}) = f_{cvx}(\tilde{Z}).$$

Figure 8 uses the maximum eigenvalue of a matrix Z of the spectahedron to monitor its rank. As previously mentioned, any rank one matrix Z of the spectahedron satisfies $\lambda_{\max}(Z) = 1$. The continuous plots of Figure 8 display the evolution of the functions $f_{ccv}(Z)$ and $f_{EVD}(Z)$ during the resolution of the convex program (27), i.e., $\mu = 0$ in (29). Point A represents the solution obtained with Algorithm 1 by solving (27) at the rank $p = 1$, whereas B and B' stand for the exact solution of (27), which is of rank larger than one. The dashed plots illustrate the effect of the parameter μ that is linearly increased by steps of 0.05 between the points B and C. For a sufficiently large μ, program (29) presents a rank one solution, which is displayed by the point C. One clearly notices that the objective function of the original problem (25), which equals $f_{EVD}(Z)$, is larger at C than at both B' and A. Hence, the projection method based on (29) outperforms the projection based on the eigenvalue decomposition of Z in terms of achieved objective value and improves the solution A that is locally optimal for the rank one problem (25).

7. Conclusion.

This paper is devoted to nonlinear, and often convex, optimization problems defined on a subset of the cone of symmetric positive semidefinite matrices and that are expected to present a low-rank solution. The proposed algorithm rests on the factorization $X = YY^T$, where the number of columns of Y fixes the rank of the positive semidefinite variable X, and solves a sequence of nonconvex programs of much lower dimension than the original one. It presents a monotone convergence toward the sought solution, uses quadratic second-order optimization methods, and provides a tool to monitor the convergence, which enables evaluation of the quality of approximate solutions for the original problem. The efficiency of the approach has been illustrated on several applications, involving convex as well as nonconvex objective functions: the maximal cut of a graph and three problems in the context of sparse principal component analysis.

[2]EVD stands for eigenvalue decomposition.

REFERENCES

[1] P.-A. Absil, C. G. Baker, and K. A. Gallivan, *Trust-region methods on Riemannian manifolds*, Found. Comput. Math., 7 (2007), pp. 303–330.

[2] P.-A. Absil and Jérôme Malick, *Constructing retractions on matrix manifolds*, in preparation, 2009.

[3] P.-A. Absil, M. Ishteva, L. De Lathauwer, and S. Van Huffel, *A geometric Newton method for Oja's vector field*, Neural Comput., 21 (2009), pp. 1415–1433.

[4] P.-A. Absil, R. Mahony, and R. Sepulchre, *Optimization Algorithms on Matrix Manifolds*, Princeton University Press, Princeton, NJ, 2008.

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

[6] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, Cambridge, UK, 2004.

[7] A. d'Aspremont, F. R. Bach, and L. El Ghaoui, *Full regularization path for sparse principal component analysis*, in Proceedings of the 24th International Conference on Machine Learning ICML 2007, Corvallis, OR, International Machine Learning Society, 2007, pp. 177–184.

[8] A. d'Aspremont, L. El Ghaoui, M. I. Jordan, and G. R. G. Lanckriet, *A direct formulation for sparse PCA using semidefinite programming*, SIAM Rev., 49 (2007), pp. 434–448.

[9] I. Grubisic and R. Pietersz, *Efficient rank reduction of correlation matrices*, Linear Algebra Appl., 422 (2007), pp. 629–653.

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

[11] M. Journée, *Geometric Algorithms for Component Analysis with a View to Gene Expression Data Analysis*, Ph.D. thesis, Université de Liège, Liège, Belgium, 2009.

[12] A. S. Lewis, *Derivatives of spectral functions*, Math. Oper. Res., 21 (1996), pp. 576–588.

[13] A. S. Lewis and H. S. Sendov, *Twice differentiable spectral functions*, SIAM J. Matrix Anal. Appl., 23 (2001), pp. 368–386.

[14] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed., Springer Ser. Oper. Res. Financ. Eng., Springer, New York, 2006.

[15] A. Teschendorff, M. Journée, P.-A. Absil, R. Sepulchre, and C. Caldas, *Elucidating the altered transcriptional programs in breast cancer using independent component analysis*, PLoS Comput. Biol., 3 (2007), pp. 1539–1554.

[16] M. Zaslavskiy, F. Bach, and J.-P. Vert, *A path following algorithm for the graph matching problem*, IEEE Trans. Pattern Analysis Mach. Intelligence, 31 (2009), pp. 2227–2242.

[17] H. Zou, T. Hastie, and R. Tibshirani, *Sparse principal component analysis*, J. Comput. Graph. Statist., 15 (2006), pp. 265–286.