DOI: https://doi.org/10.15446/recolma.v55n1.99100

Revista Colombiana de Matemáticas
Volumen 55(2021)1, páginas 109-123

A Spectral Gradient Projection Method for the Positive Semi–definite Procrustes Problem

Un Método de Gradiente Proyectado Espectral para el Problema de Mínimos Cuadrados Matricial Semi–definido Positivo

Harry Oviedo

Fundação Getulio Vargas –
Escola de Matemática Aplicada,
FGV\EMAp, Brazil, Rio de Janeiro, Brasil

Abstract. This paper addresses the positive semi-definite procrustes problem (PSDP). The PSDP corresponds to a least squares problem over the set of symmetric and semi-definite positive matrices. These kinds of problems appear in many applications such as structure analysis, signal processing, among others. A non-monotone spectral projected gradient algorithm is proposed to obtain a numerical solution for the PSDP. The proposed algorithm employs the Zhang and Hager's non-monotone technique in combination with the Barzilai and Borwein's step size to accelerate convergence. Some theoretical results are presented. Finally, numerical experiments are performed to demonstrate the effectiveness and efficiency of the proposed method, and comparisons are made with other state-of-the-art algorithms.

Key words and phrases. Non-monotone algorithm, Constrained optimization, Symmetric positive semi-definite constraints, Least-Square problems.

2020 Mathematics Subject Classification. 49Q99, 65K05, 90C22, 90C26, 90C27, 90C30.

Resumen. En este artículo abordamos el problema de mínimos cuadrados lineales sobre el conjunto de matrices simétricas y definidas positivas (PSDP). Esta clase de problemas surge en un gran número de aplicaciones tales como análisis de estructuras, procesamiento de señales, análisis de componentes principales, entre otras. Para resolver este tipo de problemas, proponemos

un método de gradiente proyectado espectral no-monótono. El algoritmo propuesto usa la técnica de globalización no-monótona de Zhang y Hager, en combinación con los tamaños de paso de Barzilai y Borwein para acelerar la convergencia del método. Además, presentamos y comentamos algunos resultados teóricos concernientes al algoritmo desarrollado. Finalmente, llevamos a cabo varios experimentos numéricos con el fin de demostrar la efectividad y la eficiencia del nuevo enfoque, y realizamos comparaciones con algunos métodos existentes en la literatura.

Palabras y frases clave. Algoritmo no-monótono, optimización con restricciones, restricciones simétricas y semi definidas positivas, problema de mínimos cuadrados.

1. Introduction

The positive semi-definite Procrustes problem (PSDP) is defined as follows: given two rectangular matrices $A, B \in \mathbb{R}^{n\times m}$, we want to find a symmetric and positive semi–definite matrix $X^* \in \mathbb{R}^{n\times n}$ that solves the following optimization problem

$$\min_{X\in\mathbb{R}^{n\times n}} \mathcal{F}(X) \quad s.t. \quad X \in \mathcal{S}_+(n), \tag{1}$$

where $\mathcal{F} : \mathbb{R}^{n\times n} \to \mathbb{R}$ is defined as $\mathcal{F}(X) = \frac{1}{2}||XA - B||_F^2$, here $||Z||_F$ denotes the Frobenius norm of $Z \in \mathbb{R}^{r\times k}$ and $\mathcal{S}_+(n)$ represents the set of the symmetric and positive semi-definite n-by-n matrices with real entries, i.e.

$$\mathcal{S}_+(n) = \{X \in \mathbb{R}^{n\times n} : X^\top = X,\ v^\top Xv \geq 0,\ \forall v \in \mathbb{R}^n\}.$$

Problem (1) arises frequently in different applications such as: analysis of structures [5, 17], signal processing [14], estimation of correlation matrices [2], among others. It is well known that the feasible set $\mathcal{S}_+(n)$ of the problem (1), is a closed convex cone of dimensions $n \times (n+1)/2$, [9]. From this result and the fact that the objective function $\mathcal{F}$ is continuous, the existence of at least one global minimizer of problem (1) is guaranteed. Additionally, if A is full rank then there exists a unique solution for (1), for more details see [17]. In addition, since $\mathcal{F}$ is a convex function, this converts (1) in a convex minimization problem, which is relatively easy to solve. There are also some particular cases of the problem (1) that have an analytical formula for the solution, for example, when $A = I_m$ [9], when $\texttt{rank}(A) = 1$ [8], or when X is considered a diagonal matrix [8].

Although there are particular cases where problem (1) has a closed solution, in most cases, such as in real applications, a solution is only possible computationally, by an iterative method capable of dealing with non-stationary points, generating a sequence of feasible points that converge to a local minimizer. However, designing efficient algorithms that generate a feasible sequence of points is generally a difficult task, because it usually leads to use some projection operator, which requires computing spectral decompositions, and this is computationally expensive for large scale situations.

Due to the vast number of applications that problem (1) captures, many researchers are interested in studying this problem, both from a theoretical point of view, as well as from the design of new efficient approaches. In [9, 10] the authors study a problem related to (1) theoretically, where they derive general analytical formulas for the solution in some particular cases. Additionally, they provide sufficient and necessary conditions to guarantee the existence of the solution. On the other hand, the *gradient projection method* was implemented by Nicolas Gillis et al. in [8]. Specifically, Nicolas Gillis et.al. propose an algorithm called "FGM", which is an accelerated version of the classical gradient projection method, which uses the Nesterov [11] acceleration technique. In essence, the FGM is an implementation of the algorithm that appears in [11]. In [8], a method called "AN-FGM" is also proposed which is a semi-analytic approach that reduces the problem (1) to the case when A is diagonal and then uses the FGM to address a more easy problem, this proposal looks quite efficient to deal with problems where A is ill-conditioned. Another alternative to compute a numerical solution of problem (1), has been studied in [1], where the authors propose an algorithm called "Parallel tangents" that is based on the gradient projection method that incorporates an over-relaxation step. One drawback of this parallel tangents method is that it does not guarantee optimal convergence. On the other hand, two algorithms that were designed to solve convex optimization problems over $\mathcal{S}_+(n)$, "SDPT3" [16] and "QSDP" [15] can be used to solve the problem (1).

In this work, we study the numerical behaviour of a spectral gradient projection method to address the positive semi-definite procrustes problem from a practical point of view. In particular, we adopt a gradient projection scheme with the non-monotone globalization technique proposed by Zhang and Hager in [18], in combination with the step size proposed by Barzilai and Borwein in [3]. Subsequently, we present some computational experiments, in order to illustrate the effectiveness of the proposed method, solving the PSDP problem under many conditioning situations of A. Our main contribution is the implementation of an efficient gradient projection procedure using MATLAB, and the numerical comparison of the performance of such method against other existing algorithms of the state of the art.

The rest of this work is organized as follows. In the next section some important notations and tools are introduced as background for the article. In Section 3, the update formula of our proposed method is presented. Subsection 3.1, addresses the problem of selecting the step size of the proposed method and describes a non-monotone globalization technique to regulate such step size, and we culminate this subsection presenting our new approach. A derivation of the proposed method from the algorithm presented in [7] is discussed in subsection 3.2. In Section 4, several numerical experiments are carried out in order to demonstrate the effectiveness and efficiency of our procedure. Finally, the conclusions are presented in Section 5.

2. Notations and Important Tools

In this section, we present some fundamental concepts and tools that we will need in this paper. Let's denote by $\langle A, B\rangle := \sum_{i,j} A_{i,j}B_{i,j} = Tr[A^\top B]$ to the usual inner product on the matrix space $\mathbb{R}^{n\times m}$, here $Tr[A]$ denotes the trace of A. Given a differentiable function $\mathcal{F} : \mathbb{R}^{n\times m} \to \mathbb{R}$, the gradient of $\mathcal{F}$ respect to X is denoted by $\nabla\mathcal{F}(X) = \left(\frac{\partial\mathcal{F}(X)}{\partial X_{i,j}}\right)$. The directional derivative of $\mathcal{F}$ at $X \in \mathbb{R}^{n\times m}$ in the direction $Z \in \mathbb{R}^{n\times m}$ is

$$\nabla\mathcal{F}(X)[Z] := \lim_{t\to 0} \frac{\mathcal{F}(X + tZ) - \mathcal{F}(X)}{t} = \langle \nabla\mathcal{F}(X), Z\rangle. \tag{2}$$

Another tool that we use is the projection operator over the feasible set $\mathcal{S}_+(n)$ which is defined below.

Definition 2.1. Let $X \in \mathbb{R}^{n\times n}$ be a real square matrix. The projection operator $\pi : \mathbb{R}^{n\times n} \to \mathcal{S}_+(n)$ over $\mathcal{S}_+(n)$ is defined by

$$\pi(X) = \arg\min_{P\in\mathbb{R}^{n\times n}} ||P - X||_F, \quad s.t. \quad P \in \mathcal{S}_+(n). \tag{3}$$

Note that the projection of any arbitrary matrix $X \in \mathbb{R}^{n\times n}$ is defined from an optimization problem, however, problem (3) has a closed solution, this fact is established below.

Proposition 2.2. *Let $X \in \mathbb{R}^{n\times n}$ be a real square matrix. Then $\pi(X)$ is well-defined. Moreover, consider the symmetric part of X, that is, $X_{sym} = \frac{1}{2}(X^\top + X)$ and let $X_{sym} = V\Sigma V^\top$ be the spectral decomposition of X_{sym}, then $\pi(X) = V(\max(\Sigma, 0))V^\top$.*

Proof. The proof of this proposition appear in [9]. ☑

3. A feasible Update Scheme

Since $\mathcal{F}$ is smooth, a natural idea is to compute the next iterates as $Y(\tau) = X - \tau\nabla\mathcal{F}(X)$, where $X \in \mathcal{S}_+(n)$ is the previous iterate and $\tau > 0$ represents the step size. The drawback of this approach is that the new point $Y(\tau)$ may not satisfy the constraints of problem (1). In order to overcome this issue, we consider the well-known *projected gradient method* [4] which computes the new iterate $Z(\overline{\tau})$ as a point on the curve

$$Z(\tau) = \pi(X - \tau\nabla\mathcal{F}(X)). \tag{4}$$

Observe that equation (4) guarantees that the new iterate preserves the feasibility. On the other hand, there are different techniques to select the step size τ. The condition that is usually used for the *gradient projection method* is known as "Armijo's condition on the arc of projection" [4]. This condition imposes to

choose the step size τ_k, at the k-th iteration, as the largest positive number τ, such that verifies the following inequality

$$\mathcal{F}(Z_k(\tau)) < \mathcal{F}(X_k)) + \sigma \nabla \mathcal{F}(X_k)[Z_k(\tau) - X_k], \tag{5}$$

where $\sigma \in (0, 1)$ and $Z_k(\tau) = \pi(X_k - \tau \nabla \mathcal{F}(X_k))$. The Armijo condition (5) is used in combination with a heuristic so-called *backtracking* in order to find an appropriate step size that satisfies the condition (5), for more details about the backtracking strategy see [4, 12].

Note that if we ensure that the directional derivative $\nabla \mathcal{F}(X_k)[Z_k(\tau) - X_k] < 0$ for all k, then we obtain a sequence $\{X_k\}$ of points such that the corresponding sequence $\{\mathcal{F}(X_k)\}$ is monotonically decreasing. With the purpose of accelerating the convergence of the gradient projection scheme (4), we adopt the non-monotone globalization technique proposed by Zhang and Hager in [18] combined with the Barzilai and Borwein step sizes [3] which usually accelerate the convergence of gradient-based methods. This strategy is described in the next section.

3.1. The Barzilai-Borwein Step Sizes.

In this subsection, we are focused on a non-monotone strategy for the step size selection as well as to present the proposed algorithm in detail.

It is well-known that sometimes the Barzilai and Borwein step sizes [3] can improve the performance of the gradient-based algorithms without increasing too much the computational cost of the procedure. Typically, this technique considers the steepest descent method, and proposes to choose any of two step sizes, presented below, at the k-th iteration,

$$\tau_k^{BB1} = \frac{||S_{k-1}||_F^2}{Tr[S_{k-1}^\top Y_{k-1}]}, \text{ or } \tau_k^{BB2} = \frac{Tr[S_{k-1}^\top Y_{k-1}]}{||Y_{k-1}||_F^2}, \tag{6}$$

where $S_{k-1} = X_k - X_{k-1}$ and $Y_{k-1} = \nabla \mathcal{F}(X_k) - \nabla \mathcal{F}(X_{k-1})$. Since the values τ_k^{BB1} and τ_k^{BB2} (BB-steps) could be negative, we used their absolute value to avoid negative step sizes that involve growth in the objective function. For more details see [3, 13]. Since the BB-steps does not necessarily decrease the objective function values at each iteration, it can invalidate convergence. However, this issue can be overcome by using a globalization technique, which guarantees global convergence by regulating the step sizes in (6), see [6, 13]. Taking in mind this considerations, we adopt a non-monotone line search method based on a strategy in [18], in our proposed algorithm. Specifically, the iterates are recursively updated as $X_{k+1} := Z_k(\tau_k) = \pi(X_k - \tau_k \nabla \mathcal{F}(X_k))$, where $\tau_k = \eta^h \tau_k^{BB1}$ or $\tau_k = \eta^h \tau_k^{BB2}$, where h is the smallest integer number that verify the following condition

$$\mathcal{F}(Z_k(\tau_k)) \leq C_k + \sigma \nabla \mathcal{F}(X_k)[Z_k(\tau_k) - X_k], \tag{7}$$

where C_{k+1} is defined as the convex combination of C_k and $\mathcal{F}(X_{k+1})$ given by $C_{k+1} = \frac{\mathcal{F}(X_{k+1})+\gamma Q_k C_k}{Q_{k+1}}$, where $Q_{k+1} = \gamma Q_k + 1$, with $Q_0 = 1$. The proposed non-monotone gradient projection method to deal with the numerical solution of the problem (1) is summarized in Algorithm 1.

Algorithm 1 OptPSDP

Require: $X_0 \in \mathcal{S}_+(n)$, $X_{-1} = X_0 + I_n$, $\tau > 0$, $0 < \tau_m \leq \tau_M$, $\sigma, \eta \in (0,1)$, $\gamma \in [0,1)$, $Q_0 = 1$, $C_0 = \mathcal{F}(X_0)$, $k = 0$.

1: **while** not convergent **do**
2: **while** $\mathcal{F}(Z_k(\tau)) > C_k + \sigma\tau\mathcal{F}(X_k)[Z_k(\tau) - X_k]$ **do**
3: $\tau = \eta\tau$,
4: **end while**
5: $X_{k+1} = Z_k(\tau)$, according to (4).
6: Compute $Q_{k+1} = \gamma Q_k + 1$ and $C_{k+1} = (\gamma Q_k C_k + \mathcal{F}(X_{k+1}))/Q_{k+1}$.
7: Take $\tau = |\alpha_{k+1}^{BB1}|$ or well $\tau = |\alpha_{k+1}^{BB2}|$, according to (6).
8: $\tau = \max(\min(\tau, \tau_M), \tau_m)$.
9: $k = k + 1$.
10: **end while**
11: $X^* = X_k$.

***Remark* 3.1.** Note that if we select $\gamma = 0$ in the previous algorithm, then Algorithm 1 is reduced to the classical gradient projection method. Observe also that the Algorithm 1 can be used to minimize any objective smooth function over the matrix set $\mathcal{S}_+(n)$, however, the interest of this work is focused on the particular problem (1).

Note that the step 5, in Algorithm 1, is the step that requires the most computational effort, because it needs to use the operator projection defined in (3), which in turn requires to compute a spectral decomposition, which it is computationally inefficient. In order to avoid the calculation of such spectral decomposition, in each step, we propose the following idea: first note that if the symmetric part of $Y_k = X_k - \tau_k \nabla\mathcal{F}(X_k)$ is positive definite then this matrix coincides with its projection over $\mathcal{S}_+(n)$. Thus, we propose to use the Cholesky's factorization to make the Algorithm 1 more efficient. Specifically, in the step 5, we try to compute the Cholesky factorization of $\frac{Y_k+Y_k^\top}{2}$, if no error is generated, then X_{k+1} is updated by $X_{k+1} = \frac{Y_k+Y_k^\top}{2}$, otherwise $X_{k+1} = Z_k(\tau_k)$ is updated using the projection operator. In Section 4, we demostraste numerically the efficiency of this strategy on some numerical tests.

3.2. Another Point of View of Algorithm 1.

In this section we derive Algorithm 1 from an algorithm proposed recently by Francisco et al. in [7]. In addition, we establish a convergence result related to our Algorithm 1.

In [7] the authors propose a globally convergent non-monotonous algorithm to numerically solve the following optimization problem,

$$\min f(x) \quad s.t. \quad x \in \Omega, \tag{8}$$

where Ω is a closed subset of $\mathbb{R}^n$ and $f : \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable function on $\hat{\Omega}$ such that $\Omega \subset \hat{\Omega}$. The proposed algorithm by Francisco et.al. builds a sequence of iterates as follows: given the current point $x_k \in \Omega$, $\rho_k > 0$ a positive scalar and two symmetric matrices A_k, B_k, with A_k definite positive, then the next trial point x_{k+1} is computed as the argument that minimizes the quadratic model

$$\min_{x\in\Omega} Q_k(x) = \langle \nabla f(x_k), x - x_k\rangle + \frac{1}{2}(x - x_k)^\top (B_k + \rho_k A_k)(x - x_k), \tag{9}$$

where ρ_k works as a regularization parameter. This method is based on the ideas of the trust region methods [12] and the well-known method of Levenberg-Marquardt [12]. The authors in [7], combine these ideas with the non-monotone technique proposed by Zhang and Hager [18], and thus obtain a very general method to solve the non-linear optimization problem (8).

The rest of this subsection is dedicated to demonstrate that the Algorithm 1 can be seen as a particular case of the algorithm proposed in [7]. To do this, it is sufficient to demonstrate that the update formula of our method (4) is equivalent to solve a quadratic model on $\mathcal{S}_+(n)$, due to the non-monotone strategy to choose the step size is the same for the two algorithms.

Proposition 3.2. *Let $X_k \in \mathcal{S}_+(n)$ be the point generated by Algorithm 1 at the k-th iteration. If $\tau > 0$ then $Z_k(\tau) = \pi(X_k - \tau\nabla\mathcal{F}(X_k))$ is the minimum of the following quadratic model,*

$$\mathcal{Q}_k(X) = Tr[\nabla\mathcal{F}(X_k)^\top (X - X_k)] + \frac{1}{2\tau}||X - X_k||_F^2, \tag{10}$$

over the set $\mathcal{S}_+(n)$.

Proof. Since $Z_k(\tau) = \pi(X_k - \tau\nabla\mathcal{F}(X_k))$ then $Z_k(\tau)$ is a solution of

$$\min \mathcal{J}(X) = \frac{1}{2}||X - (X_k - \tau\nabla\mathcal{F}(X_k))||_F^2, \quad s.t. \quad P \in \mathcal{S}_+(n). \tag{11}$$

From the definition of $J(X)$ and using trace properties we have

$$\begin{aligned}\mathcal{J}(X) &= \frac{1}{2}Tr[(X-(X_k-\tau\nabla\mathcal{F}(X_k)))^\top(X-(X_k-\tau\nabla\mathcal{F}(X_k)))]\\ &= \frac{1}{2}Tr[X^\top X-2X^\top X_k+2\tau X^\top\nabla\mathcal{F}(X_k)+X_k^\top X_k-2\tau X_k^\top\nabla\mathcal{F}(X_k)\\ &\quad+\tau^2\nabla\mathcal{F}(X_k)^\top\nabla\mathcal{F}(X_k)].\end{aligned}$$

Now, since $\tau^2\nabla\mathcal{F}(X_k)^\top\nabla\mathcal{F}(X_k)$ is constant, then minimizing $\mathcal{J}(\cdot)$ is equivalent to minimize the function $\hat{\mathcal{J}}(\cdot)$ given by

$$\hat{\mathcal{J}}(X)=\frac{1}{2}Tr[X^\top X-2X^\top X_k+2\tau X^\top\nabla\mathcal{F}(X_k)+X_k^\top X_k-2\tau X_k^\top\nabla\mathcal{F}(X_k)].$$

Rewriting this last result we arrive at

$$\hat{\mathcal{J}}(X)=\tau Tr[X^\top\nabla\mathcal{F}(X_k)-X_k^\top\nabla\mathcal{F}(X_k)]+\frac{1}{2}Tr[X^\top X-2X^\top X_k+X_k^\top X_k], \tag{12}$$

or equivalently

$$\hat{\mathcal{J}}(X)=\tau\left(Tr[\nabla\mathcal{F}(X_k^\top)(X-X_k)]+\frac{1}{2\tau}||X-X_k||_F^2\right). \tag{13}$$

Then, since τ is constant for the optimization process over $\mathcal{S}_+(n)$, we have that minimize $\hat{\mathcal{J}}(\cdot)$ over $\mathcal{S}_+(n)$, is equivalent to minimize the quadratic function $\mathcal{Q}_k(X)$ defined in (10) over the set $\mathcal{S}_+(n)$, which completes the proof. ☑

Note that Proposition 3.2 shows that the Algorithm 1 is a particular case of the algorithm proposed by Francisco et al. [7], obtained by taking $\rho_k = 1$, B_k the null matrix and $A_k = \frac{1}{\tau_k}I_n$ at each iteration. This result implies that the Algorithm 1 is globally convergent, which it is established in Theorem 3.3.

Theorem 3.3. *Let $\{X_k\}$ be a sequence generated by Algorithm 1. Assume that $\gamma < 1$, then every accumulation points of $\{X_k\}$ is a stationary point of the problem (1).*

4. Numerical Experiments

In this section, we illustrate the effectiveness and efficiency of the proposed algorithm (Algorithm 1: OptPSDP) on several positive semi-definite procrustes problems generated synthetically. An implementation in Matlab of OptPSDP is available at: http://www.mathworks.com/matlabcentral/fileexchange/64597-spectral-projected-gradient-method-for-the-positive-semi-definite-procrustes-problem.

All computational experiments were carried out using Matlab 7.0 in an intel (R) CORE (TM) i7-4770 processor, 3.40 GHz CPU with 500 Gb of HD and 16 Gb of Ram. In all experiments the following values are used for the OptPSDP

algorithm: σ = 1e-4, τ_0 = 1e-3, $\tau_{\min}$ = 1e-20, $\tau_{\max}$ = 1e20, ϵ = 1 e-5, $\gamma = 0.85$ and $\eta = 0.2$. For the other methods we used the default parameters of each algorithm, except for the tolerance fixed to ϵ = 1e-5. As a maximum number of iterations $N = 10000$ was selected for all algorithms.

Method	**Nitr**	**Nfe**	**Time**	**XErr**	**Fval**	**Global Error**
	E1: n = 100, m = 70, problem = 1, $\gamma = 0.8$					
Grad	746	747	0.808	9.96e-6	1.06e-6	17.3
FGM	1756	1757	1.889	9.99e-6	7.43e-25	61.3
ParTan	74	75	**0.107**	2.91e-6	1.02e-8	17.4
OptPSDP	102	103	0.123	8.21e-6	5.86e-7	17.3
	E2: n = 150, m = 100, problem = 1, $\gamma = 0.85$					
Grad	1356	1357	3.166	9.98e-6	2.29e-6	27.7
FGM	2091	2092	4.856	9.99e-6	2.37e-24	101.2
ParTan	101	102	**0.316**	2.38e-6	1.17e-8	27.8
OptPSDP	153	154	0.393	8.68e-6	1.42e-6	27.6
	E3: n = 1000, m = 100, problem = 1, $\gamma = 0.85$					
Grad	8	9	1.656	4.91e-6	1.60e-11	4.76e-7
FGM	8	9	1.588	2.68e-6	2.22e-12	1.31e-7
ParTan	7	8	2.287	3.83e-6	9.28e-12	3.70e-7
OptPSDP	7	8	**1.438**	5.86e-6	5.79e-12	2.72e-7
	E4: n = 1500, m = 1500, problem = 1, $\gamma = 0.85$					
Grad	8	9	5.471	4.75e-6	1.65e-11	4.52e-7
FGM	8	9	5.261	1.15e-6	5.64e-13	7.16e-8
ParTan	7	8	7.261	5.15e-6	1.64e-11	4.90e-7
OptPSDP	8	9	**4.752**	2.72e-6	2.41e-12	1.24e-7

TABLE 1. Numerical results for well conditioned PSDP (problem = 1).

In the rest of this section, we denote by "Nitr" the average number of iterations, "Nfe" the average number of functions evaluations, "Time" the average execution time in seconds, "Fval" the average value of the evaluation of the objective function at point $\hat{X}$ which denotes the optimum estimated by each algorithm, "Error" the average global error, that is, $||X^* - \hat{X}||_F$, where X^* denotes the global optimum each PSDP problem, and finally we denote by "XErr", the average error $||\hat{X} - X_k||_F$, and X_k penultimate point generated by each algorithm. In addition, we denote by **Grad** to the classical gradient projection method proposed in [8], **FGM** denotes the accelerated gradient projection method proposed in [8], **ParTan** denotes parallel tangent method introduced in [1] and **OptPSDP** denotes our proposal.

For the numerical experiments, we consider problem (1) where the matrix $A \in \mathbb{R}^{n\times m}$ is build as $A = P\Lambda Q^\top$, where $P \in \mathbb{R}^{n\times n}$ and $Q \in \mathbb{R}^{m\times m}$ are orthogonal matrices randomly generated and $\Lambda \in \mathbb{R}^{n\times m}$ is a diagonal matrix defined as we explain below. The starting point X_0 was generated as $X_0 = \pi(\bar{X}_0)$, where $\bar{X}_0$ was randomly generated. In order to monitoring the behavior of the algorithms, the optimal solution is generated by $X^* = \pi(\tilde{X})$ where

$\tilde{X} \in \mathbb{R}^{n\times n}$ was randomly generated. Then, the matrix $B \in \mathbb{R}^{n\times m}$ was taken as $B = XA$, in this way, X^* is a global optimum of the problem (1) with optimal value zero, i.e. $\mathcal{F}(X^*) = 0$. All random values were generated following a standard normal distribution using the `randn` function of Matlab.

Method	Nitr	Nfe	Tiempo	XErr	Fval	Global Error
	E5: n = 30, m = 10, problem = 2, $\gamma = 0.85$					
Grad	4392	4393	0.81	1.15e-6	6.2e-5	9.7
FGM	1289	1290	0.251	9.96e-6	3.55e-9	16.5
ParTan	187	188	**0.041**	5.04e-6	3.49e-6	9.8
OptPSDP	392	395	0.086	8.5e-6	1.22e-5	9.7
	E6: n = 100, m = 50, problem = 2, $\gamma = 0.55$					
Grad	10000	10001	12.123	3.4e-5	1.41e-2	25.3
FGM	5181	5182	6.4699	9.85e-6	5.1e-6	54.9
ParTan	1073	1074	**1.553**	8.32e-6	1.27e-4	25.6
OptPSDP	1221	1231	1.57	9.68e-6	2.4e-3	25.3
	E7: n = 60, m = 60, problem = 2, $\gamma = 0.85$					
Grad	6149	6150	2.731	1.16e-6	6.97e-4	1.93e-2
FGM	442	443	0.236	9.86e-6	2.04e-7	3.77e-4
ParTan	291	292	0.196	4.63e-6	2.51e-6	1.1e-3
OptPSDP	318	322	**0.184**	8.6e-6	1.24e-4	9.3e-3
	E8: n = 120, m = 120, problem = 2, $\gamma = 0.55$					
Grad	9962	9963	14.679	4.55e-6	1.46e-1	2.58e-1
FGM	784	785	1.394	9.93e-6	8.4e-7	6.82e-4
ParTan	498	499	**1.14**	5.1e-6	1.68e-5	2.7e-3
OptPSDP	728	743	1.387	9.08e-6	1.7e-3	3.11e-2

TABLE 2. Numerical results for ill conditioned PSDP (problem = 2).

In addition, we consider the following three distributions of the entries of Λ,

Problema 1: The Λ diagonal entries are generated by a truncated normal distribution in the interval [10,12].

Problema 2: The diagonal of Λ is given by $\lambda_{ii} = i + 2r_i$, where r_i is a randomly generated from the uniform distribution in the interval [0,1].

Problema 3: Each element of the diagonal matrix Λ is generated as $\lambda_{ii} = 1 + \frac{99(i-1)}{m+1} + 2r_i$, with r_i is a randomly generated from the uniform distribution in the interval [0,1].

Observe that if the Λ is generated following the structure of **Problema 1** then A is a well-conditioned matrix, while it is generated by the diagonal structures described in **Problema 2** and **Problema 3** then A is a ill-conditioned matrix. In order to study the numerical behavior and performance of all methods, we consider several size of problems PSDP and different conditions number of A. In all tables, we present the averages of the comparing values obtained by each algorithm in a total of 50 independent instances.

In the first experiment, we study the efficiency of the proposed method on well-conditioned PSDP problems. Table 1 summarizes the numerical results of this comparison. From Table 1, we observe that the fastest methods are **ParTan** and **OptPSDP**. In addition, it's seen that if A is rectangular then the most efficient method in terms of CPU-time is **ParTan**. However, clearly we note that our proposal is more efficient for problems where A is square. According to the error *XErr*, all algorithms reach an order less than 1e-5 and additionally, we can see that the value *Fval* is close to zero for all algorithms.

Method	Nitr	Nfe	Time	XErr	Fval	Global Error
	E9: n = 50, m = 10, problem = 3, $\gamma = 0.55$					
Grad	10000	10001	3.539	6.17e-5	1.45e-1	19.51
FGM	6233	6234	2.287	1.08e-6	1.11e-5	26.29
ParTan	514	515	**0.21**	7.06e-6	3.89e-4	19.53
OptPSDP	1867	1885	0.734	9.62e-6	4.1e-3	19.68
	E10: n = 100, m = 10, problem = 3, $\gamma = 0.55$					
Grad	10000	10001	10.299	1.2e-4	5.73e-1	43.44
FGM	7108	7109	7.951	1.39e-6	6.86e-6	48.15
ParTan	626	627	**0.791**	5.21e-6	1.19e-4	43.3
OptPSDP	3817	3840	4.206	1.04e-6	9.7e-3	43.58
	E11: n = 100, m = 100, problem = 3, $\gamma = 0.55$					
Grad	9250	9251	9.625	3.14e-6	4.89e-2	1.52e-1
FGM	686	687	0.86	9.9e-6	5.56e-7	5.91e-4
ParTan	437	438	**0.7**	4.72e-6	8.88e-6	2.2e-3
OptPSDP	606	611	0.801	8.94e-6	9.07e-4	2.42e-2
	E12: n = 150, m = 150, problem = 3, $\gamma = 0.55$					
Grad	9660	9661	21.894	4.17e-6	7.5e-2	2.04e-1
FGM	729	730	1.986	9.91e-6	5.63e-7	6.2e-4
ParTan	486	487	**1.764**	2.9e-6	3.68e-6	1.3e-3
OptPSDP	649	665	1.896	8.87e-6	1e-3	2.66e-2

TABLE 3. Numerical results for ill conditioned PSDP (problem = 3).

In tables 2 and 3 we present the results obtained by the four procedures solving ill-conditioned PSDP. These tables clearly show that **Grad** algorithm is the method that obtain the worst results, because sometimes it runs the maximum number of iterations allowed and it is the slowest in terms CPU-time. On the other hand, we observe that the **FGM**, **ParTan** and **OptPSDP** methods show similar performance both in the number of iterations, and in execution time when $m = n$. However, when A is a rectangular matrix, the most efficient method is **ParTan**. In spite of this, all the methods reach convergence, since all obtain small values of *XErr*.

For the fourth experiment group, the PSDP problems were constructed with randomly generated synthetic data as explained at the beginning of this section, however, the optimum X^* matrix was built as follows, first a matrix $M \in \mathbb{R}^{n\times n}$ is randomly generated with entries following a standard normal distribution, afterwards V is obtained as the orthogonal matrix of the QR factorization of M, from this matrix, we set $X^* = V^\top \Sigma V$, where $\Sigma \in \mathbb{R}^{n\times n}$ is a diagonal matrix whose diagonal elements were generated by $\Sigma(1,1) = \Sigma(2,2) = 0$ and $\Sigma(i,i) = \texttt{rand}$ for all $i \in \{3, 4, \ldots, n\}$ using Matlab notation. Thus, the optimal solution of the PSDP generated is a symmetric and positive semi-definite matrix with only two eigenvalues equal to zero and $n-2$ strictly positive eigenvalues.

Method	Nitr	Nfe	Time	XErr	Fval	Global Error
	E13: n = 10, m = 70, problem = 1, $\gamma = 0.55$					
Grad	25	26	0.031	7.61e-6	5.02e-9	18.85
FGM	788	789	0.956	9.98e-6	5.73e-26	22.27
ParTan	12	13	0.019	4.38e-6	1.7e-10	18.91
OptPSDP	11	12	**0.009**	6.00e-6	2e-9	18.82
	E14: n = 1000, m = 1000, problem = 1, $\gamma = 0.85$					
Grad	8	9	1.868	4.57e-6	1.56e-11	4.73e-7
FGM	8	9	1.825	1.86e-6	1.51e-12	9.8e-8
ParTan	7	8	2.482	3.76e-6	8.44e-12	3.53e-7
OptPSDP	8	9	**0.935**	2.88e-6	3.09e-12	1.45e-7
	E15: n = 60, m = 30, problem = 2, $\gamma = 0.85$					
Grad	3737	3738	1.892	9.99e-6	2.97e-5	15.9
FGM	1876	1877	0.9692	9.22e-6	2.39e-6	16.4
ParTan	185	186	0.113	3.43e-6	3.6e-7	15.9
OptPSDP	178	183	**0.063**	7.49e-6	8.42e-6	15.9
	E16: n = 100, m = 100, problem = 2, $\gamma = 0.55$					
Grad	9437	9438	11.787	3.18e-6	5.76e-2	1.57e-1
FGM	642	643	0.814	9.91e-6	5.84e-7	5.78e-4
ParTan	378	379	0.601	4.2e-6	8.99e-6	1.6e-3
OptPSDP	511	536	**0.47**	8.27e-6	8.05e-4	2.13e-2
	E17: n = 60, m = 30, problem = 3, $\gamma = 0.55$					
Grad	10000	10001	5.073	6.14e-5	1.77e-1	15.8
FGM	4179	4180	2.168	9.49e-6	6.1e-5	16.3
ParTan	596	597	**0.365**	6.54e-6	3.87e-5	15.9
OptPSDP	1366	1390	0.429	8.69e-6	1.1e-3	17.8
	E18: n = 120, m = 120, problem = 3, $\gamma = 0.55$					
Grad	9281	9282	16.522	4.21e-6	8.85e-2	2.13e-1
FGM	672	673	1.218	9.91e-6	5.64e-7	6.01e-4
ParTan	393	394	0.897	3.07e-6	6.98e-6	1.4e-3
OptPSDP	557	582	**0.686**	8.29e-6	8.56e-4	2.32e-2

TABLE 4. Numerical results for several PSDP (problem = 1,2,3).

The numerical results corresponding to the third experiment are shown in Table 4. This table shows that the **ParTan** algorithm obtained the best performance in terms of the number of iterations, in almost all experiments. In addition, we observe that the our **OptPSDP** is the most efficient procedure in terms of CPU-time in both well-conditioned and ill-conditioned problems. From all the experiments performed, we concluded that the our proposal is a competitive alternative to solve the problem 1 under different situations of conditioning and scale of A.

5. Conclusions

The problem (1) has a wide range of applications in the fields of structure analysis, physical problems, signal processing, estimation of correlation matrices, among others. To address this problem, we design and implement an efficient

and globally convergent algorithm that preserves feasibility in each iteration. Our proposal is based on the gradient projection method and we incorporate a non-monotone strategy in combination with the Barzilai and Borwein step sizes in order to accelerate the convergence. The bottleneck of the proposed algorithm is the computation of the projection operator, which is computationally inefficient. In order to improve the efficiency of our algorithm, we present a strategy based on Cholesky factorization to reduce the number of projections. This technique can be a good alternative to deal with large-scale problems. Some theoretical results were presented. Finally, from the numerical experiments we note that the performance of the resulting algorithm is quite competitive with some of the state of the art methods.

Acknowledgements. This research was supported in part by Conacyt, Mexico (PhD. studies scholarship).

References

[1] Lars-Erik Andersson and Tommy Elfving, *A constrained procrustes problem*, SIAM Journal on Matrix Analysis and Applications **18** (1997), no. 1, 124–139.

[2] Negin Bagherpour and Nezam Mahdavi-Amiri, *Efficient algorithms for positive semi-definite total least squares problems, minimum rank problem and correlation matrix computation*, arXiv preprint arXiv:1407.3894 (2014).

[3] Jonathan Barzilai and Jonathan M Borwein, *Two-point step size gradient methods*, IMA journal of numerical analysis **8** (1988), no. 1, 141–148.

[4] Dimitri P Bertsekas, *Nonlinear programming*, Journal of the Operational Research Society **48** (1997), no. 3, 334–334.

[5] John E Brock, *Optimal matrices describing linear systems.*, AIAA Journal **6** (1968), no. 7, 1292–1296.

[6] Yu-Hong Dai and Roger Fletcher, *Projected barzilai-borwein methods for large-scale box-constrained quadratic programming*, Numerische Mathematik **100** (2005), no. 1, 21–47.

[7] JB Francisco, FS Viloche Bazán, and M Weber Mendonça, *Non-monotone algorithm for minimization on arbitrary domains with applications to large-scale orthogonal procrustes problem*, Applied Numerical Mathematics **112** (2017), 51–64.

[8] Nicolas Gillis and Punit Sharma, *A semi-analytical approach for the positive semidefinite procrustes problem*, Linear Algebra and its Applications **540** (2018), 112–137.

[9] Nicholas J Higham, *Computing a nearest symmetric positive semidefinite matrix*, Linear algebra and its applications **103** (1988), 103–118.

[10] Anping Liao and Zhongzhi Bai, *Least-squares solution of ax b= d over symmetric positive semidefinite matrices x*, Journal of Computational Mathematics (2003), 175–182.

[11] Yurii Nesterov, *Introductory lectures on convex programming volume i: Basic course*, Lecture notes **3** (1998), no. 4, 5.

[12] Jorge Nocedal and Stephen Wright, *Numerical optimization*, Springer Science & Business Media, 2006.

[13] Marcos Raydan, *The barzilai and borwein gradient method for the large scale unconstrained minimization problem*, SIAM Journal on Optimization **7** (1997), no. 1, 26–33.

[14] TJ Suffridge and TL Hayden, *Approximation by a hermitian positive semidefinite toeplitz matrix*, SIAM journal on matrix analysis and applications **14** (1993), no. 3, 721–734.

[15] Kim-Chuan Toh, *An inexact primal–dual path following algorithm for convex quadratic sdp*, Mathematical programming **112** (2008), no. 1, 221–254.

[16] Kim-Chuan Toh, Michael J Todd, and Reha H Tütüncü, *Sdpt3a matlab software package for semidefinite programming, version 1.3*, Optimization methods and software **11** (1999), no. 1-4, 545–581.

[17] Keith G Woodgate, *Least-squares solution of f= pg over positive semidefinite symmetric p*, Linear algebra and its applications **245** (1996), 171–190.

[18] Hongchao Zhang and William W Hager, *A nonmonotone line search technique and its application to unconstrained optimization*, SIAM journal on Optimization **14** (2004), no. 4, 1043–1056.

(Recibido en junio de 2020. Aceptado en marzo de 2021)

Matemática Aplicada
Fundação Getulio Vargas
Escola de Matemática Aplicada,
FGV\EMAp, Brazil
e-mail: harry.leon@fgv.br