Optimization Methods and Software
Vol. 00, No. 00, Month 200x, 1–24

Nonsmooth methods for large bilinear matrix inequalities: Applications to feedback control

J. B. THEVENET † and D. NOLL ‡ and P. APKARIAN †

(Received 00 Month 200x; In final form 00 Month 200x)

We present a method to solve large optimization programs with bilinear matrix inequality (BMI) constraints. Such programs arise in control applications such as system analysis, controller synthesis or filter design. Our specific point of view here is to cast BMI problems as nonconvex eigenvalue optimization programs and to use nonsmooth optimization methods suited for eigenvalue optimization to compute locally optimal solutions. Our method is based on the ϵ-subgradient prototype, suitably adapted to include non-convex problems. In each tangent step, a small size semidefinite program (SDP) is solved to compute a search direction. Our method is tested on several large scale benchmark problems in output feedback controller design.

Keywords: Bilinear matrix inequality (BMI); Semidefinite programming (SDP); Output feedback control synthesis; Nonsmooth optimization; Clarke derivative.

1 Introduction

We consider affine $\mathcal{A} : \mathbb{R}^n \to \mathbb{S}^m$ and bilinear operators $\mathcal{B} : \mathbb{R}^n \to \mathbb{S}^m$ into the space $\mathbb{S}^m$ of symmetric $m \times m$ matrices,

$$\mathcal{A}(x) = A_0 + \sum_{i=1}^{n} A_i x_i, \quad \mathcal{B}(x) = \mathcal{A}(x) + \sum_{1 \leq i < j \leq n} B_{ij} x_i x_j, \tag{1}$$

where A_i, $B_{ij} \in \mathbb{S}^m$. A bilinear matrix inequality constrained optimization program, for short a BMI program, is a nonlinear mathematical program of the form

$$\begin{array}{l} \text{minimize } c^\top x, \ x \in \mathbb{R}^n \\ \text{subject to } \mathcal{B}(x) \preceq 0. \end{array} \tag{2}$$

† ONERA, Centre d'études et de recherche de Toulouse, 2 av. Edouard Belin, 31055 Toulouse, France - emails: thevenet@cert.fr - apkarian@cert.fr - Tel: +33 5.62.25.27.84 - Fax: +33 5.62.25.27.64.
‡ Université Paul Sabatier, Institut de Mathématiques, 118, route de Narbonne, 31062 Toulouse, France - email: noll@mip.ups-tlse.fr - Tel: +33 5.61.55.86.22 - Fax: +33 5.61.55.83.85.

The constraint $\mathcal{B}(x) \preceq 0$ is called a bilinear matrix inequality, a BMI for short, while $\mathcal{A}(x) \preceq 0$ is a linear matrix inequality, an LMI for short, and $\preceq 0$ means negative semidefinite. LMI constrained optimization programs of the form (2) are also known as semidefinite programs (SDPs). SDPs have convex feasible domains, while BMI programs can be highly non-convex. Notice that (2) contains quadratically constrained quadratic programming (QCQP) and integer quadratic programming as special cases. This means that despite its seemingly special form, the BMI class is fairly large, and a general algorithmic solution strategy can hardly be expected. The problem of finding x such that $\mathcal{B}(x) \prec 0$ or $\mathcal{B}(x) \preceq 0$ is called a BMI feasibility problem. Such problems may be attacked as special BMI programs $\min\{t \in \mathbb{R} : \mathcal{B}(x) \preceq tI\}$. Here we are interested in applications of BMI programs in automatic control, in particular, for output feedback control design.

The main theme of this paper is to approach BMI programs via eigenvalue optimization. We consider the unconstrained maximum eigenvalue program

$$\text{minimize } \; \lambda_1\left(\mathcal{B}(x)\right), \; x \in \mathbb{R}^n, \tag{3}$$

and the constrained program

$$\begin{array}{l} \text{minimize } c^\top x, \; x \in \mathbb{R}^n \\ \text{subject to } \lambda_1\left(\mathcal{B}(x)\right) \leq 0, \end{array} \tag{4}$$

where $\lambda_1 : \mathbb{S}^m \to \mathbb{R}$ is the maximum eigenvalue function, and where $\mathcal{B}$ is a bilinear operator (1). Observe that (2) is equivalent to (4), while the BMI feasibility problem can be solved using (3). Note that λ_1 is a convex map on $\mathbb{S}^m$, but is nonsmooth in general, so that $f = \lambda_1 \circ \mathcal{B}$ is neither smooth nor convex. Only in the special case of an affine operator $\mathcal{A}$ is $f = \lambda_1 \circ \mathcal{A}$ convex, but nonsmooth, and has been thoroughly studied by several authors, see e.g. [6–8, 14, 15].

Throughout this work, the term solution means a locally optimal solution. Since BMI programs are non-convex, locally optimal solutions are not always globally optimal, and may fail to solve the underlying control problem. For instance, while solving a BMI feasibility program, $\min\{t \in \mathbb{R} : \mathcal{B}(x) \preceq tI\}$, me may run into a local minimum (t^*, x^*) where $t^* > 0$. In that case x^* is not a valid solution for the control problem, and the only way round may be to restart the local method at a different initial guess.

The weak convergence certificates of the local strategy are in contrast with existing global techniques for several BMI problems in control, where strong certificates are given. Unfortunately, global methods are of very limited applicability due to serious size limitations. This is not surprising, as many BMI problems in control are known to be NP-hard, [2, 12], which means that the complexity of the global algorithms grows exponentially with the problem size. In contrast, as we have observed in our numerical experiments, local optimization methods are surprisingly efficient and provide practically valid solutions.

The structure of the paper is as follows. Section 2 examines several control problems and their BMI constrained optimization formulations. In Section 3, we recall the idea of ϵ-subgradient optimization and then adapt it to solve the unconstrained nonconvex eigenvalue program (3). This is later generalized in Section 4 to solve the constrained eigenvalue program (4). Finally in Section 5, the algorithm is tested against several large-scale optimization problems in H_∞ synthesis.

Notation

M^T denotes the transpose of the matrix M and $\mathrm{Tr}\, M$ its trace. $\mathbb{S}^m$ is equipped with the euclidean scalar product $\langle X, Y\rangle = X \bullet Y = \mathrm{Tr}\,(XY)$. For symmetric matrices, $M \succ N$ means that $M - N$ is positive definite and $M \succeq N$ means that $M - N$ is positive semi-definite. The operator svec maps the set of symmetric matrices $\mathbb{S}^m$ into $\mathbb{R}^l$ where $l = n\,(n+1)/2$, as:

$$x \;=\; \mathrm{svec}\, X = \left[X_{11},\ \cdots\ ,\ X_{1n},\ X_{21},\ \cdots\ ,\ X_{2n},\ \cdots\ ,\ X_{nn}\right]^\top .$$

Its inverse, the operator smat satisfies

$$\mathrm{smat}\ \mathrm{svec}\, X = X,\ \text{ and svec smat}\, x = x.$$

2 Output feedback synthesis as a BMI program

In this section, we discuss a number of control design problems that can be cast as BMI programs.

2.1 *H_∞ synthesis*

We recall a BMI characterization of the classical output feedback synthesis problem. To this end let $P(s)$ be a linear time invariant (LTI) system with state-space equations:

$$P(s):\qquad \begin{bmatrix}\dot{x}\\ z\\ y\end{bmatrix} = \begin{bmatrix}A & B_1 & B_2\\ C_1 & D_{11} & D_{12}\\ C_2 & D_{21} & D_{22}\end{bmatrix}\begin{bmatrix}x\\ w\\ u\end{bmatrix}, \tag{5}$$

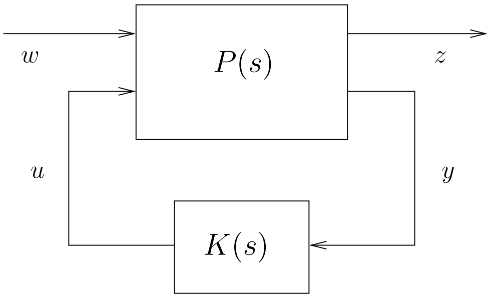
where $x \in \mathbb{R}^{n_1}$ is the state, $u \in \mathbb{R}^{m_2}$ the control, $y \in \mathbb{R}^{p_2}$ the output, $w \in \mathbb{R}^{m_1}$ the vector of exogenous inputs, $z \in \mathbb{R}^{p_1}$ the controlled or performance vector. Without loss of generality, we assume $D_{22} = 0$ throughout.

Let $T_{w,z}(K)(s)$ denote the closed-loop transfer function from w to z, depending on the unknown (to be designed) output feedback control law

$$u(s) = K(s)y(s)\,. \tag{6}$$

In the case of a static controller K, the transfer is described in state-space form by:

$$T_{w,z}(K):\quad \begin{cases} \dot{x} = (A + B_2KC_2)x + (B_1 + B_2KD_{21})w \\ z = (C_1 + D_{12}KC_2)x + (D_{11} + D_{12}KD_{21})w. \end{cases}$$



1: H_∞ synthesis interconnection.

The case of a kth-order dynamic feedback controller, $K(s) = C_K(sI - A_K)^{-1}B_K + D_K$ with $A_k \in \mathbb{R}^{k\times k}$, can be reduced to the static case by considering an augmented system, through the substitutions:

$$K \to \begin{bmatrix} A_K & B_K \\ C_K & D_K \end{bmatrix},\ A \to \begin{bmatrix} A & 0 \\ 0 & 0_k \end{bmatrix}\ B \to \begin{bmatrix} 0 & B \\ I_k & 0 \end{bmatrix},\ C \to \begin{bmatrix} 0 & I_k \\ C & 0 \end{bmatrix}. \tag{7}$$

Our aim is to compute K such that the following conditions are satisfied:

- *Internal stability:* for $w = 0$ the state vector of the closed-loop system (5) and (6) tends to zero as time goes to infinity.
- *Performance:* the H_∞ norm $\|T_{w,z}(K)\|_\infty$ is minimized among all closed-loop stabilizing K.

Finding K may be transformed into a matrix inequality using the bounded real lemma [1]. For a static K this leads to the following well-known characterization:

PROPOSITION 2.1 *A static output feedback controller K is such that $\|T_{w,z}(K)\|_\infty < \gamma$ if and only if there exists a Lyapunov matrix $P \in \mathbb{S}^{n_1}$ such that:*

$$\begin{bmatrix} (A+B_2KC_2)^T P+P(A+B_2KC_2) & P(B_1+B_2KD_{21}) & (C_1+D_{12}KC_2)^T \\ (B_1+B_2KD_{21})^T P & -\gamma I & (D_{11}+D_{12}KD_{21})^T \\ (C_1+D_{12}KC_2) & (D_{11}+D_{12}KD_{21}) & -\gamma I \end{bmatrix} \prec 0 \qquad (8)$$

$$P \succ 0. \qquad (9)$$

A similar expression for dynamic $K(s)$ is readily obtained using the substitutions (7).

Notice that (8), (9) are indeed of the form (4) if the gain γ is minimized subject to the constraint (8), as soon the strict inequality $P \succ 0$ is replaced by a non-strict inequality $P \succeq \epsilon I$ for a small threshold $\epsilon > 0$. Similarly, $\prec 0$ in (8) has to be replaced by $\preceq -\epsilon I$.

2.2 *Examples of BMI programs in control*

Many other applications in control may be turned into BMI-constrained programs. Let us discuss an application of interest, minimizing the spectral abscissa of the closed-loop system:

$$\min_K \alpha(A + B_2KC_2), \qquad (10)$$

where $\alpha : \mathbb{R}^{n\times n} \to \mathbb{R}$ is defined as

$$\alpha(A) = \max\{\mathrm{Re}\,\lambda : \lambda\,\text{eigenvalue of } A\}.$$

Notice that α is not even locally Lipschitz as a function, which renders α difficult to optimize. It is therefore interesting that an equivalent cast as a BMI optimization program can be obtained. Observe the following

LEMMA 2.2 *Let $A \in \mathbb{R}^{n\times n}$. Then $\alpha(A) \leq t$ if and only if there exists $P \in \mathbb{S}^n$, $0 \prec P \preceq I$, such that $A^TP + PA - 2tP \preceq 0$.*

This suggests that we might attack the spectral abscissa minimization program (10) by solving the following BMI:

$$\begin{array}{ll} \text{minimize} & \alpha \\ \text{subject to} & (A + B_2KC_2)^T P + P(A + B_2KC_2) - 2\alpha P \preceq 0 \\ & \vartheta I \preceq P \preceq I \end{array} \qquad (11)$$

where $0 < \vartheta \ll 1$ is some small threshold. If (11) is represented as a BMI in standard form (1), the decision variable is $x = (\gamma, K, P)$. Notice that ϑ restricts the condition number of the Lyapunov matrix P, and is therefore not known a priori. However, what can be said is that ϑ can be the same for all (γ, K, P) varying in a bounded set. Numerical experiments with program (11) will be presented in Section 5.

Following Trefethen [17], the pseudo-spectral abscissa of a matrix $A \in \mathbb{R}^{m\times m}$ is defined as

$$\alpha_\epsilon(A) \,=\, \max\left\{\mathrm{Re}\lambda : \lambda \in \Lambda_\epsilon(A)\right\},$$

where Λ_ϵ is the ϵ-pseudospectrum of A, that is, the set of all eigenvalues of matrices $A + E$, with E a perturbation of euclidean norm $\|E\| \leq \epsilon$. For $\epsilon = 0$ we recover $\alpha = \alpha_0$, the spectral abscissa, $\Lambda = \Lambda_0$ the spectrum of A. A natural extension of program (11) is the class of programs of the form

$$\min_{x\in\mathbb{R}^n} \alpha_\epsilon\left(\mathcal{A}(x)\right), \tag{12}$$

where $\mathcal{A} : \mathbb{R}^n \to \mathbb{R}^{m\times m}$ is a smooth operator. Using this function for static feedback synthesis was first proposed by Burke *et al.* in [3, 4, 10]. We have the following lemma:

LEMMA 2.3 *Let $A \in \mathbb{R}^{n\times n}$, then $\alpha_\epsilon(A) < t$ if and only if there exist $P \in \mathbb{S}^n$, $P \succ 0$, $\lambda \in \mathbb{R}$, $\mu \in \mathbb{R}$, $\mu < 0$, such that*

$$\begin{bmatrix} (\mu-\lambda)I + 2tP - A^TP - PA & -\epsilon P \\ -\epsilon P & \lambda I \end{bmatrix} \succ 0.$$

Based on this cast, program (12) may also be converted to a BMI program.

3 Unconstrained eigenvalue problem

3.1 *Steepest ϵ-enlarged descent for $f = \lambda_1 \circ \mathcal{B}$*

In this section we start with the simpler unconstrained program (3). Let us briefly explain the idea of ϵ-descent algorithms. For a convex function $f : \mathbb{R}^n \to \mathbb{R}$ and $\epsilon \geq 0$ we define according to [9] the ϵ-subdifferential of f at x as

$$\partial_\epsilon f(x) = \left\{g \in \mathbb{R}^n : g^Th \leq f(x+h) - f(x) + \epsilon \text{ for every } h \in \mathbb{R}^n\right\}.$$

Then the following simple observation can be made [9, XI, Theorem 1.1.5]. Suppose at the current x_k we have $0 \notin \partial_\epsilon f(x_k)$. Then the value of f at x_k may be decrease by at least ϵ. That is, a new iterate x_{k+1} may be found such that $f(x_{k+1}) \leq f(x_k) - \epsilon$. Conversely, if $0 \in \partial_\epsilon f(x_k)$, then the value $f(x_k)$ is ϵ-optimal, that is, f is within ϵ of its minimum value. In particular, if ϵ is sufficiently small, let's say of the size of the machine ϵ, we would certainly stop at x_k. Otherwise, we would take a step which decreases f by ϵ, and proceed.

The ϵ-subdifferential therefore seems an ideal instrument for optimization, as it gives us both a good stopping test and a means to achieve sizeable decrease of the objective function. Unfortunately, it has two drawbacks. Firstly, it is hard to compute. Secondly, it is not obviously extended to nonconvex functions. Naturally, if f is a composite function like $f = \phi \circ \mathcal{B}$ for convex ϕ and smooth $\mathcal{B}$, then we would extend the ϵ-subdifferential by setting

$$\partial_\epsilon f(x) = \mathcal{B}'(x)^\star \left[\partial_\epsilon \phi\left(\mathcal{B}(x)\right)\right],$$

using Clarke's chain rule [5]. However, $\partial_\epsilon f(x)$ so defined is no longer a global notion and its properties are weaker.

Our example of a composite function is of course the maximum eigenvalue function $f(x) = \lambda_1\left(\mathcal{B}(x)\right)$ with a bilinear operator $\mathcal{B}$, and we aim at solving program (3). In this case, we have the following well-known representation of the ϵ-subdifferential (cf. [9]):

$$\partial_\epsilon \lambda_1(X) = \{G \in \mathbb{S}^m : G \succeq 0, \mathrm{Tr}(G) = 1, \lambda_1(X) \leq G \bullet X + \epsilon\}.$$

From what we have seen, $\partial_\epsilon \lambda_1$ is hard to compute, and we need an approximation which can be computed with reasonable effort. We follow [13] and, previously, [6] and [14], where affine operators were considered, and use an approximation $\delta_\epsilon \lambda_1(X)$ of the ϵ-subdifferential $\partial_\epsilon \lambda_1(X)$ of λ_1 at $X \in \mathbb{S}^m$, called the *ϵ-enlarged subdifferential*:

$$\delta_\epsilon \lambda_1(X) = \left\{Q_\epsilon Y Q_\epsilon^T : Y \succeq 0,\ \mathrm{tr}(Y) = 1,\ Y \in \mathbb{S}^{r_\epsilon}\right\}.$$

Here r_ϵ is called the ϵ-multiplicity of $\lambda_1(X)$, where the first r_ϵ eigenvalues of $\mathcal{B}(x) \in \mathbb{S}^m$ are those which satisfy $\lambda_i > \lambda_1 - \epsilon$, and where the columns of the $r_\epsilon \times m$-matrix Q_ϵ are the associated eigenvectors. Then $\partial\lambda_1(X) \subset \delta_\epsilon\lambda_1(X) \subset \partial_\epsilon\lambda_1(X)$, and $\delta_\epsilon\lambda_1(X)$ is as we shall see a good inner approximation of $\partial_\epsilon\lambda_1(X)$. Indeed, following Oustry [14] we have the following

THEOREM 3.1 *Let $\epsilon > 0$, $\eta > 0$, $X \in \mathbb{S}^m$, and $\Delta_\epsilon(X) = \lambda_{r_\epsilon}(X) - \lambda_{r_\epsilon+1}(X) > 0$. Define*

$$\rho = \rho(\epsilon, \eta) = \left(\frac{2\eta}{\Delta_\epsilon(X)}\right)^{1/2} + \frac{2\eta}{\Delta_\epsilon(X)}. \tag{13}$$

Then

$$\partial_\eta \lambda_1(X) \subset \delta_\epsilon \lambda_1(X) + \rho B,$$

where B is the Frobenius norm unit ball in $\mathbb{S}^m$.

Using the same chain rule as before, we extend the ϵ-enlarged subdifferential to composite functions, by setting

$$\delta_\epsilon f(x) = \left\{ \mathcal{B}'(x)^\star Z : Z = Q_\epsilon Y Q_\epsilon^T,\ Y \succeq 0,\ \mathrm{tr}(Y) = 1,\ Y \in \mathbb{S}^{r_\epsilon} \right\}.$$

Then obviously

$$\partial f(x) \subset \delta_\epsilon f(x) \subset \partial_\epsilon f(x),$$

and $\delta_\epsilon f(x)$ is an inner approximation of $\partial_\epsilon f(x)$. Recall that for $\epsilon \geq 0$, the direction d of steepest ϵ-descent is obtained as

$$d = -\frac{g}{\|g\|}, \quad g = \mathrm{argmin}\left\{\|g\| : g \in \partial_\epsilon f(x)\right\}.$$

For $\epsilon = 0$ we obtain the direction of steepest descent as a special case. The following definition is therefore fairly natural:

Definition 3.1 The direction of steepest ϵ-enlarged descent d is defined as

$$d = -\frac{g}{\|g\|}, \quad g = \mathrm{argmin}\left\{\|g\| : g \in \delta_\epsilon f(x)\right\}. \tag{14}$$

Once the concept of generalized gradients is defined, it is standard to introduce an associated directional derivative. For instance, the ϵ-directional derivative of f at x in direction d is associated with $\partial_\epsilon f(x)$ and defined as

$$\begin{aligned} f'_\epsilon(x; d) &= \max\{g^\top d : g \in \partial_\epsilon f(x)\} \\ &= \max\{G \bullet D : G \in \partial_\epsilon \lambda_1(X)\}, \end{aligned} \tag{15}$$

where $D = \mathcal{B}'(x)d$ and $X = \mathcal{B}(x)$. Similarly, we introduce the following directional derivative associated with $\delta_\epsilon f(x)$.

Definition 3.2 The ϵ-enlarged directional derivative of $f = \lambda_1 \circ \mathcal{B}$ at x in direction d is

$$\begin{aligned} \tilde{f}'_\epsilon(x; d) &= \max\{g^\top d : g \in \delta_\epsilon f(x)\} \\ &= \max\{G \bullet D : G \in \delta_\epsilon \lambda_1(X)\}, \end{aligned} \tag{16}$$

where $D = \mathcal{B}'(x)d$ and $X = \mathcal{B}(x)$.

The remarkable fact, observed in [6] and [14], is that contrary to $f'_\epsilon(x;d)$, which is difficult to compute for $\epsilon > 0$, the ϵ-enlarged directional derivative can be computed efficiently.

LEMMA 3.2 *1) We have the formula*

$$\tilde{f}'_\epsilon(x;d) = \lambda_1\left(Q_\epsilon^\top D Q_\epsilon\right), \tag{17}$$

where $D = \mathcal{B}'(x)d$, and where the columns of Q_ϵ form an orthonormal basis of the invariant subspace of $X = \mathcal{B}(x)$ associated with the eigenvalues $\lambda_i(X) > \lambda_1(X) - \epsilon$.
2) If $0 \notin \delta_\epsilon f(x)$, then

$$\tilde{f}'_\epsilon(x;d) = -\mathrm{dist}\,(0, \delta_\epsilon f(x)) < 0. \tag{18}$$

Proof 1) Formula (17) follows when we substitute $G = Q_\epsilon^\top Y Q_\epsilon, Y \succeq 0, \mathrm{Tr}(Y) = 1$ in (15), because then

$$\begin{aligned}\tilde{f}'_\epsilon(x;d) &= \max\{Q_\epsilon^\top Y Q_\epsilon \bullet D : Y \succeq 0, \mathrm{Tr}(Y) = 1\}\\ &= \max\{Y \bullet Q_\epsilon D Q_\epsilon^\top : Y \succeq 0, \mathrm{Tr}(Y) = 1\}\\ &= \lambda_1\left(Q_\epsilon D Q_\epsilon^\top\right).\end{aligned}$$

2) Formula (18) follows using Fenchel duality. Indeed, if $0 \notin \delta_\epsilon f(x)$, then

$$\begin{aligned}\min_{\|d\|\le 1} \tilde{f}'_\epsilon(x;d) &= \min_{\|d\|\le 1} \max_{g\in\delta_\epsilon f(x)} g^\top d \qquad \text{(using (16))}\\ &= \max_{g\in\delta_\epsilon f(x)} \min_{\|d\|\le 1} g^\top d \qquad \text{(Fenchel duality)}\\ &= \max_{g\in\delta_\epsilon f(x)} -g^\top \frac{g}{\|g\|}\\ &= -\min_{g\in\delta_\epsilon f(x)} \|g\|\end{aligned}$$

which leads back to formula (14) and thereby proves formula (18). □

With these preparations we are now ready to use Oustry's estimate to quantify descent. First we have the following

THEOREM 3.3 *Suppose $0 \notin \delta_\epsilon f(x)$, and let $\sigma(\epsilon) := \mathrm{dist}\,(0, \delta_\epsilon f(x)) > 0$. Let $\eta(\epsilon)$*

be chosen as

$$\eta(\epsilon) = \frac{1}{8}\Delta_\epsilon(x)\left(\sqrt{1+\frac{32}{9}\frac{\sigma(\epsilon)}{\|\mathcal{B}'(x)^\star\|}}-1\right)^2 \tag{19}$$

where $\Delta_\epsilon(x) = \Delta_\epsilon(\mathcal{B}(x)) = \lambda_{r_\epsilon}(X) - \lambda_{r_\epsilon+1}(X) > 0$. *Then* $0 \notin \partial_{\eta(\epsilon)} f(x)$.

Proof Observe that the function $\eta \mapsto \rho(\epsilon,\eta)$ is monotonically increasing from 0 to $+\infty$ on $[0,\infty)$. Therefore there exists a unique $\eta(\epsilon)$ such that $\rho(\epsilon,\eta(\epsilon)) = \frac{8}{9}\sigma(\epsilon)/\|\mathcal{B}'(x)^\star\|$. This implies $0 \notin \partial f_{\eta(\epsilon)}(x)$. Indeed, suppose on the contrary that $0 \in \partial_\eta f(x)$, then $0 \in \delta_\epsilon f(x) + \rho\mathcal{B}'(x)^\star[B]$ by Theorem 3.1. That means there exists $V \in B \subset \mathbb{S}^m$ such that $\rho\mathcal{B}'(x)^\star V \in \delta_\epsilon f(x)$. Hence $\rho\|\mathcal{B}'(x)^\star V\| \geq \sigma(\epsilon)$, which in view of $\|V\| \leq 1$ contradicts the above choice $\rho\|\mathcal{B}'(x)^\star\| = (8/9)\sigma(\epsilon) < \sigma(\epsilon)$.

To conclude, it remains to observe that $\eta(\epsilon)$ above is given by formula (19), which follows with (13) by solving a quadratic equation. □

In the case of a convex composite function $f = \lambda_1 \circ \mathcal{A}$ we are now ready to quantify descent. We have the following result, which the reader might compare with Oustry [14] and [13].

PROPOSITION 3.4 *Let* $f = \lambda_1 \circ \mathcal{A}$ *be convex and suppose* $0 \notin \delta_\epsilon f(x)$. *Then there exists a step* x_+ *away from* x *such that*

$$f(x_+) \leq f(x) - \eta(\epsilon),$$

where $\eta(\epsilon)$ *is given by (19). This step is of the form* $x_+ = x + td$, *where* d *is the direction of steepest* ϵ*-enlarged descent (14), and* $t > 0$ *may be found by a finite line search.*

All the elements needed to compute $\eta(\epsilon)$ are available. Since $\mathcal{B}'(x)^\star$ is the operator $\mathbb{S}^m \to \mathbb{R}^n$ mapping $Z \mapsto (B_1(x) \bullet Z, \ldots, B_n(x) \bullet Z)$ with $B_i(x) = \frac{\partial\mathcal{B}(x)}{\partial x_i}$, the norm of $\mathcal{B}'(x)^\star$ may at least be estimated by any matrix norm of $[\mathrm{vec}B_1(x), \ldots, \mathrm{vec}B_n(x)]$.

Unfortunately, we are not done yet when $f = \lambda_1 \circ \mathcal{B}$ is non-convex, as the descent theorem of [9, XI, Theorem 1.1.5], on which Proposition 3.4 depends, is not available. Here we use the analysis obtained in [13, Lemma 8], which is lengthy and cannot be reproduced here. The result is the following

PROPOSITION 3.5 *Let* $\epsilon > 0$ *with* $0 \notin \delta_\epsilon f(x)$. *Let* d *be the direction of steepest* ϵ*-enlarged descent (14) at* x. *Then a step* $x_+ = x + td$ *away from* x *may be computed* *such that $f(x_+) \leq f(x) - \theta(x,\epsilon)$, where*

$$\theta(x,\epsilon) = \min\left\{\frac{\eta(\epsilon)}{2}, \frac{\operatorname{dist}\left(0, \delta_\epsilon f(x)\right)^2}{8L}\right\} > 0. \tag{20}$$

Here $L = |\langle d, \mathcal{B}'' d\rangle|$, d is the direction of steepest ϵ-enlarged descent (14), and $\eta(\epsilon)$ is given by formula (19). The step $t > 0$ can be computed by a finite line-search. For an affine operator $\mathcal{A}$, $L = 0$ and the right hand term in (20) is $+\infty$.

Proof For the convenience of the reader we sketch the idea of the proof, whose details are presented in [13]. As we have seen, $0 \notin \delta_\epsilon f(x)$ implies $0 \notin \partial_{\eta(\epsilon)} f(x)$ by Theorem 3.1. From the convex case we know that this implies that $t \mapsto \lambda_1\left(X + tD\right)$ decreases by at least $\eta(\epsilon)$, where $X = \mathcal{B}(x)$, $D = \mathcal{B}'(x)d$. Now in the non-convex case there is a discrepancy between $f(x+td)$ and $\lambda_1\left(X + tD\right)$, and estimating this mismatch is what takes most of the work in [13]. Roughly, what happens is the following. Suppose t_η is a step where $\lambda_1\left(X + t_\eta D\right) - f(x) \leq -\eta$. Then the situation depends on whether $f(x + t_\eta d)$ is close to $\lambda_1\left(X + t_\eta D\right)$, in which event we get the left hand branch of estimate (20), or whether the discrepancy between $f(x+td)$ and $\lambda_1\left(X + tD\right)$ at t_η is important, the case covered by the right hand branch of the formula. The minimum of both possibilities is then a pessimistic estimate of the progress achieved. □

What is remarkable here is that the order of magnitude of the descent $\theta(x,\epsilon)$ at x is the same as in the convex case, namely $\mathcal{O}\left(\eta(\epsilon)\right)$. In fact, this is obvious for the left hand term in (20). Now formula (19) shows that $\eta(\epsilon) \sim \Delta_\epsilon(X)\operatorname{dist}\left(0, \delta_\epsilon f(x)\right)^2$, if we observe that $\left(\sqrt{1+\sigma} - 1\right)^2 = \frac{\sigma^2}{4} + \mathcal{O}(\sigma^3)$ for small $\sigma > 0$. This means the left hand term is of the order $\Delta_\epsilon(X)\operatorname{dist}\left(0, \delta_\epsilon f(x)\right)^2$, while the right hand term in (20) is proportional to $\operatorname{dist}\left(0, \delta_\epsilon f(x)\right)^2$. Since $\Delta_\epsilon(X)$ is bounded by $\lambda_1(X) - \lambda_m(X)$, which is bounded on a bounded set of X, the asymptotic order of $\theta(x,\epsilon)$ is indeed $\eta(\epsilon)$.

In order to achieve maximum decrease of f at the current x, we wish to find $\epsilon > 0$ such that $\theta(x,\epsilon)$ is maximized. Let us arrange the eigenvalues of $X = \mathcal{B}(x)$ in decreasing order

$$\lambda_1(X) = \cdots = \lambda_{r_2-1}(X) > \lambda_{r_2}(X) = \cdots = \lambda_{r_3-1}(X) > \lambda_{r_3}(X)\ldots,$$

where $r_1 = 1$ and where a gap occurs between $r_i - 1$ and r_i. Let us choose ϵ_i such that $r_{\epsilon_i} = r_i - 1$. Then every ϵ_i is a candidate for the step selection. Put $d_i(x) = \operatorname{dist}\left(0, \delta_{\epsilon_i} f(x)\right)$. Then $d_1(x) \geq d_2(x) \geq \cdots \geq d_t(x) = 0$ for some $t \leq m-1$. We define

$$\begin{aligned}\theta(x) &= \max_{i=1,2,\dots} \theta(x,\epsilon_i) \qquad (21)\\ &= \max_{i=1,2,\dots} \max\left\{ \frac{\Delta_{\epsilon_i}(x)}{16}\left(\sqrt{1+\frac{32}{9}\frac{d_i(x)}{\|\mathcal{B}'(x)^\star\|}}-1\right)^2, \frac{d_i(x)^2}{8L}\right\},\end{aligned}$$

and we let $\epsilon(x)$ denote the smallest ϵ_i where this maximum is attained. As we have seen, choosing $\epsilon(x)$ gives a guaranteed decrease of $\theta(x)$. We now have the following result for program (3):

THEOREM 3.6 *Let $f = \lambda_1 \circ \mathcal{B}$ be a nonconvex maximum eigenvalue function and suppose $x_1 \in \mathbb{R}^n$ is such that $\Omega = \{x \in \mathbb{R}^n : f(x) \le f(x_1)\}$ is compact. Let x_k be a sequence of iterates such that $f(x_{k+1}) \le f(x_k) - \theta(x_k)$. Then every accumulation point x^* of x_k is a critical point, i.e., satisfies $0 \in \partial f(x^*)$.*

Proof Since the values $f(x_k)$ decrease, iterates x_k stay in Ω. Let x^* be an accumulation point of the x_k and select $\mathcal{K} \subset \mathbb{N}$ such that $x_k \to x^*$, $k \in \mathcal{K}$. Clearly $x^* \in \Omega$. By monotonicity of the sequence $f(x_k)$, we have $f(x_{k+1}) \to f(x^*)$, $k \in \mathcal{K}$, even though we do not have any information as to whether $x_{k+1} \to x^*$. Therefore $f(x_{k+1}) - f(x_k) \to 0$, $k \in \mathcal{K}$. On the other hand, by assumption, we have $f(x_{k+1}) - f(x_k) \le -\theta(x_k) < 0$, which implies $\theta(x_k) \to 0$.

Suppose now that contrary to the claim of the theorem, x^* is not a critical point. Then $0 \notin \partial f(x^*)$. Let $r_1^* = 1, r_2^*, \dots$ the indicies denoting eigenvalue gaps at $X^* = \mathcal{B}(x^*)$. Let $\epsilon_1^* > 0$ cut into the first eigenvalue gap, that is $\lambda_1^* - \epsilon_1^* > \lambda_{r_2^*}$. Then $\partial f(x^*) = \delta_{\epsilon_1^*} f(x^*)$, which means that there is at least one candidate for computing $\theta(x^*)$, namely $\theta(x^*, \epsilon_1^*) > 0$, so that $\theta(x^*) > 0$. In other words, if we follow the procedure to compute descent steps, we would find a step x^{**} away from x^* such that $f(x^{**}) - f(x^*) \le -\theta(x^*) < 0$. We will now achieve a contradiction by showing that the function $\theta(\cdot)$ is lower semi-continuous, i.e., satisfies $\liminf_{k\in\mathcal{K}} \theta(x_k) \ge \theta(x^*)$. Indeed, once this is shown, we clearly could not have $\theta(x^*) > 0$, because $\liminf_{k\in\mathcal{K}} \theta(x_k) = 0$.

Passing to a subsequence $\mathcal{K}'$ of $\mathcal{K}$, we can assume that $\theta(x_k) \to \theta^*$ converges. We have to show $\theta^* \ge \theta(x^*)$. Since each $X_k = \mathcal{B}(x_k)$ with $k \in \mathcal{K}'$ has only finitely many eigenvalue gaps, and since there are at most m places for these eigenvalue gaps, we may pass to a subsequence $\mathcal{K}''$ of $\mathcal{K}'$, where each X_k, $k \in \mathcal{K}''$, has its eigenvalue gaps in the *same* places $r_1 = 1 < r_2 < \dots$. Notice, however, that r_j are not identical with the places r_j^* where the limiting matrix X^* has its eigenvalue gaps. In fact, X^* may and will have fewer eigenvalue gaps than the X_k, but by continuity of the eigenvalues each gap r_j^* of X^* is also an eigenvalue gap of the X_k. In other words, $\{r_1^*, r_2^*, \dots\} \subset \{r_1, r_2, \dots\}$.

Now consider the computation of $\theta(x^*)$. Suppose the maximum is attained at gap j^*. Now this eigenvalue gap is also an eigenvalue gap of all the X_k, even though the number need not be the same. Suppose the number of the same gap is j for the sequence X_k, which in particular means $\lambda_{r^*_{j^*}}(X^*) = \lambda_{r_j}(X^*)$. Now the quantity $\theta(x_k, \epsilon_j)$ is considered when computing $\theta(x_k)$, where ϵ_j specifies the eigenvalue gap j at the X_k and at the same time the gap j^* at X^*. Clearly the continuity properties of $\theta(x_k, \epsilon_j)$ depend on the continuity of $d_j(x_k)$ and $\Delta_j(x_k)$.

Clearly, $\Delta_j(x_k) \to \Delta_{j^*}(x^*)$ by continuity of the eigenvalue functions. Concerning the $d_j(x_k)$, let us argue that $\liminf_{k\in\mathcal{K}''} d_j(x_k) \geq d_{j^*}(x^*)$, which hinges on some sort of set convergence of $\delta_{\epsilon_j} f(x_k)$ toward $\delta_{\epsilon_j} f(x^*)$. Let the distance $d_j(x_k)$ be attained at some $\mathcal{B}'(x_k)^\star Q_k Y_k Q_k^T$, where $Y_k \succeq 0$, $\mathrm{Tr}(Y_k) = 1$, and where the columns of Q_k are an orthonormal basis of the invariant subspace of X_k belonging to the eigenvalues up to the gap j. Passing if necessary to yet another subsequence $\mathcal{K}'''$, we may assume $Y_k \to Y^*$ with $Y^* \succeq 0$, $\mathrm{Tr}(Y^*) = 1$ and $Q_k \to Q^*$ such that the columns of Q^* are an orthonormal basis of the invariant subspace of X^* associated with the eigenvalues up to the j^*th gap of X^*. Then $\mathcal{B}'(x^*)^\star Q^* Y^* Q^{*T} \in \delta_{\epsilon_j} f(x^*)$. Notice that Q^* need not be identical with the orthonormal basis Q_{ϵ_j} chosen to define $\delta_{\epsilon_j} f(x^*)$, but the values $\mathcal{B}'(x^*)^\star Q^* Y^* Q^{*T}$ do not depend on the actual choice of this basis. It follows that $\lim_{k\in\mathcal{K}'''} d_j(x_k) \geq d_{j^*}(x^*)$.

Altogether we have shown that $\liminf_{k\in\mathcal{K}''} \theta(x_k) \geq \liminf_{k\in\mathcal{K}''} \theta(x_k, \epsilon_j) \geq \theta(x^*)$, which proves the claim. □

This result proves convergence of the following algorithm, where $0 < \tau \leq 1$ is a fixed parameter:

ϵ-enlarged descent for $f = \lambda_1 \circ \mathcal{B}$

```
1.  At the current iterate $x$ stop if $0 \in \partial f(x)$. Otherwise
2.  Compute $\theta(x)$ according to (21). Let $\epsilon_j > 0$ be where the maximum
    is attained, and let $d$ be the associated direction of steepest $\epsilon_j$-enlarged
    descent (14), obtained by solving the tangent program
        $(T) \quad \min_{\|d\|\leq 1} \lambda_1\left(Q_{\epsilon_j}^\top \mathcal{B}'(x) d Q_{\epsilon_j}\right).$
3.  Perform a line search in direction $d$ until $f(x+\beta d) \leq f(x) - \tau\theta(x,\epsilon_j)$.
4.  Put $x_+ = x + \beta d$, replace $x$ by $x_+$, and go back to step 1.
```

3.2 *Practical aspects*

Computing $\theta(x)$ at the current iterate x involves computing $d_j(x)$ for *all* the eigenvalue gaps j at $X = \mathcal{B}(x)$, which is undesirable if m is large. However, as the proof shows, we could avoid this if the position of the first eigenvalue gap $j^* = 1$ of the limiting matrix X^* was known. This is the same as to say that the multiplicity of $\lambda_1(X^*)$ is known. There are practically useful ways to estimate the multiplicity m^* of $\lambda_1(X^*)$, for instance, by saying that at iterate X_k, those m with

$$|\lambda_m - \lambda_1| \leq \max(1, |\lambda_1|)\tau$$

will coalesce with λ_1 in the limit X^*, where for instance $\tau \approx 10^{-5}$.

If we wish to compute $\theta(x)$ using (20) and (21) as planned, that is, without knowledge of t^*, we can often exploit the fact that $d_1(x) \geq d_2(x) \geq \cdots \geq d_s(x) = 0$. As long as the right hand term in (21) is dominant, things are easy, because of this monotonicity. When the left hand term in $\theta(x,\epsilon)$ is dominant, we may still exploit monotonicity of the d_j. For instance, if we have computed $\theta(x_k, \epsilon_j) \sim \Delta_j(x_k) d_j(x_k)^2$ for some gap j, we can dispense with all the following $\theta(x_k, \epsilon_{j+\nu}) \sim \Delta_{j+\nu}(x_k) d_{j+\nu}(x_k)^2$, where $\Delta_{j+\nu}(x_k)$ is smaller than $\Delta_j(x_k)$. As a rule, we observe that it is usually sufficient to consider only the initial part of the spectrum, which is of course of the essence if the BMI size m is large. As our testing shows, in order to compute $\theta(x_k)$ reliably, it is very often sufficient to locate the first relatively large eigenvalue gap $\Delta_j(x_k)$.

The following table illustrates this idea, with an example of a stabilization problem whose resulting BMI is of size $m = 28$. Since all the entries we are interested in are computed at a given x_k, the dependence on x_k has been omitted below. Several values of r_ϵ, given in the first column, corresponding to sizeable gaps, are picked. The associated values of $\epsilon = \lambda_1 - \lambda_{r_\epsilon}$ and $\Delta_\epsilon = \lambda_{r_\epsilon} - \lambda_{r_\epsilon+1}$ are reported in the second and third column. For each value of r_ϵ, an SDP program is solved to compute the steepest ϵ- enlarged descent direction d, and the related distance to the enlarged ϵ -subdifferential at x_k, which is equal to $-\tilde{f}'_\epsilon(x_k; d)$ (see 18), displayed in column four. Finally, in column 5, a line search along the direction d of golden section type, with parabolic interpolation, provides the actual achieved progress on the maximum eigenvalue function λ_1. We have here that $\lambda_1(\mathcal{B}(x_k)) = 0$.

This example indicates that only the first gap, $\Delta_\epsilon = 1.9$, should be considered, as the associated achieved progress ($= 1.14$) is best. Indeed the absolute value of $\tilde{f}'_\epsilon(x_k; d)$ decreases quickly to 0 as ϵ increases, which seems to confirm our assumption. Indeed, the next meaningful gap appears for $r_\epsilon = 27$, with corresponding $\tilde{f}'_\epsilon(x_k) = 0$, and does not allow any further progress. We have observed that this situation is typical.

3.3 *Reducing the number of variables*

The appealing feature of our algorithm is that the tangent program (T) is a semidefinite program with a matrix $Q_\epsilon^\top \left[\mathcal{B}'(x)d\right] Q_\epsilon$ of size r_ϵ, which is usually small. In

r_ϵ	ϵ	Δ_ϵ	$\widetilde{f}'_\epsilon(x_k;d)$	Achieved progress
1	–	1.9	-1.21	1.14
2	1.9	1.4e-4	$-7.7e$-1	3e-1
18	1.9	3.0e-1	$-7.3e$-2	6.4e-3
19	2.2	3.7e-1	$-7.3e$-2	6.4e-3
20	2.6	3.0e-6	0	–
21	2.9	6.5e-3	0	–
27	2.9	1.9	0	–
28	4.8	–	0	–

1: influence of the gap

order to solve it efficiently, the number of variables $d \in \mathbb{R}^n$ has to be reduced, and this is done as follows. Let us first rewrite program (T) as

$$\min_{\|d\|\leq 1} \lambda_1 \left(\sum_{i=1}^{n} C_i d_i \right), \tag{22}$$

where $C_i = Q_\epsilon^\top \frac{\partial \mathcal{B}}{\partial x_i}(x) Q_\epsilon$ for $i = 1, \cdots, n$, and with $d = [d_1 \cdots d_n]^\top$. Then, by denoting $C_\sharp = [\text{svec}\,(C_1) \cdots \text{svec}\,(C_n)]$, we have

$$\sum_{i=1}^{n} C_i d_i = \text{smat}\ [C_\sharp\, d]\,. \tag{23}$$

Now observe that $C_\sharp$ has $n_\epsilon = r_\epsilon * (r_\epsilon + 1)/2$ rows and n columns, so that in general rank$\,(C_\sharp)\ \leq n_\epsilon \ll n$. We decompose d as

$$d = Q\widetilde{d} + Z\widehat{d}, \tag{24}$$

where the columns of Q form an orthonormal basis of Range$\,(C_\sharp^\top)$, while the columns of Z form an orthonormal basis of Null$\,(C_\sharp)$. The matrix Q is found by a QR-factorization: $C_\sharp^\top = Q\,R$. In particular, R is a small invertible triangular matrix and Q satisfies $Q^\top Q = I$. The decomposition (24) guarantees that $C_\sharp Z = 0$. We now have that

$$C_\sharp d \;=\; C_\sharp\, Q\widetilde{d} \;=\; R^\top \widetilde{d}.$$

The equivalent form of program (22), in the new variables $\widetilde{d}$ and $\widehat{d}$, follows from (23) and (24):

$$\min_{\|\widetilde{d}\|^2+\|\widehat{d}\|^2\leq 1} \lambda_1\left(\operatorname{smat}\left[R^\top \widetilde{d}\right]\right). \tag{25}$$

Notice that $\|d\|^2 = \|\widetilde{d}\|^2 + \|\widehat{d}\|^2$, by virtue of the definition of Q and Z. Obviously $\widehat{d}$ can be chosen arbitrarily (equal to zero), as it has no influence on the objective function. We get the following new equivalent form of the tangent program:

$$\min_{\|\widetilde{d}\|\leq 1} \lambda_1\left(\operatorname{smat}\left[R^\top \widetilde{d}\right]\right). \tag{26}$$

Finally, (26) is equivalent to the linear SDP

$$\begin{aligned}&\text{minimize } t\\ &\text{subject to } \sum_{i=1}^{r} \widetilde{C_i}\widetilde{d_i} \preceq tI\\ &\qquad\qquad \|\widetilde{d}\| \leq 1,\end{aligned} \tag{27}$$

where

$$R^\top = \left[\operatorname{svec}(\widetilde{C_1}) \cdots \operatorname{svec}(\widetilde{C_r})\right],\ \widetilde{d} = \left[\widetilde{d_1} \cdots \widetilde{d_r}\right],\ r = \operatorname{rank}(C_\sharp).$$

Let $\left(\widetilde{d^*}, t^*\right)$ denote the optimal solution of (27), then the descent direction for the improvement function is simply obtained through

$$d^* = Q\widetilde{d^*}. \tag{28}$$

The number of variables of the resulting LMI program has been reduced to at most n_ϵ. This point is at the core of our strategy since this program has to be solved at each step of the method. And now that we have come to a small-sized convex program, it is particularly easy and efficient to handle via a LMI solver. We used our software *spec*SDP (see [16]) for the numerical examples examined in Section 5. This approach allows to treat problems of a priori any size, as soon as en efficient way to compute *all* the eigenvalues and eigenvectors of the original problem is available. Notice that for large-scale problems, say several thousands of variables, some savings need to be obtained by computing only eigenvectors associated with few of the largest eigenvalues. We finally point out that the program has always a feasible solution, because the null vector is admissible in (27).

4 Constrained eigenvalue program

In this section, we consider the constrained eigenvalue program (4). A first option is to transfer this to an unconstrained program (3) using exact penalization. Choosing a penalty parameter $p > 0$ leads to

$$\min_{x \in \mathbb{R}^n} c^\top x + p \max \left\{ \lambda_1 \left(\mathcal{B}(x) \right), 0 \right\},$$

which is of the form (3) with $f = \lambda_1 \circ \widetilde{\mathcal{B}}$, and $\widetilde{\mathcal{B}}$ the augmented bilinear operator

$$\widetilde{\mathcal{B}}(x) = \begin{bmatrix} p\mathcal{B}(x) + \left(c^\top x\right) I & 0 \\ 0 & c^\top x \end{bmatrix} \in \mathbb{S}^{m+1}.$$

Since tuning the penalty parameter p may be cumbersome, we consider an alternative way to handle program (4), which uses a successive minimization of a progress function, incorporating both feasibility and optimality. At the current iterate x_k, define the progress measure

$$\phi(x, x_k) = \lambda_1 \begin{bmatrix} c^\top (x - x_k) & 0 \\ 0 & \mathcal{B}(x) \end{bmatrix} = \max\{c^\top (x - x_k), f(x)\}, \text{ with } f = \lambda_1 \circ \mathcal{B}. \quad (29)$$

The idea is now to compute at the current x_k a descent step for this function, using the theory of enlarged ϵ-subgradients developed for program (3). This requires the same ingredients, choosing ϵ, computing the direction of steepest ϵ-enlarged descent, followed by a line search along this direction to locate a new iterate x_{k+1}. Since $\phi_k(x) := \phi(x, x_k)$ is again a bilinear maximum eigenvalue function, we can apply the tools previously developed almost without change. This leads to the following program:

Steepest ϵ-enlarged descent for (4)

1. Given iterate x_k, stop if $0 \in \partial \phi_k(x_k)$, because x_k is a F. John critical point of (4). Otherwise compute $\theta(x_k) > 0$ as in (21) and pick $\epsilon > 0$ with $\theta(x_k) = \theta(x_k, \epsilon)$.
2. Given $\epsilon > 0$, compute the solution $(\widetilde{d}, t)$ of program (27). Obtain, via (28), the direction of steepest ϵ-enlarged descent for ϕ_k at x_k, in the original variables d.
3. Do a line search in direction d and obtain $x_{k+1} = x_k + td$, such that $\phi(x_{k+1}, x_k) < \phi(x_k, x_k) - \theta(x_k)$. Increase counter $k = k + 1$, and go back to step 1.

Convergence for this algorithm has already been proved in [13] and is clarified in the next theorem. We have the following

THEOREM 4.1 *Suppose $c^\top x$ is bounded below on $\Omega = \{x \in \mathbb{R}^n : f(x) \leq 0\}$, where $f = \lambda_1 \circ \mathcal{B}$. Suppose Ω has nonempty interior. Let x_k be a sequence of iterates such that $\phi(x_{k+1}, x_k) < \phi(x_k, x_k) - \theta(x_k)$, where $\theta(x_k)$ is as in (21), but computed with respect to the function ϕ_k. Then the alternatives hold:*

(i) Either *the x_k stay infeasible all the time, that is $f(x_k) > 0$. Then every accumulation point of the sequence x_k is a critical point of f.*

(ii) Or *the sequence x_k becomes feasible at some stage k_0. Then, it stays feasible, and every accumulation point of x_k is a KKT-point of program (4).*

Proof Suppose $f(x_k) > 0$. Then the term $c^\top(x - x_k)$ is inactive at x_k, and $\delta_\epsilon \phi_k(x_k)$ coincides with $\delta_\epsilon f(x_k)$. That means the method does exactly what it did in the unconstrained case, namely, reducing the value of f. If the x_k stay infeasible all the time, we are back in the situation of Theorem 3.6, where f alone is minimized.

Suppose now that some x_k becomes feasible. Then the term $c^\top(x - x_k)$ is active at x_k, which means that this term is reduced at the next step. $c^\top(x_{k+1} - x_k) < 0 = c^\top(x_k - x_k)$. Also, if $f(x_k) = 0$, then this term is active, too, which means that $f(x_{k+1}) < 0$, so that x_{k+1} is strictly feasible. From that moment on, iterates stay strictly feasible, and the objective function is reduced at each step. Every accumulation point x^* of the sequence x_k is therefore feasible. Moreover, the argument applied in the proof of Theorem 3.6 shows that $0 \in \partial\phi(x^*, x^*)$, which implies that x^* satisfied the F. John optimality conditions: There exists $(\sigma, \tau) \neq (0, 0)$ such that $0 = \sigma c + \tau g$ for some $g \in \partial f(x^*)$. □

5 Numerical examples

5.1 *Preliminary comments*

- The bundle-type method that we present here can be used for large-scale SDP problems. Here the matrix involved in the computation of the search direction is simply obtained by projection

$$\tilde{f}'_\epsilon(x; d) = \lambda_1 \left(Q_\epsilon^T \left[\sum_{i=1}^n A_i d_i \right] Q_\epsilon \right) = \lambda_1 \left(\sum_{i=1}^n Q_\epsilon^T A_i Q_\epsilon d_i \right).$$

  On the other hand the approach may in principle be extended to any nonlinear SDP, provided $\mathcal{B}'(x)d$ is cheaply computable.
- The initial iterate x_0 is in general infeasible (Bounded Real Lemma not satisfied and/or instable system). The algorithm may therefore converge to an infeasible

local minimum. However, when prior knowledge on the system is available, it is usually possible to compute a feasible starting point x_0. Then all subsequent iterates are feasible (see [13]). Moreover the H_∞-norm is decreased from the first iterate onwards.
To illustrate this behavior, we consider the example of minimizing the H_∞ norm of a given system when a stabilizing controller K_0 is already known. Then it is easy to compute the H_∞ norm γ of the resulting closed-loop system. Now it remains to provide a Lyapunov matrix P_0 such that the matrix inequality in the Bounded Real Lemma (2.1) is negative definite, for K_0 and $\gamma_0 > \gamma$ fixed. This is done by solving an LMI-feasibility problem, which can be handled by *spec*SDP if the size is reasonable. If not, the bundle method itself can be used. A third possibility is to solve a perturbed Riccati equation:

$$\tilde{A}^T P\, E + E^T P\, \tilde{A} - \left(E^T P\, \tilde{B} + S\right) R^{-1} \left(\tilde{B}^T P\, E + S^T\right) + Q = 0,$$

with

$$\tilde{A} = A + B_2 K C_2, \tag{30}$$
$$\tilde{B} = B_1 + B_2 K D_{21}, \tag{31}$$
$$E = \gamma_0\, I, \tag{32}$$
$$S = (C_1 + D_{12} K C_2)^T\, (D_{11} + D_{12} K D_{21}), \tag{33}$$
$$R = (D_{11} + D_{12} K D_{21})^T\, (D_{11} + D_{12} K D_{21}) - \gamma_0^2\, I, \tag{34}$$
$$Q = (C_1 + D_{12} K C_2)^T\, (C_1 + D_{12} K C_2) + \delta\, I, \tag{35}$$

where δ is a small positive perturbation. It is straightforward to show through Schur complement that a solution P_0 of the previous equation satisfies the strict Bounded Real Lemma 2.1.

Once the initial point has been computed, the H_∞ performance will be enhanced at every step of the algorithm. Unfortunately, these approaches tend to produce initial points which lie on the boundary of the feasible set. This may be inconvenient for large systems, as it gives a high multiplicity for $\lambda_1(x_0)$.

- The most important feature of our approach concerns the dramatic reduction of size when computing the search direction d. The size of the modified problem becomes roughly $\frac{r_\epsilon^2}{2}$, where r_ϵ is the $\epsilon-$ multiplicity of the maximum eigenvalue. So the motivation for the algorithm is to identify a set of "active" eigenvalues, which is hopefully small, and to assume that only these largest eigenvalues play a role towards optimality. The main limitation is when the multiplicity of $\lambda_1(x_k)$ is high, the size of the problem cannot be reduced and the search direction is no longer computable. For this reason it is preferable to provide an initial point that spreads out the eigenvalues of $\mathcal{B}$.

- Notice that if for a given ϵ, the computed search direction does not lead to a significant progress, for instance in the sense that the subsequent line search provided $\tau \ll 1$ in step 4. of the algorithm. In that case it appeared useful to consider $\epsilon^{'} < \epsilon$ and to restart the computation of d, as the progress expectation from the choice ϵ was probably too large. Naturally, this can also be done at the next iterate.

5.2 *Static output feedback stabilization*

We first consider examples of static output feedback stabilization. As mentioned in Section 2, we minimize the (robust) spectral abscissa of the closed-loop system. This can be cast as a BMI problem:

$$\begin{cases} \text{minimize} & \alpha \\ \text{subject to} & (A + B_2KC_2)^T P + P(A + B_2KC_2) - 2\alpha P \preceq 0. \\ & I \succeq P \succeq \vartheta I \end{cases}$$

where $\vartheta > 0$ is some small fixed threshold, and where $x = (\alpha, K, P)$. We consider the associated improvement function, defined at step k as

$$\phi(\alpha, K, P; \alpha_k, K_k, P_k) = \lambda_1 \begin{pmatrix} \alpha - \alpha_k & 0 & 0 & 0 \\ 0 & (A+B_2KC_2)^T P + P(A+B_2KC_2) - 2\alpha P & 0 & 0 \\ 0 & 0 & \vartheta I - P & 0 \\ 0 & 0 & 0 & P - I \end{pmatrix}.$$

The YALMIP interface (see http://control.ee.ethz.ch/ joloef/yalmip.php) was used to build an internal representation of the problems examined below and to get the canonical BMI expressions as in (1) before calling the solver. All these examples were borrowed from the COMPl$_\epsilon$ib collection, see [11].

Table 2 summarizes the results obtained with our algorithm, where n_s, n_K and n_V stand for the number of states, the number of entries in the controller matrix, and the resulting number of decision variables in the BMI ($n_V = n_K + \frac{n_s(n_s+1)}{2} + 1$). The size m of the BMI matrix has been added in the fifth column in order to emphasize the level of difficulty of the resolution. CPU is in seconds, α_{OL} and α_{CL} denote the spectral abscissas of the open-loop and the closed-loop systems. Note that all systems were initially instable.

These results show the efficiency of our algorithm on 8 dynamical systems out of 9, which were stabilized successfully. In particular the last example involves 324 state variables, which gives rise to a BMI-constrained problem with 52655 variables. Our algorithm is at its best for medium size problems, but could in principle attack applications with several thousands of states. Very often, when we run into numerical difficulties with very large problems, this was due to the use of the platform

Problems	n_s	n_K	n_V	m	CPU (s)	α_{OL}	α_{CL}
CM1_IS	20	2	213	61	2	$5e$-3	$-4e$-3
AC10	55	4	1545	166	1.6	0.1	-0.02
CM2_IS	60	2	1833	181	-	$5e$-3	-
CM3_IS	120	2	7263	361	34	$5e$-3	-0.01
HF2D_CD1_M256	256	4	32901	769	33	1.1	-0.54
HF2D_CD2_M256	256	4	32901	769	48	0.7	-0.46
HF2D_IS3_M256	256	4	32901	769	26	1.3	-0.07
HF2D_IS4_M256	256	4	32901	769	23	0.5	-0.04
HF2D_CD3_M324	324	4	52655	973	47	2.0	-0.60

2: Large-scale static output feedback stabilization

YALMIP, which struggled with huge BMIs due to the limited amount of available RAM on our Bi-processor Sun Blade 1500. Similarly we used the Lapack libraries to compute the spectrum of the BMI and it is not sure that the computational time, as well as the required accuracy, remain acceptable for a (dense) matrix of size larger than $1e3$. On the modified cable-mass model CM2_IS the algorithm got stuck on an infeasible point. It was actually still progressing, but the maximum number of iterations (40) had been reached without convergence and the progress was very slow. This is a typical case as described in the previous section, where the multiplicity of the maximum eigenvalue gets to large. Practically the maximum number of selected eigenvalues has to be bounded (25 in our testing), so that the "tangent" linear SDP is solved fast and reliably enough.
As far as the computational time is concerned, we see that it globally increases with the size of the problems, because of the line search and the associated spectral computations. It still remains reasonable however for these hard problems.

5.3 *Static output feedback H_∞ synthesis*

The last numerical results that we present below concern static output feedback H_∞ synthesis presented in Proposition 2.1. We consider the same examples as in the previous paragraph. Column $||H||_\infty$ in table 3 now gives the performance index reached with our algorithm, while the last column gives the pointer to the optimal controllers given later in this section.

These results in table 3 are of interest, because most of the large-scale problems presented have to our knowledge not been solved as yet.

We observed that using a "feasible" initial point (see paragraph 5.1) is not always advisable, unless a good approximation of the optimal Lyapunov matrix is also available. We found that it is often better to use randomly generated initial points. This feature may be related to the nature of our control problems, and should not be inter-

Problems	n_s	n_K	n_V	m	$\|H\|_\infty$	name
CM1_IS	20	2	213	45	1.33	K_1
AC10	55	4	1545	119	–	K_2
CM2_IS	60	2	1833	125	3.60	K_3
CM3_IS	120	2	7263	245	–	K_4
HF2D_CD1_M256	256	4	32901	1027	12.8	K_5
HF2D_CD2_M256	256	4	32901	1027	19.5	K_6
HF2D_IS3_M256	256	4	32901	1027	11.2	K_7
HF2D_IS4_M256	256	4	32901	1027	14.8	K_8
HF2D_CD3_M324	324	4	52655	1299	23.7	K_9

3: Large-scale static output feedback H_∞ synthesis

preted as a general verdict for other classes of BMI-programs.

Interestingly, we were able to compute a suitable controller for problem CM2_IS, while we failed to accomplish this through the minimization of the spectral abscissa. As for examples CM3_IS and AC10, the algorithm did not succeed in computing a stabilizing controller within the maximum allowed number of iterations, which implies that the H_∞ norm of the closed-loop transfer functions remains infinite. It is likely that, at least for problem AC10, the "size" of the stability domain obtained with a static controller is very limited.

We finally give the computed controllers associated with each plant (See the last column of the table). The controllers corresponding to plants AC10 and CM3_IS are those obtained with the spectral minimization approach, since the minimization of the H_∞ norm failed in these cases.

$$K_1 = \begin{bmatrix} -0.160 & -1.027 \end{bmatrix}, \quad K_2 = \begin{bmatrix} 0 & 3e\text{-}5 \\ 8e\text{-}5 & 6e\text{-}5 \end{bmatrix}, \quad K_3 = \begin{bmatrix} -1.690 & -2.719 \end{bmatrix},$$

$$K_4 = \begin{bmatrix} 0.210 & -11.21 \end{bmatrix}, \quad K_5 = \begin{bmatrix} -0.201 & -0.198 \\ -0.228 & -0.223 \end{bmatrix}, \quad K_6 = \begin{bmatrix} -0.141 & -0.556 \\ -0.020 & 0.458 \end{bmatrix},$$

$$K_7 = \begin{bmatrix} -0.166 & -0.184 \\ -0.628 & -0.455 \end{bmatrix}, \quad K_8 = \begin{bmatrix} -0.078 & -0.083 \\ -0.229 & -0.192 \end{bmatrix} \quad K_9 = \begin{bmatrix} -0.540 & -1.847 \\ -0.255 & -0.702 \end{bmatrix}.$$

6 CONCLUSION

In this paper, we have developed a specialized nonsmooth technique to solve large BMI programs. Both feasibility and linear objective minimization problems are considered. The core of our approach consists in the computation of the epsilon-enlarged steepest descent directions along which a line search is performed to approximate local solutions of BMI programs.

The main thrust of our approach is that inner steps reduce to small LMI problems easily handled by currently available solvers such as specSDP. Also, progress of the cost function is ensured as soon as feasibility of BMI constraints has been achieved. Therefore, our method may as well be used to refine a given controller obtained by any other method.

A number of implementation details to improve our algorithm have been discussed especially regarding initialization difficulties. Practical validity and efficiency of our method has been demonstrated through a number of H-infinity design problems from the compleib collection. Our testing indicates that systems with several hundred of states can be handled with reasonable cputimes.

References

[1] B. Anderson and S. Vongpanitlerd,*Network analysis and synthesis: a modern systems theory approach*, Prentice-Hall, 1973.

[2] V. Blondel and M. Gevers, *Simultaneous stabilizability question of three linear systems is rationally undecidable*, Mathematics of Control, Signals, and Systems, 6 (1994), pp. 135–145.

[3] J. Burke, A. Lewis, and M. Overton, *Two numerical methods for optimizing matrix stability*, Linear Algebra and its Applications 351-352, (2002), pp. 147–184.

[4] ———, *A robust gradient sampling algorithm for nonsmooth, nonconvex optimization*, submitted to SIAM J. Optimization , (2003).

[5] F. H. Clarke, *Optimization and nonsmooth analysis*, Canadian Math. Soc. Series, John Wiley & Sons, New York, 1983.

[6] J. Cullum, W. Donath, and P. Wolfe, *The minimization of certain nondifferentiable sums of eigenvalues of symmetric matrices*, Math. Programming Stud., 3 (1975), pp. 35–55.

[7] R. Fletcher, *Semidefinite matrix constraints in optimization*, SIAM J. Control and Opt., 23 (1985), pp. 493–513.

[8] C. Helmberg and F. Rendl, *Spectral bundle method for semidefinite programming*, SIAM J. Optimization, 10 (2000), pp. 673 – 696.

[9] J.-B. Hiriart-Urruty and C. Lemaréchal, *Convex Analysis and Minimization Algorithms II: Advanced Theory and Bundle Methods*, vol. 306 of Grundlehren der mathematischen Wissenschaften, Springer-Verlag, New York, 1993.

[10] A. L. J.V. Burke and M. Overton, *Optimizing matrix stability*, in Proceedings of the American Mathematical Society, vol. 129, 2001, pp. 1635–1642.

[11] F. Leibfritz, *COMPL$_e$IB, COnstraint Matrix-optimization Problem LIbrary - a collection of test examples for nonlinear semidefinite programs, control system design and related problems*, tech. rep., Universität Trier, 2003.

[12] A. Nemirovskii, *Several NP-hard problems arising in robust stability analysis*, Mathematics of Control, Signals, and Systems, 6 (1994), pp. 99–105.

[13] D. Noll and P. Apkarian, *Spectral bundle methods for nonconvex maximum eigenvalue functions: first-order methods*, Mathematical Programming Series B, (2005).

[14] F. Oustry, *A second-order bundle method to minimize the maximum eigenvalue function*, Math. Programming Series A, 89 (2000), pp. 1–33.

[15] M. Overton, *On minimizing the maximum eigenvalue of a symmetric matrix*, SIAM J. on Matrix Analysis and Applications, 9 (1988), pp. 256–268.

[16] J. B. Thevenet, D. Noll, and P. Apkarian, *Nonlinear spectral SDP method for BMI-constrained problems: Applications to control design*, in Informatics in Control, Automation and Robotics, Kluwer Academic Publishers, 2005.

[17] L. Trefethen, *Pseudospectra of linear operators*, SIAM Review, 39 (1997), pp. 383 – 406.