A Newton–Like Method for Solving Rank Constrained Linear Matrix Inequalities

Robert Orsi, Uwe Helmke and John B. Moore

Abstract— **This paper presents a Newton–like algorithm for solving systems of rank constrained linear matrix inequalities. Though local quadratic convergence of the algorithm is not a priori guaranteed or observed in all cases, numerical experiments, including application to an output feedback stabilization problem, show the effectiveness of the algorithm.**

I. Introduction

The *linear matrix inequality* (LMI) problem is a well known type of convex feasibility problem that has found many applications to controller analysis and design. The *rank constrained LMI* problem is a natural as well as important generalization of this problem. It is a non-convex feasibility problem defined by LMI constraints together with an additional matrix rank constraint.

Interest in rank constrained LMIs arises as many important output feedback and robust control problems, that cannot always be addressed in the standard LMI framework, can be formulated as special cases of this problem [1], [2], [3], [4]. Examples include *bilinear matrix inequality* (BMI) problems, see [3] and [4], that are easily seen to be equivalent to rank one constrained LMI problems.

In addition to their importance for control, rank constrained LMI problems also appear naturally in mathematical programming and combinatorial optimization tasks: all optimization problems with polynomial objective and polynomial constraints can be reformulated as LMI optimization problems with a rank one constraint [5], [6].

In general, if the set of points that satisfy an LMI is non-empty, then a numerical solution to the LMI problem can be efficiently found using well developed interior point algorithms, see for example [7]. Lack of convexity makes the rank constrained LMI problem much harder to solve. Currently available algorithms for the rank constrained LMI problem are heuristic in nature and are not guaranteed to converge to a solution even if one exists. Solution methods for this problem, or certain specializations of the problem, include those based on modified interior point methods [8], [9]; linearization [10], [11], [12]; alternating projections [13], [14], [15]; trace minimization methods that try to solve the problem by solving a related convex problem [16], [17]; augmented Lagrangian methods [18], [19]; and sequential semidefinite programming [20]. Aside from [20], these methods do not have established superlinear convergence rates and the challenge remains to find numerical schemes with verifiable local quadratic convergence rates.

In this paper we present a new heuristic method for solving the rank constrained LMI problem. The method is closely related to existing alternating projection methods but is expected to have improved convergence properties due to a built-in Newton-type step. In [13] and [15] alternating projection algorithms are proposed that involve tangent-like ideas, similar to our approach. However, the implementation details are different and the connection to the Newton method is not mentioned, nor obvious. In fact, it is this established connection to Newton's method that distinguishes our approach to earlier ones.

Our method is based on the "tangent and lift" methodology [21], a generalization of Newton's method. While the classical Newton algorithm can be used to find zeroes of functions, the tangent and lift method is more general and can be used to find a point in the intersection of an affine subspace and a manifold. We show that the rank constrained LMI problem can be formulated as a problem of finding a point in the intersection of an affine subspace and another set which, though not a manifold, is a union of manifolds. Part of the contribution of this paper is a demonstration that tangent and lift methods can be extended to this more general setting and we present an algorithm for solving the rank constrained LMI problem based on such an extension. Numerical simulations show the effectiveness of this approach.

Since our method is based on a generalization of the Newton method, local quadratic convergence is expected. However, complications arise due to the non-smoothness of the constraints. This makes a rigorous convergence theory difficult to develop and in fact, as some of our simulations show, local quadratic convergence cannot be expected in all cases. The challenge therefore is to single out a class of problems for which local quadratic convergence can be rigorously established. This will be done in a future paper.

The rest of the paper is structured as follows. Section II contains a statement of the rank constrained LMI problem and a reformulation of this problem into an equivalent form. Section III contains a discussion of the tangent and lift method and details of how we extend this methodology so that it can be applied to the rank constrained LMI problem. Section IV discusses important geometric properties of rank constrained positive semidefinite matrices. Our algorithm

R. Orsi is with The Australian National University, Canberra ACT 0200, Australia. robert.orsi@anu.edu.au

U. Helmke is with the Department of Mathematics, University of Würzburg, AM Hubland 97074 Würzburg, Germany. helmke@mathematik.uni-wuerzburg.de

J. B. Moore is with The Australian National University, Canberra ACT 0200, Australia, and National ICT Australia Limited, Locked Bag 8001, Canberra ACT 2601, Australia. john.moore@anu.edu.au

for solving the rank constrained LMI problem is given in Section V. Section VI reports on some numerical experiments and includes an application of the algorithm to an output feedback problem. The paper ends with some concluding remarks.

II. Problem Formulation

Let $\mathbb{R}$ denote the set of real numbers and $\mathcal{S}^n$ denote the set of real symmetric $n \times n$ matrices. For $A \in \mathcal{S}^n$, let $A \geq 0$ denote the property that A is positive semidefinite. The rank constrained LMI problem is the following:

Problem 1: Find $x \in \mathbb{R}^m$ such that

$$F(x) := F_0 + \sum_{i=1}^{m} x_i F_i \geq 0, \tag{1}$$

$$G(x) := G_0 + \sum_{i=1}^{m} x_i G_i \geq 0, \tag{2}$$

$$\text{rank } G(x) \leq r. \tag{3}$$

The problem data are the real symmetric matrices $F_i \in \mathcal{S}^{n_F}$ and $G_i \in \mathcal{S}^{n_G}$, and the rank bound r, which is assumed to be less than or equal to n_G.

Problem 1 consists of two LMI constraints, (1) and (2), and a rank constraint, (3). When $r = n_G$ constraint (3) is always satisfied and the problem reduces to a standard LMI feasibility problem. The more interesting case is when $r < n_G$. In this case the problem is non-convex.

Let

$$\mathcal{S}^n_+ = \{X \in \mathcal{S}^n \mid X \geq 0\}$$

and, for each integer s, let

$$\mathcal{S}^n_+(s) = \{X \in \mathcal{S}^n \mid X \geq 0,\ \text{rank}(X) = s\}.$$

Define

$$\begin{aligned} \mathcal{M}_r &= \mathcal{S}^{n_F}_+ \times \bigcup_{s=0}^{r} \mathcal{S}^{n_G}_+(s) \\ &= \{(X,Y) \in \mathcal{S}^{n_F} \times \mathcal{S}^{n_G} \mid \\ &\qquad X \geq 0,\ Y \geq 0,\ \text{rank}(Y) \leq r\} \end{aligned} \tag{4}$$

and

$$\begin{aligned} \mathcal{L} = \{(X,Y) \in \mathcal{S}^{n_F} \times \mathcal{S}^{n_G} \mid \\ (X,Y) = (F(x), G(x)) \text{ for some } x \in \mathbb{R}^m\}. \end{aligned}$$

Problem 1 can be stated in the following equivalent form.

Problem 2:

$$\text{Find } (X,Y) \in \mathcal{M}_r \cap \mathcal{L}.$$

We will see that, for each s, $\mathcal{S}^n_+(s)$ is a manifold and hence that the rank constrained LMI problem is equivalent to finding a point in the intersection of an affine subspace and another set which is a union of manifolds. This structure will enable us to use the tangent and lift ideas that are discussed in the next section.

III. Tangent and Lift

In this section we discuss the tangent and lift methodology and present an extension that can be applied to the rank constrained LMI problem.

Before proceeding with the main discussion, a brief note on projections is required. Let x be an element in a Hilbert space H and let C be a closed (possibly non-convex) subset of H. Any $c_0 \in C$ such that $\|x - c_0\| \leq \|x - c\|$ for all $c \in C$ will be called a *projection* of x onto C. In the cases of interest here, namely that H is a finite dimensional Hilbert space, there is always at least one such point for each x. If C is convex as well as closed then each x has exactly one such minimum distance point [22]. Any function $P_C : H \to H$ will be called a *projection operator* (for C) if for each $x \in H$,

$$P_C(x) \in C \quad \text{and} \quad \|x - P_C(x)\| \leq \|x - c\| \text{ for all } c \in C.$$

The tangent and lift method is a generalization of Newton's method and can be used to find a point in the intersection of an affine subspace and a manifold. It originated in [21] and is based on a geometric interpretation of an algorithm appearing in [23].

Recall that Newton's method for finding a zero of a function $f : \mathbb{R} \to \mathbb{R}$ is iterative in nature and is given by the recursion

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}.$$

Geometrically speaking, x_{n+1} is the x-axis intercept of the line which is tangent to the graph of f at $(x_n, f(x_n))$. In tangent and lift, the role of the x-axis is replaced by an affine subspace and the role of the graph of f is replaced by a manifold. More precisely, the method works as follows. Let H be a real finite dimensional Hilbert space and suppose $\mathcal{L}$ is an affine subspace of H and that $\mathcal{M}$ is a submanifold of H. Given $x_n \in \mathcal{L}$, and assuming it is possible to calculate projections onto $\mathcal{M}$, let y_n be a projection of x_n onto $\mathcal{M}$. As $\mathcal{M}$ is a manifold, it has a tangent space T at the point y_n. T has a canonical representation as a linear subspace of H and $y_n + T$ can be thought of as an affine subspace of H that is tangent to the manifold at y_n. Assuming $y_n + T$ and $\mathcal{L}$ intersect uniquely, x_{n+1} is taken to be the intersection point of $y_n + T$ and $\mathcal{L}$. As $x_{n+1} \in \mathcal{L}$, the scheme can be iterated.

A graphically representation of the algorithm is given in Figure 1(a). Here the Hilbert space H is $\mathbb{R}^2$, $\mathcal{L}$ is the x-axis, and $\mathcal{M}$ is the graph of a function $f : \mathbb{R} \to \mathbb{R}$. In this case, finding a point in $\mathcal{M} \cap \mathcal{L}$ is equivalent to finding a zero of f. Newton's method can also be employed to solve this problem and for purposes of comparison is also illustrated in Figure 1.

Some points to note. For tangent and lift to work it must be possible to calculate projections onto $\mathcal{M}$. This step replaces the process of 'lifting' x to $(x, f(x))$ in Newton's method. In addition, at least for all points near a solution, each $y_n + T$ must intersect $\mathcal{L}$ uniquely. This essentially

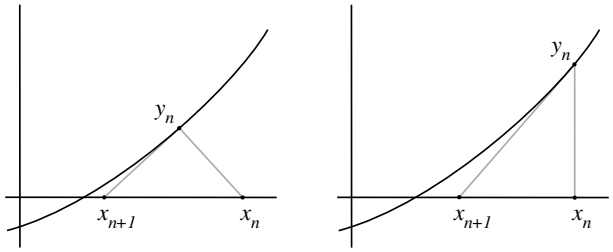

Fig. 1. Two different methods for finding a zero of a function: (a)Tangent and lift; (b)Newton's method.

places a rather strong requirement on the dimensions of $\mathcal{L}$ and $\mathcal{M}$:

$$\dim \mathcal{L} + \dim \mathcal{M} = \dim H. \tag{5}$$

The reasoning is as follows. Firstly, note that if $y \in \mathcal{M}$ and T is its tangent space then $\dim \mathcal{M} = \dim T = \dim(y+T)$. Hence (5) can be interpreted in terms of the dimensions of the affine subspaces $\mathcal{L}$ and $y+T$. Now if the dimensions of two affine subspaces sum to less than the dimension of the ambient space then we would not expect them to intersect; think of two lines in $\mathbb{R}^3$. Alternatively, if the dimensions of two affine subspaces sum to more than the dimension of the ambient space then we would expect them to have multiple intersection points; think of two planes in $\mathbb{R}^3$. It is only when the dimensions of two affine subspaces sum to the dimension of the ambient space that we would expect the affine subspaces to intersect uniquely; think of a plane and a line in $\mathbb{R}^3$.

Suppose now that (5) is satisfied and that x is an intersection point of $\mathcal{M}$ and $\mathcal{L}$. If T is the tangent plane of $\mathcal{M}$ at x, and $x+T$ and $\mathcal{L}$ intersect uniquely (this will generically be the case and in the Newton scheme is equivalent to the requirement that $f'(x) \neq 0$), then, omitting the details, the tangent planes for all points in $\mathcal{M}$ near x will have this property. Hence if x_n 'close to x' implies x_{n+1} is also close to x then the algorithm will be well defined locally near x.

Though the dimension constraint (5) is not explicitly discussed in [21], it is satisfied by the problem studied in that paper and application of the tangent and lift method to that problem results in a (locally) quadratically convergent algorithm. Other applications of tangent and lift are given in [24]. As far as we are aware, tangent and lift methods have only been employed for problems that satisfy (5).

In Problem 2, $\mathcal{M}_r$ is not a manifold but rather a union of manifolds, see Section IV. This means that each point in $\mathcal{M}_r$ lies in a manifold with a well defined tangent space. However, as will be shown, these manifolds are of varying dimensions. Depending on $y \in \mathcal{M}_r$, it may therefore happen that $y+T$ does not intersect $\mathcal{L}$ uniquely. The intersection may be empty or it may contain more than one point. This may happen even arbitrarily close to a solution point.

In order to apply tangent and lift ideas to Problem 2, the approach must be extended to deal with these intersection issues. Our method of doing this is as follows. We consider all points in $\mathcal{L}$ that are of minimum distance to $y_n + T$ and from these points choose x_{n+1} to be the point closest to y_n. As we will see in Section V, x_{n+1} can be found by solving a linearly constrained least squares problem. Numerical experiments demonstrate this methodology leads to a locally convergent algorithm which, though it not always the case, often exhibits local quadratic convergence. In fact, generically, local quadratic convergence would hold if the constraints were defined by a smooth manifold. Unfortunately, this is not the case here and the extension of the tangent and lift methodology to singular spaces therefore presents new challenges for a rigorous convergence theory.

IV. The Geometry of Rank Constrained Positive Semidefinite Matrices

Before proceeding to describe our algorithm for solving the rank constrained LMI problem in greater detail, in this section we collect together some geometric properties of rank constrained positive semidefinite matrices. In particular, we show that $\mathcal{M}_r$ is a union of manifolds and describe the tangent spaces of these manifolds.

Theorem 3: $\mathcal{S}_+^n(s)$ is a connected smooth manifold of dimension $\frac{1}{2}s(2n-s+1)$. The tangent space of $\mathcal{S}_+^n(s)$ at an element X is

$$T_X\mathcal{S}_+^n(s) = \{\Omega X + X\Omega^T \,|\, \Omega \in \mathbb{R}^{n\times n}\}.$$

Proof: See for example Proposition 1.1 in Chapter 5 of [25]. ■

Corollary 4: $\mathcal{M}_r$ is a union of manifolds.

Proof: From (4) it follows that $\mathcal{M}_r$ is a union of terms of the form $S_+^{n_F}(s) \times S_+^{n_G}(t)$. Theorem 3 implies both $S_+^{n_F}(s)$ and $S_+^{n_G}(t)$ are manifolds and the result follows as a product of manifolds is itself a manifold. ■

As the next theorem shows, after applying an appropriate transformation, $T_X\mathcal{S}_+^n(s)$ has a rather simple form.

Theorem 5: Given $X \in \mathcal{S}_+^n(s)$, let

$$X = \Theta\bar{X}\Theta^T, \quad \bar{X} = \begin{bmatrix}\Lambda & 0\\ 0 & 0\end{bmatrix},$$

where $\Theta \in \mathbb{R}^{n\times n}$ is orthogonal and $\Lambda \in \mathcal{S}^s$ is a positive definite diagonal matrix. Then

$$\begin{aligned}\Theta^T T_X\mathcal{S}_+^n(s)\Theta &= T_{\bar{X}}\mathcal{S}_+^n(s)\\ &= \left\{\begin{bmatrix}\Omega_1 & \Omega_2^T\\ \Omega_2 & 0\end{bmatrix} \,\middle|\, \Omega_1 \in \mathcal{S}^s,\ \Omega_2 \in \mathbb{R}^{(n-s)\times s}\right\}.\end{aligned}$$

Proof: Omitted due to space limitations. ■

The following useful fact is obvious from the above characterization of tangent vectors.

Lemma 6: $X \in T_X\mathcal{S}_+^n(s)$ for each $X \in \mathcal{S}_+^n(s)$.

V. ALGORITHM

This section presents our algorithm for solving the rank constrained LMI problem. It contains a description of the algorithm at a conceptual level followed by details of the various components of the algorithm, including the required projections and initialization.

In order to do projections, we need to define an appropriate Hilbert space and associated norm. From now on $\mathcal{S}^n$ will be regarded as a Hilbert space with inner product

$$\langle A, B\rangle = \mathrm{tr}(AB) = \sum_{i,j} a_{ij}b_{ij}.$$

The associated norm is the Frobenius norm $\|A\| = \langle A, A\rangle^{\frac{1}{2}}$.

The sets of most interest will the product space $\mathcal{S}^{n_F} \times \mathcal{S}^{n_G}$ and various subsets of this space such as $\mathcal{S}_+^{n_F}(s) \times \mathcal{S}_+^{n_G}(t)$. The space $\mathcal{S}^{n_F} \times \mathcal{S}^{n_G}$ will be viewed as a Hilbert space with inner product

$$\langle (A, B), (C, D)\rangle = \langle A, C\rangle + \langle B, D\rangle$$

where the two inner products on the right of the equality are the inner products for $\mathcal{S}^{n_F}$ and $\mathcal{S}^{n_G}$ respectively. The associated norm is $\|(A, B)\| = (\|A\|^2 + \|B\|^2)^{\frac{1}{2}}$.

We will have need to refer to the affine tangent space of $\mathcal{S}_+^{n_F}(s) \times \mathcal{S}_+^{n_G}(t)$ at a point (X, Y) as an affine subspace of $\mathcal{S}^{n_F} \times \mathcal{S}^{n_G}$. For this purpose we introduce the following definition.

Definition 7: For $(X, Y) \in \mathcal{S}_+^{n_F}(s) \times \mathcal{S}_+^{n_G}(t)$, define

$$\mathcal{A}_{(X,Y)} = (X, Y) + T_{(X,Y)}(\mathcal{S}_+^{n_F}(s) \times \mathcal{S}_+^{n_G}(t)).$$

Lemma 6 implies that $(X, Y) \in T_{(X,Y)}(\mathcal{S}_+^{n_F}(s) \times \mathcal{S}_+^{n_G}(t))$. Hence, $\mathcal{A}_{(X,Y)} = T_{(X,Y)}(\mathcal{S}_+^{n_F}(s) \times \mathcal{S}_+^{n_G}(t))$ and $\mathcal{A}_{(X,Y)}$ is in fact a linear subspace and not just an affine subspace.

Definition 8: The distance between two non-empty subsets V and W of a vector space with norm $\|\cdot\|$ is

$$\mathrm{dist}(V, W) = \inf\{\|v - w\| \,|\, v \in V, w \in W\}.$$

Similarly, the distance between a point v and non-empty subset W is

$$\mathrm{dist}(v, W) = \inf\{\|v - w\| \,|\, w \in W\}.$$

At a conceptual level the algorithm can be stated as follows.

Algorithm:

Problem Data. $F_0, \ldots, F_m \in \mathcal{S}^{n_F}$, $G_0, \ldots, G_m \in \mathcal{S}^{n_G}$, and $0 \le r \le n_G$.

Initialization. Either choose any $(X_1, Y_1) \in \mathcal{S}^{n_F} \times \mathcal{S}^{n_G}$, or use $(X_1, Y_1) = (F(x), G(x))$ where x is the solution of the semidefinite definite program (6).

repeat

1) Project (X_1, Y_1) onto $\mathcal{M}_r$ to give a new point (X_2, Y_2).
2) Define $\mathcal{B} = \{(X, Y) \in \mathcal{L} \,|\, \mathrm{dist}((X, Y), \mathcal{A}_{(X_2,Y_2)}) = \mathrm{dist}(\mathcal{L}, \mathcal{A}_{(X_2,Y_2)})\}$.
3) $(X_3, Y_3) = \arg\min_{(X,Y)\in\mathcal{B}} \|(X, Y) - (X_2, Y_2)\|$.
4) Set $(X_1, Y_1) = (X_3, Y_3)$.

until (X_1, Y_1) converges to a solution of Problem 2.

Here are some comments regarding the above algorithm. Step 1 is readily calculated via eigenvalue-eigenvector decompositions of X_1 and Y_1. This will be shown in Section V-B below. In Step 2, $\mathcal{B}$ is the set of points in $\mathcal{L}$ that are of minimum distance to $\mathcal{A}_{(X_2,Y_2)}$. Step 3 is the projection of (X_2, Y_2) onto $\mathcal{B}$. Note that as $\mathcal{L}$ and $\mathcal{A}_{(X_2,Y_2)}$ are closed affine subspaces, the distance between them is zero if and only if they intersect. Whether the sets intersect or not, $\mathcal{B}$ itself will always be either a single point or an affine subspace. In the case that $\mathcal{B}$ is a single point, Step 3 is trivial. In the case that $\mathcal{B}$ is an affine subspace, Step 3 is equivalent to solving a linearly constrained least squares problem. Details of how to solve this step are given in Section V-C below. Finally, note that each new (X_1, Y_1) is in $\mathcal{L}$ as $(X_1, Y_1) = (X_3, Y_3) \in \mathcal{B} \subset \mathcal{L}$. Hence the termination criterion of the algorithm can be replaced by 'until $(X_1, Y_1) \in \mathcal{M}_r$'.

A. Initialization

There is no guarantee that the algorithm will converge from an arbitrary initial condition (X_1, Y_1). While a random choice for the initial condition does often work, an alternative choice is to use $(X_1, Y_1) = (F(x), G(x))$ where x is the solution the following semidefinite programming (SDP) problem:

$$\begin{aligned} \min_{x\in\mathbb{R}^m} \quad & \mathrm{tr}(G(x)) \\ \text{subject to} \quad & F(x) \ge 0 \\ & G(x) \ge 0. \end{aligned} \tag{6}$$

This is based on the heuristic that minimizing the trace of a matrix subject to LMI constraints often leads a low rank solution. Applied to a special case of Problem 1, the same initialization scheme is used in both [15] and [26]. This trace minimization heuristic also appears in [16] and [17], and nice insights into why it might be effective can be found in [27].

As we will see in the results section, in some cases the solution of (6) will satisfy $\mathrm{rank}\, G(x) \le r$, in which case the overall problem is solved. In general, however, the solution of the SDP gives a singular matrix $G(x)$ which does not satisfy this rank constraint.

B. Projecting onto $\mathcal{M}_r$

Step 1 of the algorithm is the projection of a point (X_1, Y_1) onto the set $\mathcal{M}_r$. This projection is equivalent to componentwise projection of X_1 onto $\mathcal{S}_+^{n_F} = \bigcup_{s=0}^{n_F} \mathcal{S}_+^{n_F}(s)$ and Y_1 onto $\bigcup_{s=0}^{r} \mathcal{S}_+^{n_G}(s)$.

The projection of $X \in \mathcal{S}^n$ onto $\bigcup_{s=0}^{r} \mathcal{S}_+^{n}(s)$ is given by the following result. More precisely, the result gives *a* projection of X onto $\bigcup_{s=0}^{r} \mathcal{S}_+^{n}(s)$ as, for r strictly less than n, the set $\bigcup_{s=0}^{r} \mathcal{S}_+^{n}(s)$ is non-convex and projections onto this set are not always guaranteed to be unique.

Theorem 9: Given $X \in \mathcal{S}^n$ and $0 \leq r \leq n$, let $X = \Theta \operatorname{diag}(\lambda_1, \ldots, \lambda_n)\Theta^T$ with $\lambda_1 \geq \ldots \geq \lambda_n$ and Θ a real orthogonal matrix. Define $P_r : \mathcal{S}^n \to \mathcal{S}^n$ as follows,

$$P_r(X) = \Theta \operatorname{diag}(\max\{\lambda_1, 0\}, .., \max\{\lambda_r, 0\}, 0, .., 0)\Theta^T.$$

Then $P_r(X)$ is a best approximant in $\bigcup_{s=0}^{r} \mathcal{S}_+^n(s)$ to X in the Frobenius norm.

Proof: Omitted due to space limitations. ■

C. Projecting onto $\mathcal{B}$

How to project onto $\mathcal{B}$ is detailed in Theorem 11 below. We will need the following definition.

Definition 10: Suppose $(X, Y) \in \mathcal{M}_r$, $s = \operatorname{rank}(X)$, and $t = \operatorname{rank}(Y)$. Define $\mathcal{T}_{(X,Y)} \subset \mathcal{S}^{n_F} \times \mathcal{S}^{n_G}$ by

$$\mathcal{T}_{(X,Y)} = \left\{ \begin{bmatrix} \Omega_1 & \Omega_2^T \\ \Omega_2 & 0 \end{bmatrix} \Big| \, \Omega_1 \in \mathcal{S}^s, \ \Omega_2 \in \mathbb{R}^{(n_F - s)\times s} \right\} \times \left\{ \begin{bmatrix} \Delta_1 & \Delta_2^T \\ \Delta_2 & 0 \end{bmatrix} \Big| \, \Delta_1 \in \mathcal{S}^t, \ \Delta_2 \in \mathbb{R}^{(n_G - t)\times t} \right\}. \tag{7}$$

Note that $\mathcal{T}_{(X,Y)}$ is a linear subspace of dimension $N = s(s+1)/2 + (n_F - s)s + t(t+1)/2 + (n_G - t)t$ that depends on (X, Y) through s and t.

Theorem 11: Suppose $(X, Y) \in \mathcal{M}_r$ and define $s = \operatorname{rank}(X)$ and $t = \operatorname{rank}(Y)$. Then X and Y have eigenvalue-eigenvector decompositions

$$X = VDV^T, \qquad D = \begin{bmatrix} \Lambda_X & 0 \\ 0 & 0 \end{bmatrix}, \tag{8}$$

$$Y = WEW^T, \qquad E = \begin{bmatrix} \Lambda_Y & 0 \\ 0 & 0 \end{bmatrix}, \tag{9}$$

where $V \in \mathbb{R}^{n_F \times n_F}$ and $W \in \mathbb{R}^{n_G \times n_G}$ are orthogonal, and $\Lambda_X \in \mathcal{S}^s$ and $\Lambda_Y \in \mathcal{S}^t$ are positive definite diagonal matrices.

Let $(B_1, C_1), \ldots, (B_N, C_N)$ be any basis for $\mathcal{T}_{(X,Y)}$, see (7), and define $A \in \mathbb{R}^{(n_F^2 + n_G^2)\times N}$ by

$$A = \begin{bmatrix} \operatorname{vec}(B_1) & \ldots & \operatorname{vec}(B_N) \\ \operatorname{vec}(C_1) & \ldots & \operatorname{vec}(C_N) \end{bmatrix}.$$

Using the F_i's and G_i's of (1) and (2), and V and W from (8) and (9), define $b \in \mathbb{R}^{n_F^2 + n_G^2}$ and $B \in \mathbb{R}^{(n_F^2 + n_G^2)\times m}$ by

$$b = \begin{bmatrix} \operatorname{vec}(V^T F_0 V) \\ \operatorname{vec}(W^T G_0 W) \end{bmatrix},$$

$$B = \begin{bmatrix} \operatorname{vec}(V^T F_1 V) & \ldots & \operatorname{vec}(V^T F_m V) \\ \operatorname{vec}(W^T G_1 W) & \ldots & \operatorname{vec}(W^T G_m W) \end{bmatrix}.$$

Let

$$C = \begin{bmatrix} A & -B \end{bmatrix}.$$

If $F(\cdot)$ and $G(\cdot)$ are the functions defined in (1) and (2), and $\|\cdot\|_2$ denotes the standard vector 2-norm, then the projection of (X, Y) onto $\mathcal{B}$ equals $(F(x), G(x))$ where (v, x) is a minimizing solution of

$$\min_{v \in \mathbb{R}^N, \, x \in \mathbb{R}^m} \quad \left\| Bx + b - \begin{bmatrix} \operatorname{vec}(D) \\ \operatorname{vec}(E) \end{bmatrix} \right\|_2 \tag{10}$$

$$\text{subject to} \quad C^T C \begin{bmatrix} v \\ x \end{bmatrix} = C^T b. \tag{11}$$

If $(F_1, G_1), \ldots, (F_m, G_m)$ are linearly independent, then x is unique.

Proof: Omitted due to space limitations. ■

Hence projecting onto $\mathcal{B}$ is equivalent to solving the linearly constrained least squares problem (10), (11). Such problems can be solved in a number of ways, see for example [28]. A basic solution approach is as follows. First parameterize the points in the constraint set (using any particular solution and a basis for the null space of $C^T C$). Using this parametrization, transform the original constrained problem into a (lower dimensional) unconstrained least squares problem. Finally, use the solution of this new problem and the parametrization mapping to obtain a solution of the original problem.

VI. Numerical Experiments

This section contains some numerical experiments. Algorithm performance is investigated using both randomly generated problems and by applying the algorithm to a particular output feedback problem.

All computational results were obtained using a 2 GHz Pentium 4 machine, with 512Mb of memory, running Windows XP Professional. Our algorithm was coded using Matlab 6.5. For each problem, the initial condition was found by solving the semidefinite programming problem (6) using SeDuMi [29].

Convergence Criteria. For purposes of determining convergence, that is determining positive semidefiniteness of the matrices $F(x)$ and $G(x)$ and the rank of $G(x)$, eigenvalues of these matrices will we considered 0 if they have magnitude 10^{-12} or less. (For the problems considered, typical non-zero eigenvalues have magnitudes between 10^1 and 10^{-2}.)

A. Random Problems

All results in this subsection are for randomly generated problems. Each problem is generated as follows. Let $\mathcal{N}(0, 1)$ denote the normal distribution with zero mean and variance 1. Each entry of the matrices $F_1, \ldots, F_m$ and $G_1, \ldots, G_m$ is drawn from $\mathcal{N}(0, 1)$. To ensure feasibility, F_0 and G_0 are set to $F_0 = V_F D_F V_F^T - \sum_{i=0}^{m} \xi_i F_i$ and $G_0 = V_G D_G V_G^T - \sum_{i=0}^{m} \xi_i G_i$, where each ξ_i is drawn from $\mathcal{N}(0, 1)$; V_F and V_G are randomly generated orthogonal matrices; and D_F and D_G are randomly generated diagonal matrices: each diagonal entry in D_F is drawn from $\mathcal{N}(0, 1)$ and set to zero if it is negative, while r diagonal entries in D_G are drawn from the uniform distribution on the interval $[0, 1]$ and the others set to zero.

Table I contains results for $n_F = 10$, $n_G = 10$, $r = 5$ and various values of m. For each value of m in the table, the algorithm is given 100 random problems to solve. Listed are a distribution of the number of iterations taken for the algorithm to converge, the average number of iterations, and the average CPU time. Iteration 1 is the initialization step based on the rank minimization heuristic.

TABLE I

EXPERIMENTS FOR RANDOM F AND G WITH $n_F = 10$, $n_G = 10$ AND $r = 5$. i DENOTES THE AVERAGE NUMBER OF ITERATIONS AND T DENOTES AVERAGE CPU TIME IN SECONDS. i AND T DO NOT INCLUDE THE PROBLEMS THAT HAD NOT CONVERGED AFTER 100 ITERATIONS. THE NUMBER OF SUCH PROBLEMS FOR EACH m IS GIVEN IN THE 'NC' OR 'NON-CONVERGENCE AFTER 100 ITERATIONS' COLUMN.

m	iterations					i	T
	1	$2-10$	$11-20$	$21-100$	NC		
10	97	3	0	0	0	1.1	0.36
20	33	41	13	5	3	10	1.4
30	29	53	8	4	6	7.0	1.5

For $m = 10$, solutions for all 100 problems were found. The rank minimization heuristic was very effective, finding solutions for almost all the problems. The 3 problems that were not solved in this first iteration, were all solved using less than 10 iterations. For both $m = 20$ and $m = 30$, the rank minimization heuristic was no longer quite as successful though it did still manage to find a solution in about 30% of cases. Approximately 90% of problems were solved in 20 iterations or less while about 5% had not converged after 100 iterations. Examination of the problems that had not converged after 100 iterations showed that about half were slowly converging to a solution while progress for the other half seemed to have stopped. For all values of m, the average CPU time (excluding problems that had not converged after 100 iterations) was less than 1.5 seconds.

Results for some larger problems are given in Table II. Here again, for each value of m, the algorithm was given 100 random problems to solve. There are a number of observations to be made. Let us first consider the results for $m = 20$ and $m = 60$. In both these cases, performance was again very good. In terms of average iterations and the distribution of iterations, the results for $m = 20$ and $m = 60$ were respectively very similar to the results for $m = 10$ and $m = 30$ in Table I. Average CPU times understandably increased due to the larger problem sizes. For $m = 40$, while 78 problems converged in 20 iterations or less, 17 had not converged after 100 iterations. (After 1000 iterations, 7 had not converged. Of these 4 seemed to be converging slowly.) It is not completely clear why this value of m leads to slower convergence.

TABLE II

EXPERIMENTS FOR RANDOM F AND G WITH $n_F = 20$, $n_G = 15$ AND $r = 10$. †FOR $m = 40$, 7 PROBLEMS HAD NOT CONVERGED AFTER 1000 ITERATIONS.

m	iterations					i	T
	1	$2-10$	$11-20$	$21-100$	NC		
20	94	3	3	0	0	1.5	2.7
40	45	23	10	5	$17^\dagger$	5.7	23
60	32	56	7	2	3	5.2	23

B. Reduced Order Output Feedback Problem

In this subsection we present results of using the algorithm to solve a particular reduced order output feedback problem. The problem is taken from [14] (see also [13] and [15]). The problems considered in [14], including the particular problem we will consider here, result in the following special case of Problem 1 ('< 0' means 'is negative definite'):

Problem 12: Find $X, Y \in \mathcal{S}^n$ such that

$$\begin{aligned} E_X X F_X + (E_X X F_X)^T + Q_X &< 0 \\ E_Y Y F_Y + (E_Y Y F_Y)^T + Q_Y &< 0 \\ \begin{bmatrix} X & I \\ I & Y \end{bmatrix} &\geq 0 \\ \operatorname{rank} \begin{bmatrix} X & I \\ I & Y \end{bmatrix} &\leq n + n_c. \end{aligned}$$

The data in the problem are the matrices E_X, F_X, Q_X, E_Y, F_Y, Q_Y and the non-negative integer n_c. I is the $n \times n$ identity matrix. For the output synthesis problems considered, n denotes the order of the LTI system to be stabilized by output feedback and $0 \leq n_c \leq n$ denotes a user specified bound on the order of the dynamic output feedback controller. There exists a dynamic output feedback controller of order $\leq n_c$ that stabilizes the system if and only if Problem 12 has a solution. A solution to the feedback synthesis problem can be found from any solution of Problem 12 by solving an additional LMI or via explicit formulas [2].

The system considered is a two-mass-spring system with state space representation:

$$\dot{x} = Ax + Bu, \qquad y = Cx$$

where

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \end{bmatrix}, \; B = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \; C = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}^T.$$

Given $\alpha > 0$ we wish to find an order 2 dynamic controller that places the closed loop poles in the set $\{z \in \mathbb{C} \,|\, \mathrm{Re}(z) \leq -\alpha\}$. As is shown in [14], this is equivalent to solving Problem 12 with $n = 4$, $n_c = 2$, $E_X = B^{\perp}(A + \alpha I)$, $F_X = B^{\perp T}$, $Q_X = 0$, $E_Y = C^{T\perp}$, $F_Y = (A + \alpha I)C^{T\perp T}$, and $Q_Y = 0$, where

$$B^{\perp} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad C^{T\perp} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Technical note: Problem 12 contains two constraints of the form '$Z < 0$'. In the computations, each of these has been replaced by '$-Z \geq \epsilon I$' with $\epsilon = 10^{-6}$.

The problem was solved for two of the same values of α given in [14]. The results are listed in Table III. As can be seen, the harder problem ($\alpha = 0.42$) took longer to converge than the easier problem ($\alpha = 0.2$). Hence, speed

TABLE III

RESULTS FOR THE TWO-MASS-SPRING SYSTEM. α DENOTES THE STABILITY DEGREE, i THE NUMBER OF ITERATIONS, AND T THE CPU TIME IN SECONDS.

α	i	T
0.2	289	15
0.42	1503	70

of convergence seems to be influenced by the size of the feasible set.

VII. CONCLUSIONS

In this paper we have presented an algorithm for solving the rank constrained LMI problem. Like all other algorithms that attempt to solve this problem, the algorithm is heuristic in nature and convergence from an arbitrary initial condition is not guaranteed. Though the convergence properties of the algorithm are not yet completely understood, as demonstrated by the experiments, the algorithm can be quite effective. Given that the algorithm is based on a Newton like methodology, it is not completely apparent why the algorithm is not always locally quadratically convergent and further investigations need to be made in this regard.

VIII. ACKNOWLEDGEMENTS

The first and third authors acknowledge the support of the Australian Research Council through grants DP0450539 and A00105829.

The last two authors were partially supported by DAAD project PPP Australia/Germany D0243869.

National ICT Australia is funded by the Australian Department of Communications, Information Technology and the Arts and the Australian Research Council through Backing Australia's Ability and the ICT Centre of Excellence Program.

REFERENCES

[1] L. El Ghaoui and P. Gahinet, "Rank minimization under LMI constraints: a framework for output-feedback problems," in *Proceedings European Control Conference*, 1993.

[2] R. E. Skelton, T. Iwasaki, and K. M. Grigoriadis, *A Unified Algebraic Approach to Linear Control Design*. London: Taylor & Francis, 1998.

[3] K. C. Goh, M. G. Safonov, and J. H. Ly, "Robust synthesis via bilinear matrix inequalities," *International J. of Robust and Nonlinear Control*, vol. 6, no. 9-10, pp. 1079–1095, 1996.

[4] M. Mesbahi, M. G. Safonov, and G. P. Papavassilopoulos, "Bilinearity and complementarity in robust control," in *Advances on Linear Matrix Inequality Methods in Control*, L. El Ghaoui and S.-I. Niculescu, Eds. SIAM, 1999, pp. 269–292.

[5] Y. Nesterov and A. Nemirovskii, *Interior-Point Poylnomial Algorithms in Convex Programming*. Philadelphia: SIAM, 1994.

[6] S. Boyd and L. Vandenberghe, "Semidefinite programming relaxations of non-convex problems in control and combinatorial optimization," in *Communications, Computation, Control and Signal Processing – a Tribute to Thomas Kailath*, A. Paulraj, V. Roychowdhury, and C. Schaper, Eds. New York: Kluwer Academic Publishers, 1997.

[7] L. Vandenberghe and S. Boyd, "Semidefinite programming," *SIAM Review*, vol. 38, no. 1, pp. 49–95, 1996.

[8] J. David and B. De Moor, "The opposite of analytic centering for solving minimum rank problems in control and identification," in *Proc. 32nd IEEE Conference on Decision and Control*, San Antonio, Texas, 1993, pp. 2901–2902.

[9] ——, "Designing reduced order output feedback controllers using a potential reduction method," in *Proc. American Control Conference*, Baltimore, Maryland, 1994, pp. 845–849.

[10] L. El Ghaoui, F. Oustry, and M. Ait Rami, "A cone complementarity linearization algorithm for static output-feedback and related problem," *IEEE Trans. on Automatic Control*, vol. 42, no. 8, pp. 1171–1176, 1997.

[11] S. Ibaraki and M. Tomizuka, "Rank minimization approach for solving BMI problems with random search," in *Proceedings American Control Conference*, 2001, pp. 25–27.

[12] F. Leibfritz, "An LMI-based algorithm for designing suboptimal static $\mathcal{H}_2/\mathcal{H}_\infty$ output feedback controllers," *SIAM J. Control Optim.*, vol. 39, no. 6, pp. 1711–1735, 2001.

[13] K. M. Grigoriadis and R. Skelton, "Low-order control design for LMI problems using alternating projection methods," *Automatica*, vol. 32, no. 8, pp. 1117–1125, 1996.

[14] E. Beran and K. Grigoriadis, "A combined alternating projections and semideinite programming algorithm for low-order control design," in *Proceedings IFAC 96*, vol. C, San Franscisco, 1996, pp. 85–90.

[15] K. M. Grigoriadis and E. B. Beran, "Alternating projection algorithms for linear matrix inequalities problems with rank constraints," in *Advances on Linear Matrix Inequality Methods in Control*, L. El Ghaoui and S.-I. Niculescu, Eds. SIAM, 1999, pp. 251–267.

[16] T. E. Pare, "Analysis and control of nonlinear systems," Ph.D. dissertation, Stanford University, Stanford, CA, 2000.

[17] M. Mesbahi, "On the semi-definite programming solution of the least order dynamic output feedback synthesis," in *Proceedings American Control Conference*, San Diego, CA, 1999, pp. 2355–2359.

[18] B. Fares, P. Apkarian, and D. Noll, "An augmented Lagrangian method for a class of LMI-constrained problems in robust control theory," *Int. J. Control*, vol. 74, no. 4, pp. 348–360, 2001.

[19] P. Apkarian, D. Noll, and H. D. Tuan, "Fixed-order H_∞ control design via a partially augmented Lagrangian method," *Int. J. Robust Nonlinear Control*, vol. 13, pp. 1137–1148, 2003.

[20] B. Fares, D. Noll, and P. Apkarian, "Robust control via sequential semidefinite programming," *SIAM J. Control Optim.*, vol. 40, no. 6, pp. 1791–1820, 2002.

[21] M. T. Chu, "Numerical methods for inverse singular value problems," *SIAM J. Numer. Anal.*, vol. 29, no. 3, pp. 885–903, 1992.

[22] D. Luenberger, *Optimization by Vector Space Methods*. New York: Wiley, 1969.

[23] S. Friedland, J. Nocedal, and M. L. Overton, "The formulation and analysis of numerical methods for inverse eigenvalue problems," *SIAM J. Numer. Anal.*, vol. 24, no. 3, pp. 634–667, 1987.

[24] M. T. Chu, "Inverse eigenvalue problems," *SIAM Rev.*, vol. 40, no. 1, pp. 1–39, 1998.

[25] U. Helmke and J. B. Moore, *Optimization and Dynamical Systems*. London: Springer-Verlag, 1994.

[26] T. Iwasaki, "The dual iteration for fixed-order control," *IEEE Trans. on Automatic Control*, vol. 44, no. 4, pp. 783–788, 1999.

[27] M. Fazel, "Matrix rank minimization with applications," Ph.D. dissertation, Stanford University, Stanford, CA, 2002.

[28] C. L. Lawson and R. J. Hanson, *Solving Least Squares Problems*, ser. Classics in Applied Mathematics. Philadelphia: SIAM, 1995, vol. 15.

[29] J. Sturm, "Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones," *Optimization Methods and Software*, vol. 11–12, pp. 625–653, 1999, special issue on Interior Point Methods (CD supplement with software).