A NONLINEAR SDP ALGORITHM FOR STATIC OUTPUT FEEDBACK PROBLEMS IN *COMPl$_e$ib*

M. Kočvara [*,**,1] **F. Leibfritz** [***] **M. Stingl** [****]
D. Henrion [†,*,2]

[*] *Czech Technical University, Faculty of Electrical Engineering, Technická 2, 166 27 Prague, Czech Republic*
[**] *Institute of Information Theory and Automation, Pod vodárenskou věží 4, 182 08 Prague, Czech Republic*
[***] *Department of Mathematics, University of Trier, Universitätsring 15, 54286 Trier, Germany*
[****] *Institute of Applied Mathematics, University of Erlangen, Martensstr. 3, 91058 Erlangen, Germany*
[†] *LAAS-CNRS, 7 Avenue du Colonel Roche, 31077 Toulouse, France*

Abstract: We present an algorithm for the solution of static output feedback problems formulated as semidefinite programs with bilinear matrix inequality constraints and collected in the library *COMPl$_e$ib* . The algorithm, based on the generalized augmented Lagrangian technique, is implemented in the publicly available general purpose software PENBMI. Numerical results demonstrate the behavior of the code. *Copyright© 2005 IFAC*

Keywords: static output feedback, bilinear matrix inequalities

1. INTRODUCTION

Even though several relevant control problems boil down to solving convex linear matrix inequalities (LMI)—see (Boyd *et al.* 1994) for a long list—there are still fundamental problems for which no convex LMI formulation has been found. Typical examples are simultaneous stabilization by a unique controller of a set of linear plants and stabilization of a linear plant by a fixed-order controller. The most fundamental of these problems is perhaps static output feedback (SOF) stabilization: given a triplet of matrices A,B,C of suitable dimensions, find a matrix F such that the eigenvalues of matrix $A + BFC$ are all in a given region of the complex plane, say the open left half-plane (Blondel and Tsitsiklis 2000).

No LMI formulation is known for the static output feedback stabilization problem, but a straightforward application of Lyapunov's stability theory leads to a bilinear matrix inequality (BMI) formulation: matrix $A + BFC$ has all its eigenvalues in the open left half-plane if and only if there exists a matrix X such that

$$(A+BFC)^T X+(A+BFC)X \prec 0, \quad X = X^T \succ 0$$

where $\prec 0$ and $\succ 0$ stand for positive and negative definite, respectively.

BMI formulation of the control problems was made popular in the mid 1990s; there were,

[1] Supported by project 102/05/0011 of the Czech Science Foundation.
[2] Supported by project ME 698/2003 of the Ministry of Education of the Czech Republic.

however, no computational methods for solving non-convex BMIs, in contrast with convex LMIs for which powerful interior-point algorithms were available. One decade later, this unsatisfactory state in BMI solvers is almost unchanged. There were several attempts to solve BMI problems numerically, based on branch-and-bound schemes (Goh *et al.* 1995), generalized Benders decomposition (Beran *et al.* 1997) concave minimization (Apkarian and Tuan 1999), various linearization algorithms (El Ghaoui *et al.* 1997).

More recently, several researchers applied nonlinear optimization techniques to BMI problems, with moderate success so far. Interior-point constrained trust region methods are proposed in (Leibfritz and Mostafa 2002) in the special case of static output feedback and low-order controller design BMIs. The method is a sequential minimization method of a logarithmic barrier function subject to a nonlinear matrix constraint. A similar approach using a sophisticated method to the minimization of the unconstrained subproblems was proposed in (Jarre 2000).

The framework of the presented algorithm is given by the augmented Lagrangian method. It is based on the method introduced in (Polyak 1992) for convex optimization problems. A generalization of this method for convex semidefinite programming (SDP) problems was recently proposed in (Kočvara and Stingl 2003). More recently, the algorithm has been generalized to nonlinear semidefinite programming problems. The non-convex unconstrained minimization subproblems are solved either by a variant of the modified Newton method or by the trust-region algorithm. This algorithm for general non-convex SDPs was adopted to BMI problems and gave rise to a specialized code called PENBMI. To our knowledge, PENBMI is the first available general-purpose code for BMIs[3]. It is the purpose of this paper to show that the code can efficiently solve many nontrivial static output feedback problems collected in the publicly available set *COMPl$_e$ib*.

2. *COMPl$_e$ib*

We present a short description of the benchmark collection *COMPl$_e$ib* : the *CO*nstrained *M*atrix–optimization *P*roblem *lib*rary (Leibfritz 2003)[4]. *COMPl$_e$ib* can be used as a benchmark collection for a very wide variety of algorithms solving matrix optimization problems. For example it can be used for testing solvers for nonlinear SDPs, BMI problems or linear SDPs and other related matrix problems. Currently *COMPl$_e$ib* consists of 124 examples collected from the engineering literature and real–life applications for LTI control systems of the form

$$\begin{aligned}\dot{x}(t) &= Ax(t) + B_1w(t) + Bu(t),\\ z(t) &= C_1x(t) + D_{11}w(t) + D_{12}u(t),\\ y(t) &= Cx(t) + D_{21}w(t),\end{aligned} \quad (1)$$

where $x \in \mathbb{R}^{n_x}$, $u \in \mathbb{R}^{n_u}$, $y \in \mathbb{R}^{n_y}$, $z \in \mathbb{R}^{n_z}$, $w \in \mathbb{R}^{n_w}$ denote the state, control input, measured output, regulated output, and noise input, respectively.

The heart of *COMPl$_e$ib* is the MATLAB function file *COMPleib.m*. This function returns the data matrices A, B_1, B, C_1, C, D_{11}, D_{12} and D_{21} of (1) of each individual *COMPl$_e$ib* example. For more details we refer to (Leibfritz and Lipinski 2003). Depending on specific control design goals, it is possible to derive particular matrix optimization problems using the data matrices provided by *COMPl$_e$ib*. A non exhaustive list of matrix optimization problems arising in feedback control design are stated in (Leibfritz 2003). Many more control problems leading to NSDPs, BMIs or SDPs can be found in the literature.

Here we only state the BMI formulation of two basic static output feedback control design problems: SOF–$\mathcal{H}_2$ and SOF–$\mathcal{H}_\infty$. The goal is to determine the matrix $F \in \mathbb{R}^{n_u \times n_y}$ of the SOF control law $u(t) = Fy(t)$ such that the closed loop system

$$\begin{aligned}\dot{x}(t) &= A(F)x(t) + B(F)w(t),\\ z(t) &= C(F)x(t) + D(F)w(t),\end{aligned} \quad (2)$$

fulfills some specific control design requirements, where $A(F) = A + BFC$, $B(F) = B_1 + BFD_{21}$, $C(F) = C_1 + D_{12}FC$, $D(F) = D_{11} + D_{12}FD_{21}$.

We begin with the SOF–$\mathcal{H}_2$ problem: *Suppose that* $D_{11} = 0$ *and* $D_{21} = 0$. *Find a SOF gain* F *such that* $A(F)$ *is Hurwitz and the* $\mathcal{H}_2$*–norm of (2) is minimal.* This problem can be rewritten to the following $\mathcal{H}_2$–BMI problem formulation, see, e.g. (Leibfritz 2003):

$$\min \ Tr(X) \quad \text{s.t.} \quad Q \succ 0,$$
$$(A + BFC)Q + Q(A + BFC)^T + B_1B_1^T \preceq 0,$$
$$\begin{bmatrix} X & (C_1 + D_{12}FC)Q \\ Q(C_1 + D_{12}FC)^T & Q \end{bmatrix} \succeq 0, \quad (3)$$

where $Q \in \mathbb{R}^{n_x \times n_x}$, $X \in \mathbb{R}^{n_z \times n_z}$. Note, (3) is bilinear, hence non-convex in F and Q.

$\mathcal{H}_\infty$ synthesis is an attractive model–based control design tool and it allows incorporation of model uncertainties in the control design. The optimal SOF–$\mathcal{H}_\infty$ problem can be formally stated in the following term: *Find a SOF matrix* F *such that* $A(F)$ *is Hurwitz and the* $\mathcal{H}_\infty$*–norm of (2) is minimal.* We consider the following well known $\mathcal{H}_\infty$–BMI version, see, e.g. (Leibfritz 2003):

[3] See `http://www.penopt.com` for a free developer version.
[4] See `http://www.mathematik.uni-trier.de/~leibfritz/Proj_TestSet/NSDPTestSet.htm`

$$\min \ \gamma \quad \text{s.t.} \quad X \succ 0, \quad \gamma > 0,$$
$$\begin{bmatrix} A(F)^T X + XA(F) & XB(F) & C(F)^T \\ B(F)^T X & -\gamma\, I_{n_w} & D(F)^T \\ C(F) & D(F) & -\gamma\, I_{n_z} \end{bmatrix} \prec 0, \tag{4}$$

where $\gamma \in \mathbb{R}$, $X \in \mathbb{R}^{n_x \times n_x}$. Due to the bilinearity of the free matrix variables F and X, the BMI–formulation of the SOF–$\mathcal{H}_\infty$ is a non–convex and nonlinear constrained optimization problem, too.

3. THE ALGORITHM

Our goal is to solve optimization problems with a nonlinear objective subject to matrix inequalities as constraints:

$$\min_{x \in \mathbb{R}^n} f(x) \quad \text{s.t.} \quad \mathcal{A}(x) \preccurlyeq 0\,. \tag{5}$$

Here $f : \mathbb{R}^n \to \mathbb{R}$ is a C^2 function and $\mathcal{A} : \mathbb{R}^n \to \mathbb{S}^m$ a generally non-convex smooth matrix operator.

The method is based on the penalty/barrier function $\Phi_p : \mathbb{S}^m \to \mathbb{S}^m$, $p \in \mathbb{R}_+$, defined as follows:

$$\Phi_p(\mathcal{A}(x)) = -p^2(\mathcal{A}(x) - pI)^{-1}(x) - pI\,. \tag{6}$$

The advantage of this choice is that we can easily compute the first and second derivatives of Φ_p (Kočvara and Stingl 2003).

It is straightforward to show that for any $p > 0$

$$\mathcal{A}(x) \preccurlyeq 0 \iff \Phi_p(\mathcal{A}(x)) \preccurlyeq 0\,.$$

Hence, for any $p > 0$, problem (5) has the same solution as the following "augmented" problem

$$\min_{x \in \mathbb{R}^n} f(x) \quad \text{s.t.} \quad \Phi_p(\mathcal{A}(x)) \preccurlyeq 0\,. \tag{7}$$

The Lagrangian of (7) can be viewed as a (generalized) augmented Lagrangian of (5):

$$F(x, U, p) = f(x) + \operatorname{tr}\left(U\, \Phi_p\left(\mathcal{A}(x)\right)\right); \tag{8}$$

here $U \in \mathbb{S}^m_+$ are a Lagrangian multiplier associated with the inequality constraint.

The basic algorithm combines ideas of the (exterior) penalty and (interior) barrier methods with the augmented Lagrangian method.

Algorithm 3.1. (Main algorithm). Let x^1 and U^1 be given. Let $p^1 > 0$. For $k = 1, 2, \ldots$ repeat until a stopping criterion is reached:

(i) Find x^{k+1} satisfying $\|\frac{\partial}{\partial x} F(x, U^k, p^k)\| \leq \alpha^k$ for given $\alpha^k > 0$
(ii) $U^{k+1} = D_{\mathcal{A}} \Phi_p(\mathcal{A}(x); U^k)$
(iii) $p^{k+1} < p^k$.

With assumptions on problem (5), one can prove that any cluster point of the sequence $\{(x_k, U_k)\}_{k>0}$ generated by Algorithm 3.1 is a KKT point of (5). The proof is rather technical and long (about 40 pages) and we refer for it to (Stingl 2004).

3.1 Unconstrained minimization

The tool used in Step (i) of Algorithm 3.1 is a version of the Modified Newton method. The search direction is computed by the algorithm below. The step length is calculated by a gradient free line search satisfying the Armijo condition.

Algorithm 3.2. (Unconstrained minimization).

(i) Given a current iterate (x, U, p), compute the gradient g and Hessian H of F at x.
(ii) Perform Cholesky factorization of H. If the factorization fails, go to the next step; otherwise, set $\widehat{H} = H$ and go to Step (iv).
(iii) Find $\beta \in [-\lambda_{\min}, -2\lambda_{\min}]$, where $\lambda_{\min}$ is the minimal eigenvalue of H and set $\widehat{H} = H + \beta I$.
(iv) Calculate the search direction $d = -\widehat{H}^{-1} g$.
(v) Perform Armijo-type line search in direction d. Denote the step-length by s.
(vi) Set $x_{\text{new}} = x + sd$.

Obviously, for a convex F, this is just a Newton step with line search. In the non-convex case β is found by a sequence of Cholesky factorizations.

Numerical tests indicated that in the quality of the search direction gets poor, if we choose β too close to $-\lambda_{\min}$. In this case we use a bisection technique to calculate $\lambda_{\min}$ almost exactly and replace β by $-1.5\lambda_{\min}$.

Algorithm 3.2 proved to be quite robust as long as the Hessian H of F is not too ill conditioned. In the ill conditioned case, we are still able to calculate approximations of KKT-points in many cases, but the precision we achieve is comparably low. Motivated by this fact, we implemented—as an alternative to Algorithm 3.2—a version of the trust region method. The trust region variant is often slower, but more robust in a neighborhood of first order points. Therefore we combine both approaches in the following way: At the beginning (typically during the first 10 to 15 iterations) of Algorithm 3.1 we use the first approach to solve step (i). As soon as a certain criterion is met or when we run into numerical difficulties, the trust region variant is used instead. For most cases very few (3 to 5) iterations are sufficient to improve the precision of the solution.

3.2 Penalty and multiplier update, stopping criteria

Given an initial iterates x^1 and U^1, p^1 is chosen large enough to satisfy the inequality

$$p^1 I - \mathcal{A}(x^1) \succ 0.$$

In the following iterations we distinguish two cases: If the condition

$$\lambda_{\max}(\mathcal{A}(x^{k+1})) \leq \pi_{\max} p^k \tag{9}$$

is valid, the penalty parameter is updated using the formula

$$p^{k+1} = \max(\pi p^k, \lambda_{\max}(\mathcal{A}(x^{k+1}))), \qquad (10)$$

where $\lambda_{\max}(\mathcal{A}(x^{k+1})) \in \big(0, p^k\big)$ denotes the maximal eigenvalue of $\mathcal{A}(x^{k+1})$ and $\pi \le \pi_{\max} < 1$ are constant factors typically chosen between 0.1 and 0.6. Otherwise, if (9) is violated we set $x^{k+1} = x^1, p^{k+1} = p^1$ and $U^{k+1} = \gamma U_1$, where $\gamma > 1$ is a predefined constant and restart Algorithm 3.1. When certain $p_{\min}$ (typically 10^{-6}) is reached, the penalty parameter is kept constant, as long as the inequality $\lambda_{\max}(\mathcal{A}(x^{k+1})) \le \lambda_{\max}(\mathcal{A}(x^k))$ holds. Note that the theory allows for a certain range of the parameters. The actual values of these parameters (within the allowed range) is, as always, the result of many experiments.

For our choice of the penalty function Φ_p and with

$$\mathcal{Z}(x) = -(\mathcal{A}(x) - pI)^{-1} \qquad (11)$$

the update of the multiplier U can be written as

$$U^{k+1} = (p^k)^2 \mathcal{Z}(x) U^k \mathcal{Z}(x). \qquad (12)$$

This formula may lead, in certain situations, to big changes in the approximate multipliers. These changes may lead to a large number of Newton steps in the subsequent iteration. Also, it may happen that already after first few steps the multipliers become ill-conditioned and the algorithm suffers from numerical difficulties. Therefore, we restrict the multipliers from (12) by the formula

$$U^{new} = U^k + \lambda_A (U^{k+1} - U^k),$$

where $0 < \lambda_{Ak} < 1$ for all k. Typical values for λ_{Ak} are 0.5 in the beginning and 0.95 in a neighborhood of the approximate solution.

In the first phase of Algorithm 3.1, the approximate minimization of F is stopped when $\|\frac{\partial}{\partial x} F(x, U, p)\| \le \alpha$, where $\alpha = 0.01$ is a good choice in most cases. In the second phase, after switching to trust region method, α is reduced in each outer iteration by a constant factor, until a certain $\alpha_{\min}$ (typically 10^{-6}) is reached.

Algorithm 3.1 is stopped if $\lambda_{\max}(\mathcal{A}(x^k)) < \epsilon$ (where ϵ is typically 10^{-6}), $\alpha \le \alpha_{\min}$ and

$$(|f(x^k) - F(x^k, U^k, p)|)/(1 + |f(x^k)|) < \epsilon\,.$$

4. NUMERICAL EXAMPLES

Here we present results of our numerical experiences for the static output feedback problems of *COMPl$_e$ib* . The link between *COMPl$_e$ib* and PENBMI was provided by the MATLAB parser YALMIP 3 (Löfberg 2004). All tests were performed on a 2.5 GHz Pentium with 1 GB RDRAM under Linux. Tables 1 and 2 show the results of PENBMI for $\mathcal{H}_2$-BMI and $\mathcal{H}_\infty$-BMI problems. The results can be divided into seven groups: The first group consists of examples solved without any difficulties. The second and third group contain all cases, for which we had to relax our stopping criterion. These examples are marked by "a" in the tables below, if the achieved precision is still close to our predefined stopping criterion, and by "A", if the deviation is significant. Then there are examples, for which we could calculate almost feasible solutions, but which failed to satisfy the Hurwitz-criterion, namely AC5 and NN10. The fourth group consists of medium and small scale cases for which PENBMI failed, due to ill conditioned Hessian of F—the Cholesky algorithm used for its factorization did not deliver accurate solution and the Newton method failed. In the $\mathcal{H}_2$-setting these are AC7, AC9, AC13, AC18, JE1, JE2, JE3, REA4, DIS5, WEC1, WEC2, WEC3, UWV, PAS, NN1, NN3, NN5, NN6, NN7, NN9, NN12 and NN17, in the $\mathcal{H}_\infty$-setting JE1, JE2, JE3, REA4, DIS5, UWV, PAS, TF3, NN1, NN3, NN5, NN6, NN7 and NN13. The cases in the sixth group are large scale, ill conditioned problems, where PENBMI ran out of time (AC10, AC14, CSE2, EB5). Finally, for very large test cases our code runs out of memory (HS1, BDT2, EB6, TL, CDP, NN18). Only the cases of the first three groups are listed in the tables.

REFERENCES

Apkarian, P. and H. D. Tuan (1999). Concave programming in control theory. *J. Global Optimization* **15**, 343–370.

Beran, E., L. Vandenberghe and S. Boyd (1997). A global BMI algorithm based on the generalized benders decomposition. In: *Proc. of the European Control Conf., Brussels, Belgium.*

Blondel, V. D. and J. N. Tsitsiklis (2000). A survey of computational complexity results in systems and control. *Automatica* **36(9)**, 1249–1274.

Boyd, S., L. El Ghaoui, E. Feron and V. Balakrishnan (1994). *Linear matrix inequalities in system and control theory.* SIAM, Philadelphia, PA.

El Ghaoui, L., F. Oustry and M. Ait-Rami (1997). A cone complementarity linearization algorithm for static output-feedback and related problems. *IEEE Transactions on Automatic Control* **42**, 1171–1176.

Goh, K. C., M. G. Safonov and G. P. Papavassilopoulos (1995). Global optimization for the biaffine matrix inequality problem. *Journal of Global Optimization* **7**, 365–380.

Jarre, F. (2000). An interior method for nonconvex semidefinite programs. *Optimization and Engineering* **1**, 347–372.

Kočvara, M. and M. Stingl (2003). PENNON—a code for convex nonlinear and semidefi-

Table 1. Results of PENBMI on $\mathcal{H}_2$-BMI problems

Ex.	CPU (sec)	n	m	n_x	n_y	n_u	n_w	n_z	maximal real EV of $A(F)$	$\mathcal{H}_2$-perf	prec
AC1	7.900e-01	27	17	5	3	3	3	2	-2.6507e-01	1.0070e-03	
AC2	1.673e+00	39	20	5	3	3	3	5	-2.0115e-07	5.0407e-02	
AC3	7.254e-01	38	20	5	4	2	5	5	-4.5331e-01	4.5704e+00	
AC4	7.998e-01	15	14	4	2	1	2	2	-5.0000e-02	1.1027e+01	
AC6	3.262e+00	64	28	7	4	2	7	7	-8.7223e-01	3.7982e+00	
AC8	8.124e+00	53	29	9	5	1	10	2	-3.1404e-01	1.1935e+00	
AC11	3.544e+00	38	20	5	4	2	5	5	-3.2322e-01	3.9417e+00	
AC12	1.200e+00	23	13	4	4	3	3	1	-4.4183e-01	2.4236e-04	a
AC15	5.395e-01	37	18	4	3	2	4	6	-3.4603e-01	1.2612e+01	
AC16	2.662e+00	39	18	4	4	2	4	6	-3.1996e-01	1.2298e+01	
AC17	3.572e-01	22	16	4	2	1	4	4	-7.1900e-01	4.1096e+00	
HE1	2.225e-01	15	14	4	1	2	2	2	-1.2101e-01	9.5462e-02	
HE2	7.609e-01	24	16	4	2	2	4	4	-3.1776e-01	3.4343e+00	
HE3	1.956e+01	115	34	8	6	4	1	10	-1.4347e-01	8.1179e-01	
HE4	2.127e+01	138	36	8	6	4	8	12	-9.2845e-02	2.0823e+01	
HE5	8.666e+00	54	28	8	2	4	3	4	-9.8307e-03	5.4382e+00	
HE6	1.101e+03	370	76	20	6	4	6	16	-5.0000e-03	6.3174e+01	a
HE7	5.135e+03	370	76	20	6	4	9	16	-5.0000e-03	6.3718e+01	A
REA1	9.565e-01	26	16	4	3	2	4	4	-1.6809e+00	1.8204e+00	
REA2	5.543e-01	24	16	4	2	2	4	4	-1.2287e+00	1.8615e+00	
REA3	2.629e+01	159	48	12	3	1	12	12	-2.0658e-02	1.2087e+01	
DIS1	6.456e+00	88	32	8	4	4	1	8	-4.3379e-01	2.6601e+00	a
DIS2	2.351e-01	16	12	3	2	2	3	3	-7.5830e-01	1.4160e+00	
DIS3	2.478e+00	58	24	6	4	4	6	6	-1.4038e+00	1.8394e+00	
DIS4	3.736e+00	66	24	6	6	4	6	6	-1.0179e+00	1.6924e+00	
TG1	1.071e+02	114	40	10	2	2	10	10	-3.3982e-01	2.2312e+01	a
AGS	1.675e+02	160	48	12	2	2	12	12	-2.0302e-01	6.9954e+00	
BDT1	3.566e+00	96	39	11	3	3	1	6	-1.8860e-03	1.7445e-02	
MFP	5.920e-01	26	16	4	2	3	4	4	-3.2088e-02	9.1620e+00	
IH	3.760e+02	407	74	21	10	11	21	11	-4.7990e-01	8.2603e-04	
CSE1	5.096e+01	308	72	20	10	2	1	12	-5.2977e-02	1.2083e-02	
EB1	2.072e+01	59	32	10	1	1	2	2	-5.5283e-02	1.7000e+00	
EB2	2.691e+01	59	32	10	1	1	2	2	-8.6896e-02	7.7356e-01	
EB3	8.926e+00	59	32	10	1	1	2	2	-4.7092e-02	8.3445e-01	
EB4	4.992e+02	214	62	20	1	1	2	2	-1.7161e-05	5.0428e+02	A
TF1	1.655e+00	46	25	7	4	2	1	4	-6.8809e-02	1.8255e-01	
TF2	3.769e+01	44	25	7	3	2	1	4	-1.0000e-05	1.4491e-01	A
TF3	2.612e+00	44	25	7	3	2	1	4	-3.2000e-03	2.7805e-01	
PSM	2.571e+00	49	26	7	3	2	2	5	-7.8437e-01	1.5043e+00	
NN2	2.815e-01	7	8	2	1	1	2	2	-4.0825e-01	1.5651e+00	
NN4	3.837e-01	26	16	4	3	2	4	4	-5.8731e-01	1.8752e+00	
NN8	3.406e-01	16	12	3	2	2	3	3	-4.6487e-01	2.2797e+00	
NN11	2.856e+02	157	51	16	5	3	3	3	-3.4506e-01	1.1982e+02	A
NN13	3.006e+00	31	21	6	2	2	3	3	-2.3798e+00	2.6217e+01	
NN14	3.484e+00	31	21	6	2	2	3	3	-1.8389e+00	3.5361e+01	
NN15	3.579e-01	20	13	3	2	2	1	4	-1.1779e-01	4.9028e-02	
NN16	4.657e+01	62	28	8	4	4	8	4	-8.0625e-06	3.0732e-01	A

nite programming. *Optimization Methods and Software* **18(3)**, 317–333.

Leibfritz, F. (2003). *COMPl$_e$ib*: *CO*nstrained *M*atrix–optimization *P*roblem *lib*rary – a collection of test examples for nonlinear semidefinite programs, control system design and related problems. Technical report. University of Trier, Dept. of Mathematics.

Leibfritz, F. and E. M. E. Mostafa (2002). An interior point constrained trust region method for a special class of nonlinear semidefinite programming problems. *SIAM Journal on Optimization* **12(4)**, 1048–1074.

Leibfritz, F. and W. Lipinski (2003). Description of the benchmark examples in *COMPl$_e$ib* 1.0. Technical report. University of Trier, Dept. of Mathematics.

Löfberg, J. (2004). Yalmip : A toolbox for modeling and optimization in MATLAB. In: *Proceedings of the CACSD Conference*. Taipei, Taiwan.

Polyak, R. (1992). Modified barrier functions: Theory and methods. *Mathenatical Programming* **54**, 177–222.

Stingl, M. (2004). On the solution of nonlinear Semidefinite Programs by Augmented Lagrangian Methods. PhD thesis. Inst. of Appl. Math., University of Erlangen.

Table 2. Results of PENBMI on $\mathcal{H}_\infty$-BMI problems

Ex.	CPU (sec)	n	m	n_x	n_y	n_u	n_w	n_z	maximal real EV of $A(F)$	$\mathcal{H}_\infty$-perf	prec
AC1	3.781e-01	25	16	5	3	3	3	2	-2.0268e-01	2.5047e-06	
AC2	5.779e-01	25	19	5	3	3	3	5	-2.2644e-07	1.1149e-01	
AC3	4.965e+00	24	21	5	4	2	5	5	-4.1184e-01	3.4017e+00	
AC4	8.211e-01	13	13	4	2	1	2	2	-5.0000e-02	9.3547e-01	
AC6	1.289e+00	37	29	7	4	2	7	7	-7.6361e-01	4.1140e+00	
AC7	6.083e+00	48	24	9	2	1	4	1	-1.8002e-02	2.0969e+00	a
AC8	6.634e+00	51	31	9	5	1	10	2	-3.5472e-01	2.3667e+00	a
AC9	4.242e+01	76	33	10	5	4	10	2	-1.6241e-01	1.0392e+00	
AC11	8.008e+00	24	21	5	4	2	5	5	-4.3284e+00	2.8203e+00	
AC12	1.596e+00	23	13	4	4	3	3	1	-1.1293e-01	3.9777e-01	a
AC13	6.270e+03	419	113	28	4	3	28	28	-2.1340e-02	9.4376e+02	A
AC15	1.971e-01	17	19	4	3	2	4	6	-4.5109e-01	1.5169e+01	
AC16	1.084e+00	19	19	4	4	2	4	6	-9.1532e-01	1.4856e+01	
AC17	1.331e-01	13	17	4	2	1	4	4	-7.2570e-01	6.6124e+00	
AC18	5.677e+01	60	29	10	2	2	3	5	1.3308e+04	4.4104e+02	A
HE1	6.329e-01	13	13	4	1	2	2	2	-1.2883e-01	1.5382e-01	a
HE2	1.667e-01	15	17	4	2	2	4	4	-4.0333e-01	4.2492e+00	
HE3	4.325e+00	61	28	8	6	4	1	10	-2.2199e-01	9.5002e-01	
HE4	1.848e+01	61	37	8	6	4	8	12	-6.7570e-02	2.2838e+01	
HE5	4.912e+00	45	24	8	2	4	3	4	-1.2566e-01	8.8952e+00	
HE6	1.066e+03	235	63	20	6	4	6	16	-5.0000e-03	9.7121e+02	A
HE7	1.096e+03	235	66	20	6	4	9	16	-5.0000e-03	1.3568e+03	A
REA1	9.837e-01	17	17	4	3	2	4	4	-2.0278e+00	8.6571e-01	
REA2	3.007e+00	15	17	4	2	2	4	4	-2.6309e+00	1.1489e+00	
REA3	2.853e+00	82	49	12	3	1	12	12	-2.0658e-02	7.4251e+01	a
DIS1	1.023e+01	53	26	8	4	4	1	8	-7.1484e-01	4.1607e+00	
DIS2	3.028e-01	11	13	3	2	2	3	3	-9.9539e-01	1.0548e+00	
DIS3	1.203e+01	38	25	6	4	4	6	6	-1.3096e+00	1.0649e+00	
DIS4	2.839e+00	46	25	6	6	4	6	6	-1.4542e+00	7.3178e-01	
TG1	3.847e+00	60	41	10	2	2	10	10	-3.2765e-01	1.2846e+01	a
AGS	4.181e+00	83	49	12	2	2	12	12	-2.0663e-01	8.1732e+00	
WEC1	1.030e+01	68	41	10	4	3	10	10	-8.0848e-01	4.0500e+00	a
WEC2	3.393e+01	68	41	10	4	3	10	10	-1.1870e+00	4.2450e+00	a
WEC3	1.091e+01	68	41	10	4	3	10	10	-1.1409e+00	4.4496e+00	a
BDT1	5.318e+00	76	30	11	3	3	1	6	-3.3061e-03	2.6623e-01	
MFP	5.497e-01	17	17	4	2	3	4	4	-3.6371e-02	3.1590e+01	a
IH	3.262e+02	342	75	21	10	11	21	11	-2.1461e-01	4.1873e-02	
CSE1	3.605e+01	231	54	20	10	2	1	12	-9.2244e-02	1.9881e-02	
EB1	1.838e+00	57	25	10	1	1	2	2	-5.6132e-02	3.1225e+00	
EB2	2.296e+00	57	25	10	1	1	2	2	-7.8340e-02	2.0201e+00	
EB3	1.899e+00	57	25	10	1	1	2	2	-3.9478e-02	2.0575e+00	
EB4	7.481e+01	212	45	20	1	1	2	2	-2.0088e-07	2.0564e+00	
TF1	5.033e+00	37	20	7	4	2	1	4	-6.5673e-02	4.0416e-01	A
TF2	7.771e+00	35	20	7	3	2	1	4	-1.0000e-05	2.5560e-01	
PSM	7.475e-01	35	22	7	3	2	2	5	-1.0028e+00	9.2024e-01	
NN2	8.331e-02	5	9	2	1	1	2	2	-6.3576e-01	2.2216e+00	
NN4	9.095e-01	17	17	4	3	2	4	4	-9.4233e-01	1.3591e+00	
NN8	1.850e+00	11	13	3	2	2	3	3	-1.3908e+00	2.8854e+00	
NN9	8.568e+00	22	17	5	2	3	2	4	-3.4983e-01	2.4999e+01	A
NN11	2.681e+02	152	39	16	5	3	3	3	-5.4317e-01	1.3594e-01	A
NN12	5.759e+00	26	25	6	2	2	6	6	-1.6870e-01	1.6294e+01	
NN14	1.416e+00	26	19	6	2	2	3	3	-2.3004e+00	1.7476e+01	a
NN15	3.548e-01	11	12	3	2	2	1	4	-9.2031e-01	9.8090e-02	
NN16	1.644e+00	53	29	8	4	4	8	4	-7.7992e-05	9.5559e-01	
NN17	3.669e-01	9	10	3	1	2	1	2	-4.3593e-01	1.1218e+01	