disturbances; however, dynamic SMC accommodates unmatched disturbances. Future research will involve the chattering elimination and discrete realization of the designed SMC's as well. The developed approach will be extended to a nonminimal phase nonlinear output tracking. Preliminary results look promising.

REFERENCES

[1] V. I. Utkin, *Sliding Modes in Control Optimization.* Berlin, Germany: Springer-Verlag, 1992.

[2] R. DeCarlo, S. H. Zak, and G. P. Matthews, "Variable structure control of nonlinear multivariable systems: A tutorial," *Proc. IEEE,* vol. 76, pp. 212–232, 1988.

[3] G. C. Vergese, B. Fernandez, and J. K. Hedrick, "Stable, robust tracking by sliding mode control," *Syst. Contr. Lett.,* vol. 10, pp. 27–34, 1988.

[4] H. Elmali and N. Olgac, "Robust output tracking control of nonlinear MIMO systems via sliding mode technique," *Automatica,* vol. 28, no. 1, pp. 145–151, 1992.

[5] S. L. Campbell and C. D. Meyer, *Generalized Inverses of Linear Transformations.* New York: Dover, 1991.

[6] Y. Shtessel, "Nonlinear sliding manifolds in nonlinear output tracking problem," in *Proc. Amer. Contr. Conf.,* Seattle, WA, June 21–23, 1995, pp. 1026–1027.

[7] P. V. Kokotovic and H. J. Sussmann, "A positive real condition for global stabilization of nonlinear systems," *Syst. Contr. Lett.,* vol. 13, pp. 125–133, 1989.

[8] H. Sira-Ramires, "A dynamical variable structure control strategy in asymptotic output tracking problem," *IEEE Trans. Automat. Contr.,* vol. 38, pp. 615–620, 1993.

[9] K. D. Young and Ü. Özgüner, "Frequency shaping compensator design for sliding mode," *Int. J. Contr.,* vol. 57, pp. 1005–1019, 1993.

The Real Structured Singular Value is Hardly Approximable

Minyue Fu

***Abstract*—This paper investigates the problem of approximating the real structured singular value (real μ). A negative result is provided which shows that the problem of checking if $\mu = 0$ is NP-hard. This result is much more negative than the known NP-hard result for the problem of checking if $\mu < 1$. One implication of our result is that μ is hardly approximable in the following sense: there does not exist an algorithm, polynomial in the size n of the μ problem, which can produce an upper bound $\overline{\mu}$ for μ with the guarantee that $\mu \le \overline{\mu} \le K(n)\mu$ for any $K(n) > 0$ (even exponential functions of n), unless NP = P. A similar statement holds for the lower bound of μ. Our result strengthens a recent result by Toker, which demonstrates that obtaining a sublinear approximation for μ is NP-hard.**

***Index Terms*—Computational complexity, robust stability, robustness, structured singular value.**

I. MAIN RESULTS

The problem of real structured singular value (real μ) arises in many robust control problems where the control system is subject to uncertain parameters; see, e.g., [2], [3], and [9]–[11] for motivations and references.

Manuscript received May 1, 1996; revised September 5, 1996. This work was supported by the Australian Research Council.

The author is with the Department of Electrical and Computer Engineering, The University of Newcastle, Newcastle, Callaghan, NSW 2308, Australia.

Publisher Item Identifier S 0018-9286(97)05959-X.

Given a matrix $M \in \mathbf{C}^{n\times n}$ and a set $\mathbf{\Delta}$ described by

$$\mathbf{\Delta} = \{\Delta = \operatorname{diag}\{\delta_1 I_{k_1}, \cdots, \delta_m I_{k_m}\} | \delta_i \in \mathbf{R}\}$$
$$k_i > 0, \sum_{i=1}^{m} k_i = n \tag{1}$$

the real μ problem is to compute the value of $\mu_{\mathbf{\Delta}}(M)$. This value is defined to be zero if $I_n - \Delta M$ is nonsingular for all $\Delta \in \mathbf{\Delta}$, or otherwise

$$\mu_{\mathbf{\Delta}}(M) = (\inf\{\rho > 0 | \det(I_n - \Delta M) = 0 \quad \Delta \in \rho B(\mathbf{\Delta})\})^{-1} \tag{2}$$

where

$$B(\mathbf{\Delta}) = \{\Delta \in \mathbf{\Delta} | \delta_i \in [-1, 1], i = 1, \cdots, m\}. \tag{3}$$

For notational simplicity, we denote $\mu_{\mathbf{\Delta}}(M)$ by μ which now has double meanings (the μ problem and the μ value). Note that n is the data size of the problem.

It is known that the μ problem is NP-hard; see Poljak and Rohn [4], Braatz *et al.* [1], and Coxson and DeMarco [2]. More specifically, these papers show that checking if $\mu < 1$ is NP-hard. This negative result means that finding an algorithm for computing μ is very unlikely if the algorithm is required to run in polynomial time. See the Appendix for a brief explanation on computational complexity.

The next logical step is to see how we can approximate μ. We want to know how hard it is to obtain a good approximation for μ. Put in another way, the question is how good an estimate of μ can be obtained using a polynomial algorithm (polynomial in n). To this end, a result in Coxson and DeMarco [2] shows that approximation of μ with arbitrarily small relative error is also an NP-hard problem, following a well-known result on the inapproximability of the so-called maxcut problem. Recently, Toker [5] offers a more negative answer. Toker's result shows that computing an upper bound $\overline{\mu}$ with the guarantee that $\mu \le \overline{\mu} \le Cn^{1-\epsilon}\mu(M)$ for some (very large) constant $C > 0$ and (very small) $\varepsilon > 0$ is an NP-hard problem. This implies that any polynomial time algorithm for computing $\overline{\mu}$ must yield a relative gap between $\overline{\mu}$ and μ at least $O(n^{1-\varepsilon})$, unless P = NP (which is a very unlikely event; see [7]). Note that these negative results refer to the worst case.

The purpose of this paper is to strengthen the negative results above. Our main result is simply stated as follows (with the proof deferred to Section II).

Theorem I.1: The problem of determining if $\mu = 0$ is NP-hard.

This result, while looking similar to Poljak and Rohn [4], Braatz *et al.* [1], and Coxson and DeMarco [2], is actually a much more negative one. We show that the implication of this result is that μ is hardly approximable. The precise result is given below (see Section II for proof).

Theorem I.2: Assume P $\neq$ NP. Then, there does not exist a polynomial algorithm which can produce an upper bound $\overline{\mu}$ for μ with the guarantee that $\mu \le \overline{\mu} \le \overline{K}(n)\mu$ for any (arbitrarily large) $\overline{K}(n) > 0$. Similarly, no polynomial algorithm exists which can produce a lower bound $\underline{\mu}$ for μ with the guarantee that $\underline{K}(n)\mu \le \underline{\mu} \le \mu$ for any (arbitrarily small) $\underline{K}(n) > 0$.

The result above certainly strengthens that by Toker because in [5] $K(n) = n^{1-\varepsilon}$, while in our case, $K(n)$ is allowed to be, for example, an exponential function of n. A similar comment applies to the lower bound approximation.

The rest of this paper is devoted to the proof of Theorems I.1 and I.2.

II. Proof of Theorems 1 and 2

As in almost all NP-hardness analysis cases, our basic idea is to polynomially transform a known NP-complete problem to the problem of determining if $\mu = 0$. Polynomial transformation means that the resulting problem is obtained in polynomial time, and the size of the resulting problem is polynomial of the size of the original problem. We will use the following NP-complete problem.

Problem I–3-SAT Problem: The instance of this problem consists of n, the number of Boolean variables $x = (x_1, x_2, \cdots, x_n)$, and a CNF formula $F(x) = C_1 \wedge C_2 \wedge \cdots \wedge C_m$, with each clause $C_i = z_{i_1} \vee z_{i_2} \vee z_{i_3}$ with z_{i_j} being either one of the x_k or its Boolean negations. For example, $C_2 = x_3 \vee \overline{x}_6 \vee x_8$. The decision is whether or not there exists a Boolean assignment for x such that $F(x)$ is satisfied, i.e., $F(x)$ is true. Since each clause contains three variables (or their negations), this problem is called a 3-SATISFIABILITY problem, or 3-SAT, for short. Note that the number of clauses m is at most C_{2n}^3 (i.e., $2n$-choose-3), thus polynomial in n.

It is well-known that the 3-SAT problem is NP-complete; see, e.g., Papadimitroiu and Steiglitz [8].

The polynomial transformation from the 3-SAT problem to the $\mu = 0$ problem takes two steps. First, we transform it to a problem of testing if a specially constructed multivariate polynomial has a zero solution. We then show that the latter problem can be transformed into a $\mu = 0$ problem. In the sequel, we assume that a Boolean variable takes values zero (for "false") or one (for "true").

Problem II—Multivariate Polynomial Problem: The instance of this problem is the same as the 3-SAT problem with the restriction that $F(0)$ is not satisfied. For each clause C_i of a given instance, we convert each negated variable x_k to $1 - x_k$ but leave the other variables alone. We then define y_i to be their sum. For example, for $C_2 = x_3 \vee \overline{x}_6 \vee x_8$, the corresponding $y_2 = x_3 + (1 - x_6) + x_8$. Then define the following multivariate polynomial:

$$f_0(x) = f_1(x) + f_2(x) \tag{4}$$

where

$$f_1(x) = \sum_{i=1}^{n} x_i^2 (x_i - 1)^2 \tag{5}$$

and

$$f_2(x) = \sum_{i=1}^{m} (1 - y_i)^2 \left(1 - \frac{y_i}{2}\right)^2 \left(1 - \frac{y_i}{3}\right)^2. \tag{6}$$

Then normalize $f_0(x)$ to give

$$f(x) = f_0(x)/f_0(0). \tag{7}$$

Note that $f_0(0) \neq 0$ is guaranteed when $F(0)$ is not satisfied (see proof of the lemma below). The corresponding decision problem is to find if there exists a rational $x \in \mathcal{R}^n$ such that $f(x) = 0$ for each given instance.

We have the following result.

Lemma II.1: Problem II defined above is NP-complete.

Proof: First, it is obvious that the 3-SAT problem with the restriction that $F(0)$ is not satisfied is still NP-complete because checking if $F(0)$ is satisfied takes only polynomial time. Secondly, it follows from the construction of $f_0(\cdot)$ that, for each given instance, $f_0(x) = 0$ if and only if all x_i takes either zero or one and $f_2(x) = 0$. Observe that, if all x_i takes either zero or one, then each y_i takes only zero, one, two, or three. The corresponding clause C_i is satisfied if and only if $y_i = 1, 2,$ or 3. That is, C_i is satisfied if and only if $(1 - y_i)(1 - y_i/2)(1 - y_i/3) = 0$. Subsequently, $f_0(x) = 0$ has a solution if and only if $F(x)$ is satisfied for some assignment x. The assumption that $F(0)$ is not satisfied implies that $f_0(0) \neq 0$. So $f(x)$ is well defined. Since the 3-SAT problem is NP-complete, so is Problem II.

Lemma II.2: Consider any multivariate polynomial $f(x)$: $\mathcal{R}^n \to \mathcal{R}$ with degree l with $f(0) \neq 0$. There exists a polynomial $p(n)$ (denoted by p only), a matrix $M \in \mathcal{R}^{np \times np}$ such that

$$f(x) = \det(I - \Delta M) \tag{8}$$

where

$$\Delta = \operatorname{diag}\{x_1 I_p, x_2 I_p, \cdots, x_n I_p\}. \tag{9}$$

Further, such an M can be constructed in polynomial time.

Proof: We simply give a procedure for constructing such an M and show that its size is polynomial in n and that it takes only polynomial time to construct it.

Procedure:

Step 1) Set $p = 1$, $A_p(x) = f(x)$ (a matrix function of size $p \times p$).

Step 2) If all entries of $A_p(x)$ have degree less than or equal to one, go to Step 4).

Step 3) Take any entry $a_{ij}(x)$ of $A_p(x)$ with degree higher than one. Denote its highest order term (or one of them) by $\alpha_{ij}(x) x_k$ for some x_k (it can always be written this way) and the rest of $a_{ij}(x)$ by $\beta_{ij}(x)$. That is

$$a_{ij}(x) = x_k \alpha_{ij}(x) + \beta_{ij}(x). \tag{10}$$

Define $\tilde{A}_p(x)$ to be the matrix function identical to $A_p(x)$ except that the ijth entry is replaced by $\beta_{ij}(x)$. Let $a(x)$ be the p-column vector with ith element equal to $\alpha_{ij}(x)$ and zero elsewhere. Similarly, let $b(x)$ be the p-row vector with jth element equal to x_k and zero elsewhere. Then, define

$$A_{p+1}(x) = \begin{bmatrix} \tilde{A}_p(x) & -a(x) \\ b(x) & 1 \end{bmatrix} \tag{11}$$

and update $p := p + 1$. Return to 2).

Step 4) Since $A_p(x)$ is affine in x now, express $A_p(x) = A_0 + \sum_{i=1}^{n} x_i A_i$. Define

$$M = \begin{bmatrix} A_1 \\ A_2 \\ \cdots \\ A_n \end{bmatrix} [-A_0^{-1} \quad -A_0^{-1} \quad \cdots \quad -A_0^{-1}]. \tag{12}$$

It is simple to see that each step takes a polynomial time and that the procedure terminates in a polynomial number of steps because $f(x)$ has n variables and degree l. So the construction of M takes only polynomial time, and p is a polynomial in n. It remains to show (8). To see this, we note that the update of $A_{p+1}(x)$ in (11) ensures that $\det A_{p+1}(x) = A_p(x)$ due to the following equation:

$$\det \begin{bmatrix} X_{11} & X_{12} \\ X_{21} & I \end{bmatrix} = \det(X_{11} - X_{12} X_{21}).$$

It follows that the final $A_p(x)$ has determinant equal to $f(x)$. In particular, we have $\det A_p(0) = f(0) = 1$. (Recall that $f(x)$ is normalized.) Also note the following equation:

$$\det(I - XY) = \det(I - YX)$$

for any matrices X and Y as long as both XY and YX are square. Taking the matrix M defined in (12) and Δ defined in (9), we have

$$\begin{aligned} \det(I - \Delta M) &= \det\left(I + A_0^{-1} \sum_{i=1}^{n} x_i A_i\right) \\ &= \det A_0^{-1} \det\left(A_0 + \sum_{i=1}^{n} x_i A_i\right) \\ &= \det A_p(x) \\ &= f(x). \end{aligned}$$

■

Proof of Theorem I.1: Lemma II.1 shows that Problem II is NP-complete. It follows from Lemma II.2 that solving Problem II is the same as finding $x \in \mathcal{R}^n$ such that $\det(I - \Delta M) = 0$, where Δ and M are constructed from the given instance of Problem II. From the definition of the μ, we know that solving Problem II is the same as checking if $\mu = 0$ for this particularly constructed μ problem. Hence, the $\mu = 0$ problem is NP-hard.

Proof of Theorem I.2: We first prove the result for $\overline{\mu}$ by contradiction. Suppose there exists $K(n) > 0$ and a polynomial time algorithm which produces a $\overline{\mu}$ with the guarantee that $\mu \leq \overline{\mu} \leq K(n)\mu$. Then, $\mu = 0$ if and only if $\overline{\mu} = 0$. Hence, this algorithm will be able to solve the $\mu = 0$ problem in polynomial time. Using Theorem 1, the above will imply that NP = P, contradicting our assumption that NP $\neq$ P.

The same argument works for the lower bound because $K(n)\mu \leq \underline{\mu} \leq \mu$ with $K(n) > 0$ implies that $\mu = 0$, if and only if $\underline{\mu} = 0$.

III. Conclusion

The negative results in Theorems I.1 and I.2 imply that any algorithm attempting to give a good approximation for the real μ is expected to be exponential time, or equivalently, any polynomial algorithm is expected to behave very badly in the worst case as the problem size grows unless P = NP. However, these negative results should not be viewed as the end of searching for efficient algorithms. For example, we may aim at small-to-medium sized problems. We may simplify the μ problem by exploiting structural information or by reformulating the μ problem and looking for a different robustness measure. Some recent attempts for improving the standard $D - G$ scaling upper bound by Fan *et al.* [11] are given in Fu and Barabanov [3] and Meinsma *et al.* [9]. Further, we may attempt to search for algorithms which are nonpolynomial but perform well "in practice."

In contrast to the real μ problem, the approximability of the complex μ problem is easier. In fact, checking if the complex $\mu < 1$ is proved to be NP-hard [6]. But the D-scaling method [10] for computing an upper bound $\overline{\mu}$ of the complex μ is a polynomial algorithm and is known to give a guaranteed linear approximation [12], i.e., $\overline{\mu} \leq O(n)\mu$ for some linear function $O(n)$. Finally, we note that the 3-SAT problem is not only NP-complete, but *strongly* NP-complete. For this type of problem, it is known that not only do no polynomial algorithms exist there, but also there exist no pseudo-polynomial algorithms either, unless P = NP. The class of pseudo-polynomials is much larger than the class of polynomials, including functions such as $2^{\log^c n}$ for any constant $c > 0$; see details on strong NP-completeness and pseudo polynomial algorithms in [8]. Hence, the results in Theorems I.1 and I.2 can all be strengthened accordingly.

Appendix
Basics of Computational Complexity

The complexity class P denotes a class of decision problems (problems giving binary answers) which can be solved by a deterministic Turing machine in polynomial time. The class NP denotes a class of decision problems which can be solved by a nondeterministic Turing machine in polynomial time, including P as a subclass. The exact definitions of these two classes are involved and can be found in Garey and Johnson [7] and Papadimitroiu and Steiglitz [8]. Roughly speaking, every P problem has a deterministic polynomial time algorithm, and every NP problem has a deterministic exponential time algorithm. Examples of NP problems which are not known to be P include the traveling salesman problem, the maximum cut problem, the 3-SAT problem used in this paper, [7], [8], and the decision problem for μ (i.e, is $\mu > 1$?) [1], [2], [4], [6]. It is generally believed that NP $\neq$ P, although it has been a great challenge in combinatoric optimization for the last several decades to prove or disprove it. A decision problem is called NP-complete if it lies in NP and every NP problem can be transformed in polynomial time into this problem. All the examples above are NP-complete. A problem (decision or nondecision) is called NP-hard if an NP-complete problem can be reduced to this problem in polynomial time. For example, computing μ is known to be NP-hard, in all real, mixed, and complex cases [1], [4], [6]. So, NP-hard problems are at least as "hard" as NP-complete problems, which in turn are the hardest in NP. The term polynomial algorithm means an algorithm which is deterministic and requires only polynomial time and polynomial storage to execute on a Turing machine.

The data (called instance) of an optimization problem are assumed to be rational to avoid the complexity issues for real numbers. However, no assumption is made for the optimal value to be rational because we are only interested in approximating the value. The computational complexity is analyzed with respect to the size of the data (n in the μ problem).

Acknowledgment

The author wishes to thank the anonymous reviewers for their comments on the paper.

References

[1] R. P. Braatz *et al.*, "Computational complexity of μ calculation," *IEEE Trans. Automat. Contr.*, vol. 39, pp. 1000–1002, May 1994.

[2] G. E. Coxson and C. L. DeMarco, "The computational complexity of approximating the minimal perturbation scaling to achieve instability in an interval matrix," *Math. Contr. Signals Syst.*, vol. 7, no. 4, pp. 279–292, 1994.

[3] M. Fu and N. E. Barabanov, "Improved upper bounds for mixed μ," in *Proc. 34th Conf. Decision Contr.*, New Orleans, LA, Dec. 1995.

[4] S. Poljak and J. Rohn, "Checking robust nonsingularity is NP-hard," *Math. Contr. Signals Syst.*, vol. 6, pp. 1–9, 1993.

[5] O. Toker, "On the conservatism of upper bound tests for structured singular value analysis," to be published.

[6] O. Toker and H. Ozbay, "On the NP-hardness of the purely complex μ computation, analysis/synthesis and some related problems in multidimensional systems," to be published.

[7] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to NP-Completeness.* New York: W. H. Freeman, 1983.

[8] C. H. Papadimitriou and K. Steglitz, *Combinatorial Optimization: Algorithms and Complexity.* Englewood Cliffs, NJ: Prentice-Hall, 1982.

[9] G. Meinsma, V. Shrivastava, and M. Fu, "Some properties of an upper bound of μ," *IEEE Trans. Automat. Contr.*, 1995.

[10] J. C. Doyle, "Analysis of feedback systems with structured uncertainties," *Proc. Inst. Elec. Eng.*, vol. 129, pt. D, pp. 240–250, 1982.

[11] M. K. H. Fan, A. L. Tits, and J. C. Doyle, "Robustness in the presence of mixed parametric uncertainty and unmodeled dynamics," *IEEE Trans. Automat. Contr.*, vol. 36, pp. 25–38, Jan. 1991.

[12] A. Megretski, "On the gap between structured singular values and their upper bounds," in *Proc. 32nd Conf. Decision Contr.*, San Antonio, TX, Dec. 1993.