Case 3 (Suboptimal Controller with $d = 4$): In this case, $H(z) = z^3 I$. The suboptimal controller parameters become the equations shown at the bottom of the previous page, from which it follows that $J_4 = 0.0065$. Fig. 4 shows the output responses of the plant with the same measurement noise as in case 1. Fig. 5 shows the input plots in this case.

VI. CONCLUSION

A design method for multivariable EMM systems which reduces the effect of measurement noise on the plant outputs is presented in this paper. This method uses the freedom of the controller parameters due to increasing the degree of the observer polynomial matrix. For some fixed degree of the observer polynomial matrix, the controller for this EMM is obtained as a solution of a simple optimization problem. It is also shown that the effect of the measurement noise in the plant outputs become smaller as the degree of the observer polynomial matrix is increased. This method will be applicable to pole placement control for a multivariable plant with measurement noise, which will be reported in a forthcoming paper [13].

REFERENCES

[1] B. C. Moore and L. M. Silverman, "Model matching by state feedback and dynamic compensation," *IEEE Trans. Automat. Contr.*, vol. AC-17, pp. 491–497, 1972.

[2] W. A. Wolovich, *Linear Multivariable Systems*. Berlin, Germany: Springer-Verlag, 1974.

[3] G. C. Goodwin and K. S. Sin, *Adaptive Filtering, Prediction and Control*. Englewood Cliffs, NJ: Prentice-Hall, 1984.

[4] H. Elliott and W. A. Wolovich, "Parametrazation issues in multivariable adaptive control," *Automatica*, vol. 20, pp. 533–545, 1984.

[5] W. Kase and K. Tamura, "Design of G-interactor and its application to direct multivariable adaptive control," *Int. J. Contr.*, vol. 51, pp. 1067–1088, 1990.

[6] Y. Mutoh and R. Ortega, "Interactor structure estimation for adaptive control of discrete-time multivariable nondecouplable systems," *Automatica*, vol. 29, pp. 635–647, 1993.

[7] Y. Mutoh and P. N. Nikiforuk, "Suboptimal perfect model matching for a plant with measurement noise and its application to MRACS," *IEEE Trans. Automat. Contr.*, vol. 39, pp. 422–425, 1994.

[8] W. A. Wolovich and P. L. Falb, "Invariants and canonical forms under dynamic compensations," *SIAM J. Contr. Optimiz.*, vol. 14, pp. 996–1008, 1976.

[9] M. Das, "Multivariable adaptive model matching using less a priori information," *ASME J. Dynam. Syst. Measurement Contr.*, vol. 108, pp. 151–153, 1986.

[10] W. Kase and Y. Mutoh, "Suboptimal exact model matching for multivariable systems with measurement noise and its application to MRACS," in *Proc. 12th Int. Conf. Systems Engineering*, U.K., 1997, vol. 1, pp. 361–366.

[11] W. Kase, "A solution of polynomial matrix equations using extended division algorithm and another description of all-stabilizing controllers," *Int. J. Syst. Sci.*, vol. 30, pp. 95–104, 1999.

[12] A. Albert, *Regression and the Moore–Penrose Pseudoinverse*. New York, NY: Academic, 1972.

[13] W. Kase, "Suboptimal dead beat controller for a discrete-time multivariable plant with measurement noise," in *Proc. 1999 American Control Conf.*, 1999, vol. 1, pp. 810–814.

Use of CPWL Approximations in the Design of a Numerical Nonlinear Regulator

Mirta S. Padín and José Luis Figueroa

***Abstract*—This paper presents the state regulation problem for nonlinear plants with initial conditions in a given bounded region, proposed originally by Kreisselmeier and Birkholzer [1]. An efficient computer implementable algorithm is presented, based on canonical piecewise lineal approximation for the discrete model of the plant.**

I. INTRODUCTION

A control engineer is often faced with nonlinear systems, and, therefore, design methods that can handle nonlinearities are of great practical interest. Kreisselmeier and Birkholzer [1] present a solution for the following problem: Given a nonlinear plant P and a set of initial conditions G, find a numerical controller for stable state regulation from G, if one exists. A numerical design method is, in essence, a computer-implementable algorithm. The numerical design method proposed has been *well founded theoretically*, such that the feedback controller defined by the algorithm is guaranteed to be a stabilizing one. The particular technique proposed, however, has the disadvantage of requiring excessive computation time. In this paper, we propose a computational method that retains the theoretical foundations of the methodology of [1] in a more efficient algorithm using a canonical piecewise linear (CPWL) approximation for the system.

II. PROBLEM DESCRIPTION

Consider a nonlinear plant P described by

$$\mathbf{x_{t+1}} = f(\mathbf{x_t}, \mathbf{u_t}) \tag{1}$$

with state $\mathbf{x_t} \in \Re^n$, control $\mathbf{u_t} \in \Re^m$ and discrete-time variable $t \in \mathbf{Z}_o^+$. It is assumed that $f: \Re^n \times \Re^m \rightarrow \Re^n$ is continuous. Further, let $G \subset \Re^n$ be any prespecified bounded region in the state space, which contains a neighborhood of the origin.

Given P and G, it is typically unknown whether a feedback controller exists, such that the resulting feedback system is asymptotically stable for all initial conditions $\mathbf{x} \in G$. The objective here is to set up a numerical design method that can handle this situation and compute a stabilizing feedback controller, if one exists.

The design strategy is to compute the optimal cost and an associated feedback controller $(\mathbf{u_t} = k(\mathbf{x_t}))$ *off-line* for all $\mathbf{x}$ in the region of interest. Once computed, the feedback can be stored in the control computer, so that in the actual feedback loop virtually no *on-line* control computations are required. To find the optimal solution for all $\mathbf{x}$ in the region of interest, Kreisselmeier and Birkholzer [1] use the method of *dynamic programming* ([2], [3]).

Kreisselmeier and Birkholzer [1] show that a feedback controller k exists such that the closed-loop system $\mathbf{x_{t+1}} = f(\mathbf{x_t}, k(\mathbf{x_t}))$ is asymptotically stable from G to the origin iff the plant is asymptotically controllable from G to the origin. Moreover, under this condition,

Manuscript received September 21, 1999. Recommended by Associate Editor, Q. Zhang.

M. S. Padin is with the Centro de Investigación, Unidad Académica Río Gallegos, Universidad Nacional Patagonia Austral, Rio Gallegos, 9400 Argentina (e-mail: mspadin@millennium.com.ar).

J. L. Figueroa is with PLAP/QUI-UNS—CON/CET and with the Departamento de Ingeniería Eléctrica, Universidad Nacional del Sur, Bahía Blanca, 8000 Argentina (e-mail: cofiguer@criba.edu.ar).

Publisher Item Identifier S 0018-9286(00)04221-5.

a continuous, positive definite function $l(\mathbf{x}, \mathbf{u})$ exists such that for all $\mathbf{x} \in G$, the right-hand side of

$$J(\mathbf{x}) = \inf_{\underline{\mathbf{u}}} \sum_{t=0}^{\infty} l(\phi_t(\mathbf{x}, \underline{\mathbf{u}}), \mathbf{u_t}) \tag{2}$$

is well defined and assumes values below some finite constant $\gamma_o = \sup_{\mathbf{x} \in G} J(\mathbf{x})$, where $\phi_t(\mathbf{x}, \underline{\mathbf{u}})$ is the solution of (1) with initial state $\mathbf{x}$ and control sequence $\underline{\mathbf{u}} = \{\mathbf{u}_0, \mathbf{u}_1, \mathbf{u}_2, \cdots,\}$. Let Γ_o denote the set of points in $\Re^n$ such that the right-hand side of (2) exists and satisfies $J(\mathbf{x}) \leq \gamma_o$. Moreover $J(\mathbf{x})$ is positive and decreasing as $\mathbf{x} \to 0$ in Γ_o. Then, for all $\mathbf{x} \in \Gamma_o$ the function $J(\mathbf{x})$ satisfies the equation

$$J(\mathbf{x}) = \inf_{\mathbf{u}} [l(\mathbf{x}, \mathbf{u}) + J(f(\mathbf{x}, \mathbf{u}))]. \tag{3}$$

Since $l(\mathbf{x}, \mathbf{u})$ is positive definite for $\mathbf{x} \neq 0$, the existence of $k(\mathbf{x})$ implies that $J(\mathbf{x})$ is a Lyapunov function in Γ_o and $k(\mathbf{x})$ is an asymptotically stabilizing control from G to the origin. Setting the iteration

$$J_{T+1}(\mathbf{x}) = \inf_{\mathbf{u}} [l(\mathbf{x}, \mathbf{u}) + J_T(f(\mathbf{x}, \mathbf{u}))], \quad J_0(\mathbf{x}) = 0, \quad T \in \mathbf{Z_o^+} \tag{4}$$

which is of the dynamic programming type, it follows that $J_T(\mathbf{x})$ converges to $J(\mathbf{x})$ in Γ_o as $T \to \infty$.

Now, to obtain a finite implementable algorithm, there are some basic subproblems involved in the solution of this problem. They are associated with the computing region, with the parameterization of the controllers and with the systematic errors in the computations. To account for these subproblems, [1] performs some modifications to the problem. First, a *cost criterion with terminal cost and free terminal time* is used. Second, a *parameterized controller that is piecewise-constant in the state space* is used. Third, a *practical asymptotical stability* is used, which can be made arbitrarily close to asymptotic stability. Let us define this concept.

Definition 1: A feedback system $\mathbf{x_{t+1}} = f(\mathbf{x_t}, k(\mathbf{x_t}))$ with trajectories $\phi_t^c(\mathbf{x})$ is said to be *practically asintotically stable* from G to $N(r_o)$, if a function $\sigma(r, t) \in K_\sigma$ exists such that for all $\mathbf{x} \in G$

$$\|\phi_t^c(\mathbf{x})\| + \|k(\phi_t^c(\mathbf{x}))\| \leq \max\{r_o, \sigma(\|\mathbf{x}\|, t)\}, \qquad t \in \mathbf{Z_o^+} \tag{5}$$

where $N(r_o)$ is a neighborhood of the origin with radius r_0 and the definition of the set K_σ is in Appendix A.

As a candidate of a practical control Lyapunov function, i.e., a control Lyapunov function for practical asymptotic stability, we define the extended cost criterion as

$$V(\mathbf{x}) = \inf_{\underline{\mathbf{u}}, t' \in \mathbf{Z_o^+}} \left[\sum_{t=0}^{t'-1} l(\phi_t(\mathbf{x}, \underline{\mathbf{u}}), \mathbf{u}_t) + \overline{V}(\phi_{t'}(\mathbf{x}, \underline{\mathbf{u}})) \right]. \tag{6}$$

It differs from the infinite-time horizon criterion $J(\mathbf{x})$ because it considers the cost over all possible finite time horizons with terminal cost $\overline{V}$ and takes the infimum of those. Throughout this section, $l(\mathbf{x}, \mathbf{u})$ and $\overline{V}(\mathbf{x})$ are continuous, positive-definite and satisfy $\underline{\varphi}_l(\|\mathbf{u}\|) + \underline{\varphi}_l(\|\mathbf{x}\|) \leq l(\mathbf{x}, \mathbf{u}) \leq \overline{\varphi}_l(\|\mathbf{u}\|) + \overline{\varphi}_l(\|\mathbf{x}\|)$ and $\underline{\varphi}_{\overline{V}}(\|\mathbf{x}\|) \leq \overline{V}(\mathbf{x}) \leq \overline{\varphi}_{\overline{V}}(\|\mathbf{x}\|)$ for appropriate functions $\underline{\varphi}_l, \underline{\varphi}_{\overline{V}}, \overline{\varphi}_l, \overline{\varphi}_{\overline{V}} \in K_\varphi$,[1] for all $\mathbf{x} \in \Re^n$ and $\mathbf{u} \in \Re^m$. Also, a bounded region $\overline{\Gamma} \subset \Re^n$ is defined such that $l(\mathbf{x}, \mathbf{u}) > \overline{V}(\mathbf{x})$ for all $\mathbf{x} \in \overline{\Gamma}$ and $\mathbf{u} \in \Re^m$. From these definitions, it is possible to show ([1]) that only a bounded set of controllers $\mathbf{u} \in \overline{U}$ needs to be considered. Then, it is possible to write the following feedback theorem.

Theorem 1: Let $V(\mathbf{x})$ be defined by (6), and suppose that

$$V(\mathbf{x}) < \overline{V}(\mathbf{x}) \quad \forall x \in \Gamma - N(\rho) \tag{7}$$

holds for a sufficiently small $\rho = \rho(r_o)$. Further, let $k(\mathbf{x})$ be an associate feedback satisfying

$$l(\mathbf{x}, k(\mathbf{x})) + V(f(\mathbf{x}, k(\mathbf{x}))) - \kappa \leq \inf_{\underline{\mathbf{u}} \in \overline{U}} [l(\mathbf{x}, \mathbf{u}) + V(f(\mathbf{x}, \mathbf{u}))] \tag{8}$$

with κ lower than the minimum of a function of the class K_φ varying with ρ. Then, the feedback system $\mathbf{x_{t+1}} = f(\mathbf{x_t}, k(\mathbf{x_t}))$ is practically asymptotically stable from G to $N(r_0)$.

Theorem 1 says that $V(\mathbf{x})$ is a practical control Lyapunov function, provided that $V(\mathbf{x})$ is lower than $\overline{V}(\mathbf{x})$ in a suitable region. This requirement is related to the fact that the extended performance criterion includes the minimization over a sliding time horizon. If $V < \overline{V}$ at a certain point, then the time horizons greater than zero are effective at this point, and, therefore, V contains descent information. If this information is provided in a region that contains Γ, except possibly a small enough neighborhood of the origin $N(\rho)$, then a controller can be deduced from $V(\mathbf{x})$, which is practically asymptotically stabilizing from G to $N(r_0)$.

The results, as *stated* above, cannot be implemented on a computer without modifications. Therefore, [1] modifies the feedback construction by including state and input discretization, interpolation, and an appropriate feedback parameterization. A discretization (D) for the state-space $\Re^n$ is performed defining a canonical grid of points with a distance d_1 between them and approximating each state $\mathbf{x}$ to the nearest point. A similar discretization (D_2) is performed for the control variables $\mathbf{u}$. Then, let $\overline{\Gamma}_D$ and $\overline{U}_D$ be the discrete versions of the sets $\overline{\Gamma}$ and $\overline{U}$. Since $\overline{\Gamma}$ and $\overline{U}$ are bounded, $\overline{\Gamma}_D$ and $\overline{U}_D$ are finite. To define the *interpolation*, let us consider a function W from the discrete set defined above to $\Re$. Then, it is possible to extend the range of definition to all $\Re^n$ using a multilinear interpolation (a linear interpolation is considered in each dimension, [1]) that we will describe as $I(W, \mathbf{x}) : \Re^n \to \Re$.

Finally, to perform an implementable design algorithm, we set up the following design function:

$$W(\mathbf{x}) = \min\left\{\overline{V}\mathbf{x}, \min_u \left[l(\mathbf{x}, \mathbf{u}) + I\left(W, f(\mathbf{x}, \mathbf{u})\right)\right]\right\} \tag{9}$$

where only the function values of W at the discretization points are involved. Associated with this design equation, we define the design iteration

$$W_0(\mathbf{x}) = \overline{V}(\mathbf{x}) \tag{10}$$

$$W_{T+1}(\mathbf{x}) = \min\left\{W_T(\mathbf{x}), \min_u [l(\mathbf{x}, \mathbf{u}) + I(W_T, f(\mathbf{x}, \mathbf{u}))]\right\} \tag{11}$$

that admits a computer implementable algorithm. Moreover, [1] shows that W_{T+1} converges to a solution of the design equation as $T \to \infty$ and is in some sense close to $V(\mathbf{x})$ for sufficiently large T.

Let $W_T(x)$ be any member of the design iteration sequence, and let an associate (pointwise) feedback h from the state discretization grid to $\Re^m$ be chosen such that $h(z) \in \overline{U}_D$ for all z in the grid and for all $z \in \overline{\Gamma}_D$

$$l(z, h(z)) + I(W_T, f(z, h(z))) = \min_{\mathbf{u} \in \overline{U}_D} [l(z, \mathbf{u}) + I(W_T, f(z, \mathbf{u}))] \tag{12}$$

i.e., such that $\mathbf{u} = h(z)$ is a minimizing argument of the right-hand side in $\overline{\Gamma}_D$. Then, using the discretization D, we obtain a state feedback $h(D(\mathbf{x})) : \Re^n \to \Re^m$, which is piecewise constant in the state-space.

Based on this setup, [1] proposes a design algorithm using dynamic programming. Let $z_1, z_2, \cdots, z_{\overline{n}}$ denote the discrete set of points contained in $\overline{\Gamma}_D$, and $u_1, u_2, \cdots, u_{\overline{m}}$ denote the discrete set of control actions constrained in $\overline{U}_D$. Write from (10), (11), the following directly implementable design algorithm.

[1]The class of functions K_φ is defined in Appendix A.

1) Initialization: For all $i \in [1, \overline{n}]$

$$W_0(z_i) = \overline{V}(z_i). \tag{13}$$

2) Iteration from T to $T+1$: For all $i \in [1, \overline{n}]$

$$W_{T+1}(z_i) = \min\left\{W_T(z_i), \min_{j\in[1,\overline{m}]}[l(z_i, u_j) + I(W_T, f(z_i, u_j))]\right\} \tag{14}$$

After finishing the iteration, say at stage $\overline{T}$, the feedback function $h(\cdot)$ for all $i \in [1, \overline{n}]$ is given by the minimizing argument

$$h(z_i) = \arg\min_{u_j\in[u_1\cdots,u_{\overline{m}}]}[l(z_i, \mathbf{u}_j) + I(W_{\overline{T}}, f(z_i, \mathbf{u}_j))] \tag{15}$$

the computation of which is part of (14).

To terminate the iteration, the criterion used is that the difference $W_{T-1}(z) - W_T(z)$ be smaller than a given tolerance. Note that the iteration can be executed and it will converge (because the sequence $W_T(z)$ is nomotonically decreasing and bounded from below) no matter if the plant is asymptotically controllable. Therefore, asymptotic controllability does not need to be checked *a priori* to using the design algorithm. Moreover, since the sequence $W_T(z)$ is monotonically decreasing and bounded from below, if we obtain an efficient way to solve (14), we can do it for only a sufficiently large horizon T.

In this paper, let us consider the following expressions for the Lyapunov functions

$$\overline{V}(\mathbf{x}, \mathbf{u}) = \mathbf{x}^T\mathbf{Q}_\mathbf{x}^1\mathbf{x} + \mathbf{u}^T\mathbf{Q}_\mathbf{u}^1\mathbf{u} \tag{16}$$

and

$$l(\mathbf{x}, \mathbf{u}, t) = \mathbf{x}^T\mathbf{Q}_\mathbf{x}^t\mathbf{x} + \mathbf{u}^T\mathbf{Q}_\mathbf{u}^t\mathbf{u} \tag{17}$$

where $\mathbf{Q}_\mathbf{x}^t$ and $\mathbf{Q}_\mathbf{u}^t$ are positive definite weighting matrices for all $t = 1, 2, \cdots, T$. Then, taking a large enough horizon time T, the solution of (15) can be obtained by solving the following optimization problem:

$$\min_{\mathbf{u}_1,\mathbf{u}_2,\cdots,\mathbf{u}_T}\sum_{i=1}^{T}\mathbf{x_i}^TQ_\mathbf{x}^\mathbf{i}\mathbf{x_i} + \sum_{i=1}^{T}\mathbf{u_i}^T\mathbf{Q}_\mathbf{u}^\mathbf{i}\mathbf{u_i} \quad \text{s.t.}$$
$$\mathbf{x_{i+1}} = f(\mathbf{x_t}, \mathbf{u_t}) \tag{18}$$

III. Solution to CPWL Problem

In this paper, we consider the use of a CPWL approximation for the nonlinear model (1). These functions allows us to write nonlinear functions as several linear expressions, each of those valid in a certain region. Based on the properties of this representation, we will obtain a practical and efficient approach for the optimization problem. In mathematical terms, this can be described as follows.

Let $X \subset \Re^n$ and $U \subset \Re^m$ be the domains of $\mathbf{x}$ and $\mathbf{u}$ variables, respectively, and consider the set

$$\aleph = \{[\mathbf{x}^T, \mathbf{u}^T]^T : \mathbf{x} \in X, \mathbf{u} \in U\} \tag{19}$$

which is the domain where we want to approximate the system (1). Consider also the following partition in the set $\aleph$ such that $\aleph = \cup_{j=1}^{\sigma}\aleph^j$, where $\aleph^j$ is called the "jth *partition*." Then, the CPWL representation [4], [5] of the system is

$$\mathbf{x_{k+1}} = \mathbf{a_x} + \mathbf{B_{xx}x_k} + \mathbf{B_{xu}u_k} + \sum_{i=1}^{\sigma}\mathbf{c_{xi}}\left|\alpha_\mathbf{xi}\mathbf{x_k} + \alpha_\mathbf{ui}\mathbf{u_k} - \beta_\mathbf{i}\right| \tag{20}$$

where all the matrices and vectors have appropriate dimensions with elements in the real field. If the system is constrained to the jth region (i.e., we constrain $(\mathbf{x}, \mathbf{u} \in \aleph^j)$), the model (20) can be reformulated as a linear function,

$$\mathbf{x_{k+1}} = \xi_\mathbf{xx}^\mathbf{j}\cdot\mathbf{x_k} + \xi_\mathbf{xu}^\mathbf{j}\cdot\mathbf{u_k} + \eta_\mathbf{x}^\mathbf{j} \tag{21}$$

where $\xi_\mathbf{xx}^\mathbf{j} = \mathbf{B_{xx}} + \sum_{i=1}^{\sigma}\mathbf{c_{xi}}\cdot\alpha_\mathbf{xi}\cdot\gamma_i^j$, $\xi_\mathbf{xu}^\mathbf{j} = \mathbf{B_{xu}} + \sum_{i=1}^{\sigma}\mathbf{c_{xi}}\cdot\alpha_\mathbf{ui}\cdot\gamma_i^j$, $\eta_\mathbf{x}^\mathbf{j} = \mathbf{a_x} + \sum_{i=1}^{\sigma}\mathbf{c_{xi}}\cdot\beta_\mathbf{i}\cdot\gamma_i^j$, with $\gamma_i^j = \operatorname{sign}(\alpha_\mathbf{xi}\mathbf{x} + \alpha_\mathbf{ui}\mathbf{u} - \beta_i)$.

The *sign* function in the last expression determines the ***sector belonging condition,***[2] i.e., the sign vector γ^j defined as $\gamma^\mathbf{j} = [\gamma_1^j, \gamma_2^j, \cdots, \gamma_\sigma^j]$ is univocally related to the jth partition ([6]). Consequently, a point $(\mathbf{x}^h, \mathbf{u}^h)$ will lie in $\aleph^j$ iff it satisfies the inequality

$$\mathbf{z}_\aleph^\mathbf{j} = \xi_{\aleph\mathbf{x}}^\mathbf{j}\cdot\mathbf{x^h} + \xi_{\aleph\mathbf{u}}^\mathbf{j}\cdot\mathbf{u^h} + \eta_\aleph^\mathbf{j} \le \mathbf{0} \tag{22}$$

where $[\xi_{\aleph\mathbf{x}}^\mathbf{j}]_i = -\gamma_i^j\alpha_\mathbf{xi}$, $[\xi_{\aleph\mathbf{u}}^\mathbf{j}]_i = -\gamma_i^j\alpha_\mathbf{ui}$, $[\eta_\aleph^j]_i = \gamma_i^j\beta_i$ and $[\cdot]_i$ means the ith row in the matrix $[\cdot]$. The problems associated with the determination of these regions and the CPWL approximations have been extensively studied in the literature (see, e.g., [7]–[9]).

Now, consider the system at any initial condition $(\mathbf{x}_0, \mathbf{u}_0) \in \aleph^0$ that verifies the steady-state equations (i.e., $\mathbf{x}_0 = f(\mathbf{x}_0, \mathbf{u}_0)$). While the system is in sector $\aleph^0$ (suppose that this occurs for n^0 samples), it is easy to see that the state vector will be

$$\mathbf{x_i} = \xi_\mathbf{xx}^0\mathbf{x_{i-1}} + \xi_\mathbf{xu}^0\cdot\mathbf{u_{i-1}} + \eta_\mathbf{x}^0 \quad \forall i = 1, \cdots, n^0. \tag{23}$$

This expression is valid till the moment the system reaches next sector (called $\aleph^1$), when the value of the state is

$$\begin{aligned}\mathbf{x}_{\mathbf{n}^0} = {} & \left(\xi_\mathbf{xx}^0\right)^{n^0}\mathbf{x}_0 + \xi_\mathbf{xu}^0\mathbf{u}_{\mathbf{n}^0-1} + \xi_\mathbf{xx}^0\xi_\mathbf{xu}^0\mathbf{u}_{\mathbf{n}^0-2} \\ & + \left(\xi_\mathbf{xx}^0\right)^2\xi_\mathbf{xu}^0\mathbf{u}_{\mathbf{n}^0-3} + \cdots + \left(\xi_\mathbf{xx}^0\right)^{n^0-1}\xi_\mathbf{xu}^0\mathbf{u}_0 + \eta_\mathbf{x}^0 \\ & + \left(\xi_\mathbf{xx}^0\right)\eta_\mathbf{x}^0 + \cdots + \left(\xi_\mathbf{xx}^0\right)^{n^0-1}\eta_\mathbf{x}^0\end{aligned}$$

and, using this state as an initial condition for sector $\aleph^1$, it is possible to compute

$$\mathbf{x_i} = \xi_\mathbf{xx}^1\mathbf{x_{i-1}} + \xi_\mathbf{xu}^1\cdot\mathbf{u_{i-1}} + \eta_\mathbf{x}^i \quad \forall i = n^0 + 1\cdots, n^1. \tag{24}$$

This expression will be valid till the system reaches sector $\aleph^2$ (at n^1). Note that since the points $\mathbf{x_i}$ are solutions of the approximated model, they satisfy (1) at any time. Then, it is possible to obtain a "Predictive Model" by using the following algorithm.

Algorithm 1: CPWL Model Formulation

Data: A set of control variables $[\mathbf{u}_0, \mathbf{u}_1, \cdots, \mathbf{u}_T]$, the final horizon T, and the initial state vector $\mathbf{x_0}$ solution of the model approximation (17).

- **Step 1)** Set $j = 0$, $k = 0$ and $n^{-1} = 0$.
- **Step 2)** Determine in which sector, $\aleph^j$, point $(\mathbf{x_k}, \mathbf{u_k})$ lies, and compute the linear model valid in this sector.
- **Step 3)** If $k < T$, continue; Otherwise, ***stop***.
- **Step 4)** Compute $\mathbf{x_{k+1}} = \xi_\mathbf{xx}^j\mathbf{x_k} + \xi_\mathbf{xu}^j\cdot\mathbf{u_k} + \eta_\mathbf{x}^j$.
- **Step 5)** If no entry of the vector $\mathbf{z}_\aleph^j = \xi_{\aleph\mathbf{x}}^j\mathbf{x_{k+1}} + \xi_{\aleph\mathbf{u}}^j\mathbf{u_k} + \eta_\aleph^j$ changes sign, let $k = k+1$ and return to Step 3. Otherwise, let $n^j = k - n^{j-1}$ and $j = j+1$, and return to Step 2 to proceed similarly in the next sector.

Using the results of this algorithm, a generic expression for the predictive model, when the system goes through sectors $[\aleph^0, \aleph^1, \cdots, \aleph^h]$, can be written as

$$\mathbf{X} = \Phi_\mathbf{xx}\mathbf{x}_0 + \Phi_\mathbf{xu}\mathbf{U} + \Phi_x \tag{25}$$

where the matrices are defined in Appendix B. It is important to remark the dependence of the matrices Φ_{xx}, Φ_{xu} and Φ_x on the sec-

[2]The domain $\aleph$ is divided into a finite number of polyhedral regions $\aleph^j$ bounded by a set of hyperplanes of the type $\mathbf{z}_\aleph^\mathbf{j} = \xi_{\aleph\mathbf{x}}^\mathbf{j}\cdot\mathbf{x^h} + \xi_{\aleph\mathbf{u}}^\mathbf{j}\cdot\mathbf{u^h} + \eta_\aleph^\mathbf{j} = \mathbf{0}$ with dimensions no lower than $\sigma - 1$.

tors $[\aleph^0, \aleph^1, \aleph^2, \cdots,]$ and on the times $[t^0, t^1, t^2, \cdots,]$. This means, in general, expression (25) is no longer valid if any change occurs in the inputs $\mathbf{u}$.

In some applications, there is no need to consider changes of the manipulated vectors every period. Moreover, it could unnecessarily increase the dimension of the vector $\mathbf{U}$ for the solution of the control problem. To avoid this problem, we propose to modify the matrices involved in (25) in order to consider another sampling time. To do this, we consider that the manipulated variable changes each t sampling times. Then, for horizon m (the input signal changes m times) the length of this horizon is $t_{ih} = m \cdot t$. Note that in this case, the matrices involved in (25) can be easily computed by adding up all t columns inside each t rows.

Now, let us rewrite the control problem using these model. We should note that the initial state vector $\mathbf{x_0}$ is known *a priori*, for our application we can pose this model as

$$\mathbf{X} = \tilde{\Phi}_{\mathbf{x}}(\mathbf{x}_0) + \Phi_{\mathbf{xu}}\mathbf{U} \tag{26}$$

with $\tilde{\Phi}_x(\mathbf{x}_0) = \Phi_{\mathbf{xx}}\mathbf{x}_0 + \Phi_x$. Placing these expressions in the objective function of (18), we obtain the objective function for the optimization as

$$\begin{aligned} V_T(\mathbf{x}, \mathbf{u}) &= \sum_{i=1}^{T} \mathbf{x_i}^T \mathbf{Q_x^i} \mathbf{x_i} + \sum_{i=1}^{T} \mathbf{u_i}^T \mathbf{Q_u^i} \mathbf{u_i} \\ &= \mathbf{X}^T \mathbf{Q_x} \mathbf{X} + \mathbf{U}^T \mathbf{Q_u} \mathbf{U} \\ &= \mathbf{U}^T \Theta_{\mathbf{uu}} \mathbf{U} + \Theta_{\mathbf{u}} \mathbf{U} + \Theta \end{aligned} \tag{27}$$

where $\mathbf{Q_x} = \operatorname{diag}\{\mathbf{Q_x^i}, i = 1, \cdots, T\}$, $\mathbf{Q_u} = \operatorname{diag}\{\mathbf{Q_u^i}, i = 1, \cdots, T\}$, $\Theta_{\mathbf{uu}} = (\Phi_{\mathbf{xu}}^T \mathbf{Q_x} \Phi_{\mathbf{xu}} + \mathbf{Q_u})$, $\Theta_{\mathbf{u}} = 2(\tilde{\Phi}_{\mathbf{x}}^T \mathbf{Q_x} \Phi_{\mathbf{xu}})$ and $\Theta = (\tilde{\Phi}_{\mathbf{x}}^T \mathbf{Q_x} \tilde{\Phi}_{\mathbf{x}})$.

The bounded set of controllers is now expressed as

$$\mathbf{u} \in \overline{U} = \{\mathbf{U} : \mathbf{HU} \leq \mathbf{h}\} \tag{28}$$

with $\mathbf{H}^T = [\mathbf{I}^T \ \ -\mathbf{I}^T]$ and $\mathbf{h}^T = [\mathbf{U}_u^T \ \ -\mathbf{U}_l^T]$.

Then, the control problem of (18) can be solved for each point in the state grid by

$$\min_U V_T = \mathbf{U}^T \Theta_{\mathbf{uu}} \mathbf{U} + \Theta_{\mathbf{u}} \mathbf{U} + \Theta \quad \text{s.t. } \mathbf{HU} \leq \mathbf{h} \tag{29}$$

which is a typical quadratic programming problem that can be solved using any commercial algorithm. Then, we can store the first step of the manipulated variable for each point in the state space, thereby obtaining the numeric controller. Note that in this solution for the problem, we assume that the model (26) is not depending on $\mathbf{u}$. This assumption is unrealistic. Since that only the first control action is implemented, however, this is not a substantial problem. Moreover, an algorithm solution could be implemented by an alternated solution of problem (29) and algorithm for compute CPWL model. This iteration should continue while the norm of two successive solutions of $\mathbf{U}$ is larger than a relative small tolerance. Note that this kind of algorithmic solution is necessary due to the nonlinear nature of the original problem. Moreover, the convergence of this algorithm toward the global optimum cannot always be guaranteed; however, this is the typical problem of nonlinear optimization.

IV. Example: Control of an Inverted Pendulum

Consider the simple inverted pendulum of Fig. 1. The state variables of this system are the angle x_1 and the angular velocity $\dot{x}_1 = x_2$. The input u is a torque in the shaft, which is bounded to such an amount that the pendulum cannot directly be turned from the hanging into the upright position. Instead, it is first necessary to "gain enough momentum," which requires complex trajectory planning, even for this simple example. It is this nonlinearity that poses the main x difficulty

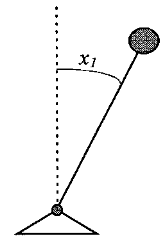

Fig. 1. Inverted pendulum.

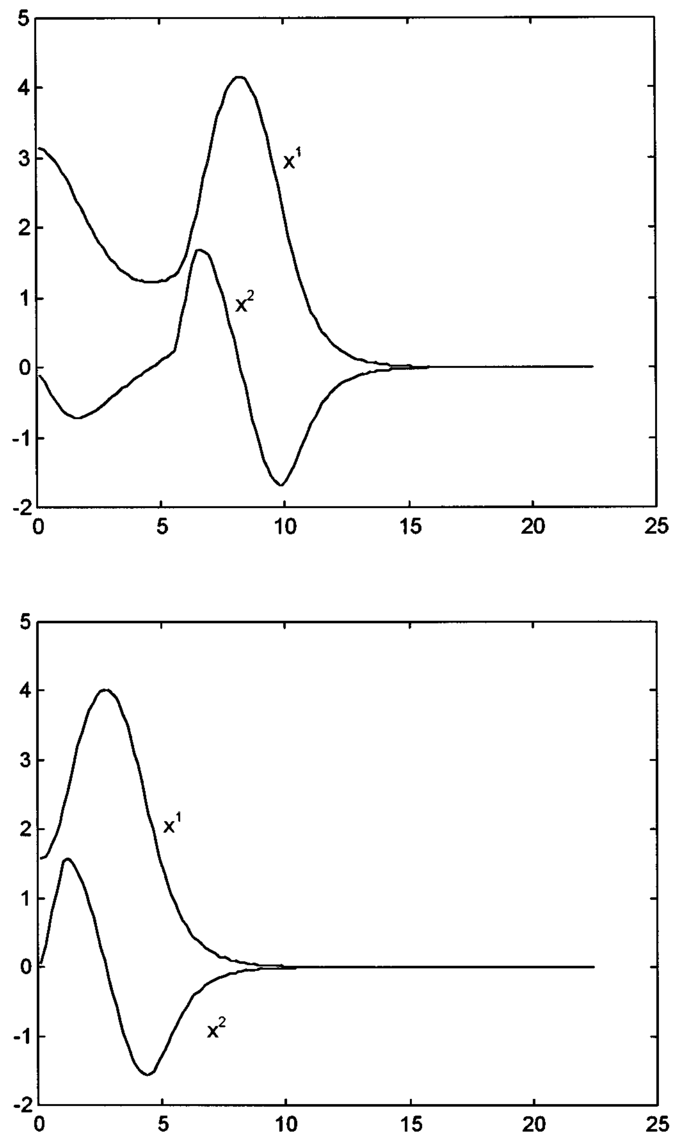

Fig. 2. Time Simulation for cases A and B.

for the feedback design. The control objective is to retain the pendulum in the upright position, which it is an unstable equilibrium point.

The normalized system is described by the differential equation

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= \sin x_1 + g(u) \end{aligned}$$

where

$$g(u) = \begin{cases} -0.7, & \text{for } u \leq -0.7 \\ u & \text{for } -0.7 \leq u \leq -0.7, \\ 0.7 & \text{for } u \geq 0.7 \end{cases}$$

The states x_1 and x_2 are the controlled outputs and the torque u is the manipulated variable. A discrete CPWL model was used to solve the numerical control problem. In this approximation, the space is divided into 43 sectors. The sample time for simulation is 0.005 s. The input sample time is 0.15 s, and the control horizon is 4.5 s. We consider such a small sample time because the system is unstable; therefore, the simulation error will be bounded. The computing region is $\Gamma = \{\mathbf{x} \in \Re^2 \mid |\mathbf{x}_1| \leq 6.6 \text{ and } |\mathbf{x_s}| \leq 3.6\}$.

The manipulated variables are constrained to $-0.7 \leq u \leq 0.7$, and the weighting matrices are defined as

$$Q_x^i = \begin{cases} \begin{bmatrix} 0.01 & 0 \\ 0 & 0.002 \end{bmatrix} \cdot (1.2)^{i-1} & \text{for } i = 1, \cdots, T-1 \\ 2 \cdot \begin{bmatrix} 0.01 & 0 \\ 0 & 0.002 \end{bmatrix} \cdot (1.2)^{i-1}, & \text{for } i = T \end{cases}$$

and

$$Q_u^i = 0.1.$$

Using these data, the numerical controller is computed. Simulations for two sets of initial conditions were performed.

$$\text{Case A: } x_1 = \pi/4 \quad \text{and} \quad x_2 = 0$$
$$\text{Case B: } x_1 = \pi/2 \quad \text{and} \quad x_2 = 0.$$

The results are presented in Fig. 2(a) and (b), respectively. The plots in the phase plane are show in Fig. 3. The numerical results are equivalent to those obtained in [1]. However, the CPWL-based algorithm needs at least one-fifth of the time for convergence as compared with the approach in [1].

V. Conclusion

The result of this paper is an improvement of an existing computer design algorithm, which defines a practically asymptotically stabilizing control from an initial regimen to a target state, if one exists, due to [1]. The improvement is based on a transformation of the original nonlinear problem into a simple quadratic programming problem by using a canonical piecewise lineal approximation of the system model. The applicability of this method is tested by an example.

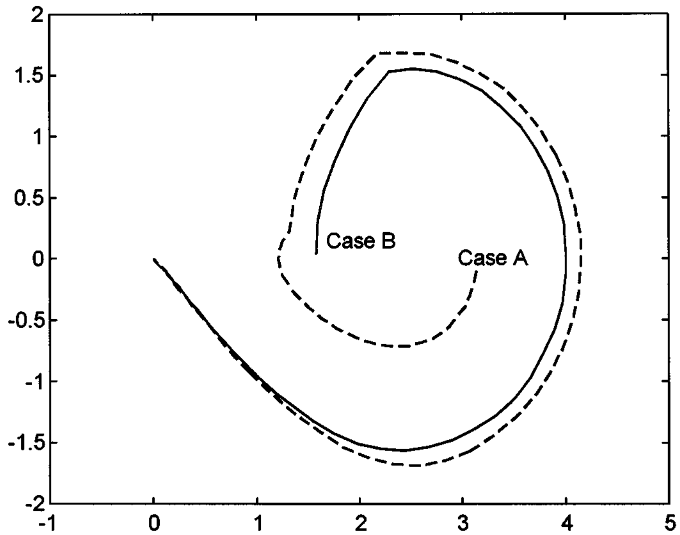

Fig. 3. Phase plane plot.

Appendix A
Function Classes

A function class K_φ, which is used to characterize positive definiteness and decrescence (including radial unboundedness), is defined as follows.

Definition: A function φ: $\Re_o^+ \to \Re_o^+$ is said to belong to the class K_φ, if it is continuous, strictly increasing, and satisfies $\varphi(0) = 0$, and $\lim_{t\to\infty} \varphi(r) = \infty$.

A function class K_σ, which is used to characterize asymptotic stability, is defined as follows.

Definition: A function φ: $\Re_o^+ \times \mathbf{Z_o^+} \to \Re_o^+$ is said to belong to the class K_σ if

1) $\sigma(r,t) \in K_\varphi$ for each fixed $t \in \mathbf{Z_o^+}$,
2) $\sigma(r,t_1) > \sigma(r,t_2)$ for all $t_2 > t_1$ and $r \neq 0$,
3) $\sigma(0,t) = 0$ for all $t \in \mathbf{Z_o^+}$, and
4) $\lim_{t\to\infty} \sigma(r,t) = 0$ for all $r \in \Re_o^+$.

From this definition, $\sigma(r,t)$ is a function that is, as a function of t, strictly monotone decreasing and goes to zero asymptotically, and that is, as a function of T (for all t), positive definite.

$$\Phi_{\mathbf{xu}} = \begin{bmatrix} \xi^0_{\mathbf{xu}} & 0 & \cdots & 0 & \cdots & 0 & 0 \\ \xi^0_{\mathbf{xx}}\xi^0_{\mathbf{xu}} & \xi^0_{\mathbf{xu}} & \cdots & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \\ \left(\xi^0_{\mathbf{xx}}\right)^{n^0-1}\xi^0_{\mathbf{xu}} & \left(\xi^0_{\mathbf{xx}}\right)^{n^0-2}\xi^0_{\mathbf{xu}} & \cdots & 0 & \cdots & 0 & 0 \\ \xi^1_{\mathbf{xx}}\left(\xi^0_{\mathbf{xx}}\right)^{n^0-1}\xi^0_{\mathbf{xu}} & \xi^1_{\mathbf{xx}}\left(\xi^0_{\mathbf{xx}}\right)^{n^0-2}\xi^0_{\mathbf{xu}} & \cdots & \xi^1_{\mathbf{xu}} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots \\ \left(\xi^1_{\mathbf{xx}}\right)^{n^1}\left(\xi^0_{\mathbf{xx}}\right)^{n^0-1}\xi^0_{\mathbf{xu}} & \left(\xi^1_{\mathbf{xx}}\right)^{n^1}\left(\xi^0_{\mathbf{xx}}\right)^{n^0-2}\xi^0_{\mathbf{xu}} & \cdots & \left(\xi^1_{\mathbf{xx}}\right)^{n^0-1}\xi^1_{\mathbf{xu}} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \cdots & \vdots & \\ \left(\xi_{\mathbf{xx}}\right)^{n^h-1}\cdots\left(\xi^1_{\mathbf{xx}}\right)^{n^1}\left(\xi^0_{\mathbf{xx}}\right)^{n^0-1}\xi^0_{\mathbf{xu}} & \left(\xi^h_{\mathbf{xx}}\right)^{n^h-1}\cdots\left(\xi^1_{\mathbf{xx}}\right)^{n^1}\left(\xi^0_{\mathbf{xx}}\right)^{n^0-2}\xi^0_{\mathbf{xu}} & \cdots & \left(\xi^h_{\mathbf{xx}}\right)^{n^h-1}\cdots\left(\xi^1_{\mathbf{xx}}\right)^{n^0-1}\xi^1_{\mathbf{xu}} & \cdots & \xi^h_{\mathbf{xu}} & 0 \\ \left(\xi_{\mathbf{xx}}\right)^{n^h}\cdots\left(\xi^1_{\mathbf{xx}}\right)^{n^1}\left(\xi^0_{\mathbf{xx}}\right)^{n^0-1}\xi^0_{\mathbf{xu}} & \left(\xi^h_{\mathbf{xx}}\right)^{n^h}\cdots\left(\xi^1_{\mathbf{xx}}\right)^{n^1}\left(\xi^0_{\mathbf{xx}}\right)^{n^0-2}\xi^0_{\mathbf{xu}} & \cdots & \left(\xi^h_{\mathbf{xx}}\right)^{n^h}\cdots\left(\xi^1_{\mathbf{xx}}\right)^{n^0-1}\xi^1_{\mathbf{xu}} & \cdots & \xi^h_{\mathbf{xu}}\xi^h_{\mathbf{xu}} & \xi^h_{\mathbf{xu}} \end{bmatrix}$$

APPENDIX B
MODEL MATRICES

The matrices that define the Predictive Model

$$\mathbf{X} = \Phi_{\mathbf{xx}}\mathbf{x}_0 + \Phi_{\mathbf{xu}}\mathbf{U} + \Phi_x$$

are defined as $\mathbf{U} \equiv [\mathbf{u}_o^T \;\; \mathbf{u}_1^T, \cdots, \mathbf{u}_T^T]^T$, $\mathbf{X} = [\mathbf{x}_1^T \;\; \mathbf{x}_2^T, \cdots, \;\; \mathbf{x}_T^T]^T$ and as shown at the bottom of the previous page

$$\Phi_{\mathbf{xx}} = \begin{bmatrix} \xi_{\mathbf{xx}}^0 \\ \left(\xi_{\mathbf{xx}}^0\right)^2 \\ \vdots \\ \left(\xi_{\mathbf{xx}}^0\right)^{n^0} \\ \xi_{\mathbf{xx}}^1 \left(\xi_{\mathbf{xx}}^0\right)^{n^0} \\ \vdots \\ \left(\xi_{\mathbf{xx}}^1\right)^{n^1} \left(\xi_{\mathbf{xx}}^0\right)^{n^0} \\ \vdots \\ \left(\xi_{\mathbf{xx}}^h\right)^{n^h-1} \cdots \left(\xi_{\mathbf{xx}}^1\right)^{n^1} \left(\xi_{\mathbf{xx}}^0\right)^{n^0} \\ \left(\xi_{\mathbf{xx}}^h\right)^{n^h} \cdots \left(\xi_{\mathbf{xx}}^1\right)^{n^1} \left(\xi_{\mathbf{xx}}^0\right)^{n^0} \end{bmatrix}$$

$$\Phi_{\mathbf{x}} = \begin{bmatrix} \eta_x^0 \\ \xi_{\mathbf{xx}}^0 \eta_x^0 + \eta_x^0 \\ \vdots \\ \sum_{t=1}^{n^0} (\xi_{\mathbf{xx}}^0)^{t-1} \eta_x^0 \\ \xi_{\mathbf{xx}}^1 \sum_{t=1}^{n^0} \left(\xi_{\mathbf{xx}}^0\right)^{t-1} \eta_x^0 + \eta_x^1 \\ \vdots \\ \sum_{j=1}^{n^1} \left(\left(\xi_{\mathbf{xx}}^1\right)^j \sum_{t=1}^{n^0} \left(\xi_{\mathbf{xx}}^0\right)^{t-1} \eta_x^0 + \left(\xi_{\mathbf{xx}}^0\right)^{j-1} \eta_x^1 \right) \\ \vdots \\ \sum_{j=1}^{n^{n-1}} \left(\left(\xi_{\mathbf{xx}}^h\right)^j \left(\cdots \sum_{t=1}^{n^0} \left(\xi_{\mathbf{xx}}^0\right)^{t-1} \eta_x^0 \cdots \right) + \left(\xi_{\mathbf{xx}}^h\right)^{j-1} \eta_x^h \right) \\ \sum_{j=1}^{n^n} \left(\left(\xi_{\mathbf{xx}}^h\right)^j \left(\cdots \sum_{t=1}^{n^0} \left(\xi_{\mathbf{xx}}^0\right)^{t-1} \eta_x^0 \cdots \right) + \left(\xi_{\mathbf{xx}}^h\right)^{j-1} \eta_x^h \right) \end{bmatrix}$$

where n^i is the sample time at which the system leaves sector $\aleph^i$. Note the dependence of the matrices $\Phi_{\mathbf{xx}}$, $\Phi_{\mathbf{xu}}$ and $\Phi_{\mathbf{x}}$ on the sectors $[\aleph^0, \aleph^1, \aleph^2, \cdots, \aleph^h]$ and on the "times" $[n^0, n^1, n^2, \cdots, n^h]$. This means, in general, the expression is no longer valid if any change occurs in the inputs $\mathbf{u}_k$ for all $k = 1, \cdots, T$. In these expressions, $p = n^0 + n^1 + n^2 +, \cdots, +n^h$.

REFERENCES

[1] G. Kreisselmeier and T. Birkholzer, "Numerical nonlinear regulator design," *IEEE Trans. Automat. Contr.*, vol. 39, pp. 33–46, 1994.
[2] R. Bellman, *Dynamic Programming*. Princeton, NJ: Princeton Univ. Press, 1957, ch. 3 and 4.
[3] D. Bertsekas, *Dynamic Programming and Optimal Control*. Englewood Cliffs, NJ: Prentice-Hall, 1995, vol. I, ch. 1 and 3.
[4] L. O. Chua and L. P. Ying, "Canonical piecewise-linear analysis," *IEEE Trans. Circuits Syst.*, vol. CAS-30, Mar. 1983.
[5] L. O. Chua and A. C. Deng, "Canonical piecewise-linear modeling," *IEEE Trans. Circuits Syst.*, vol. CAS-33, May 1986.
[6] J. L. Figueroa and A. C. Desages, "Use of piecewise linear approximations for steady-state back-off analysis," *Opt. Contr.: Applicat. Methods*, vol. 19, pp. 93–100, 1998.
[7] J. N. Lin and R. Unbehahuen, "Adaptive nonlinear digital filter with canonical piecewise-linear function structure," *IEEE Trans. Circuits Syst.*, vol. 37, pp. 347–353, 1990.
[8] ——, "Canonical piecewise-linear approximations," *IEEE Trans. Circuits Syst.*, vol. 39, pp. 697–699, 1992.
[9] K. Yamamura, "An algorithm for representing functions of many variables by superposition's of functions of one variable and addition," *Trans. Circuits Syst.*, vol. 43, pp. 338–340, 1996.

Constrained Robustness Analysis by Randomized Algorithms

Xinjia Chen and Kemin Zhou

***Abstract*—This paper shows that many robust control problems can be formulated as constrained optimization problems and can be tackled by using randomized algorithms. Two different approaches in searching reliable solutions to robustness analysis problems under constraints are proposed, and the minimum computational efforts for achieving certain reliability and accuracy are investigated and bounds for sample size are derived. Moreover, the existing order statistics distribution theory is extended to the general case in which the distribution of population is not assumed to be continuous and the order statistics is associated with certain constraints.**

***Index Terms*—Order statistics, randomized algorithm, robustness, sample size.**

I. INTRODUCTION

It is now well known that many deterministic worst case robustness analysis and synthesis problems are NP hard, which means the exact analysis and synthesis of the corresponding robust control problems may be computationally demanding [5], [13]. On the other hand, the deterministic worst case robustness measures may be quite conservative due to overbounding of the system uncertainties. As pointed out in [9] by Khargonekar and Tikku, the difficulties of deterministic worst case robust control problems are inherent to the problem formulations, and a major change of the paradigm is necessary. An alternative to the deterministic approach is the probabilistic approach, which has been studied extensively by Stengel, *et al.* (see, e.g., [10], [11]), and references therein. Aimed at breaking through the NP-hardness barrier and reducing the conservativeness of the deterministic robustness measures, the probabilistic approach has recently received a renewed attention in the works of Barmish and Lagoa [4], Barmish, *et al.* [2], Barmish and Polyak [3], Khargonekar and Tikku [9], Bai, *et al.* [1], Tempo, *et al.* [12], Yoon and Khargonekar [14], Zhu, *et al.* [16], Chen and Zhou [6], [7], and references therein.

In addition to its low computational complexity, the advantages of randomized algorithms can be found in the flexibility and adaptiveness in dealing with control analysis (and possibly synthesis) problems with

Manuscript received May 19, 1999. Recommended by Associate Editor, R. Tempo. This work was supported in part by the ARO under Grant DAAH04-96-1-0193, by the AFOSR under Grant F49620-94-1-0415, and by the LEQSF under Grant DOD/LEQSF(1996-99)-04.

The authors are with the Department of Electrical and Computer Engineering, Louisiana State University, Baton Rouge, LA 70803 USA (e-mail: chan@ee.lsu.edu; kemin@ee.lsu.edu).

Publisher Item Identifier S 0018-9286(00)04225-2.