Int. J. Appl. Math. Comput. Sci., 2003, Vol. 13, No. 4, 515–526

TEST SIGNAL DESIGN FOR FAILURE DETECTION: A LINEAR PROGRAMMING APPROACH

HÉCTOR RUBIO SCOLA*, RAMINE NIKOUKHAH**
FRANÇOIS DELEBECQUE**

* Departamento de Electrónica FCEIA – CIUNR
Universidad Nacional de Rosario, Rio Bamba 245 bis, 2000 Rosario, Argentina
e-mail: erubio@fceia.unr.edu.ar

** INRIA, Rocquencourt BP 105, 78153 Le Chesnay Cedex, France
e-mail: {ramine.nikoukhah, francois.delebecque}@inria.fr

A new methodology for the design of filters that permits failure detection and isolation of dynamic systems is presented. Assuming that the normal and the faulty behavior of a process can be modeled by two linear systems subject to inequality bounded perturbations, a method for the on-line implementation of a test signal, guaranteeing failure detection, is proposed. To improve the fault detectability of the dynamic process, appropriate test signals are injected into the system. All the computations required by the proposed method are implemented as the solution of large sparse linear optimization problems. A simple numerical example is given to illustrate the proposed procedure.

Keywords: failure detection, large scale programming, failure isolation, bounded perturbations, active detection

1. Introduction

The design of failure detection systems entails the consideration of several issues. On one hand, a fast reaction of the failure detection mechanism when a failure occurs in the system is desired. On the other hand, in systems with high performance important degradations in performance, reflected in a high incidence of false alarms, are generally not tolerated during the normal system operation. These two considerations are often conflicting since a system which is designed to react quickly to sudden changes needs to be sensitive to high frequency effects, and therefore it is more sensitive to noise, increasing the probability of the occurrence of false alarm signals by the failure detection system.

The relative importance of these two design issues depends on the system's configuration and is best illustrated through particular examples in which the cost of the different design choices can be evaluated. For instance, false alarms will be better tolerated in a highly redundant system configuration than in a system deprived of significant backup capacities (Gertler, 1998; Mangoubi, 1998).

There are two ways of tackling the problem of failure detection and isolation, which are known as the passive and active approaches. In the so-called passive approach, the detector monitors the inputs and the outputs of a system to find out whether a failure has occurred and, if possible, what kind it is. To achieve this, the measured input-output relation is compared with the normal behavior of the system. The passive approach is used to continuously monitor the system, particularly when the detector cannot act upon the system for material or security reasons. In the field of failure detection, most of the work is devoted to this type of approach.

This approach for detecting changes in dynamical systems has been carefully studied (Basseville and Benveniste, 1985; Clark and Setzer, 1980; Isermann, 1984; Mironovski, 1980; Willsky, 1976) in many application fields, to achieve failure detection in controlled systems or on signal segmentation for speech recognition. Most time-domain model-based methods use all the known or estimated model parameters to perform the two fundamental steps of change detection, that is to say, residual generation and the choice of the statistical decision function (Willsky, 1976).

For instance, both filter innovations and parity checks involve all model parameters, with the possible inclusion of parameter uncertainties, and classical coefficients of probability or Bayesian tests proceed similarly (Basseville *et al.*, 1987).

The active approach to failure detection, which is the one used in this paper, involves acting upon the system on a periodic basis or at critical times using a test signal so as to show abnormal behavior which would otherwise re-

main undetected during the normal operation (Nikoukhah, 1998). The detector can act either by taking the usual inputs of the system or through a special input channel, perhaps modifying the system structure. The structure of the failure detection method considered in this paper is depicted in Fig. 1.

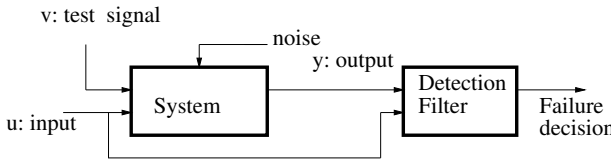

Fig. 1. Active failure detection.

At some given time during the normal operation of the system, a test signal is injected into the system for a finite period of time. This signal exposes the failure modes of the system which are detected by the detection filter.

The design of test signals has been an important issue in system identification for many years, but their use to detect failures has been introduced only recently (Kerestecioğlu, 1993; Uosaki *et al.*, 1984; Zhang, 1989). The test signals (called auxiliary inputs) in these works are regarded as linear outputs of stochastic models, and their objective is to optimize some statistical properties of the detector.

This work considers the detection and isolation of failures using an active approach. The selection of the test signal, as well as the construction of the filter, are performed off-line. The complexity of filter computation and the design of the test signal is important when dealing with high dimensional systems and long test signals. All the operations needed for our method are implemented by solving large linear optimization problems. There are many algorithms for solving such problems (Gondzio, 1996; Mehrotra, 1992; Zhang, 1995), which have been implemented in packages using sparse-matrix data structure utilities for Matlab (Zhang, 1995) and for Scilab (Rubio Scola, 1999). On the other hand, the mathematical operations needed for implementing the filter are a simple scalar product followed by a comparison test.

The method proposed here consists of two basic parts:

Part 1: Finding a signal v such that the set of possible input-output pairs for normal systems is disjoint from the set of possible input-output pairs for failed systems.

Part 2: For the test signal v found, and given an input-output pair, recognizing whether this input-output relation belongs to the normal system's set or to the failed system's set.

The outline of the paper is as follows. In Section 2, basic assumptions are presented and the model is introduced. The solution for Part 2 is characterized in Section 3. This solution is computed off-line over a finite horizon, see, e.g., (Nikoukhah, 1998). The complexity of the off-line computation can be important when dealing with very large systems and long test signals, and this solution cannot be used for many real applications. In Section 4, a solution for Part 1 is proposed. The solution is only given in the case where the test signal enters the system linearly.

Computationally implementable solutions for Part 2, using sparse matrix algorithms, are presented in Section 5. Since the computation time remains acceptable even for very large systems and long test signals, the proposed method can be used in real applications. A simple numerical example is presented in Section 6.

2. System Model

The systems under consideration can be modeled as follows:

$$\begin{aligned} x_i(k+1) &= A_i(k)x_i(k) + B_i(k)u(k) + b_i(k) \\ &\quad + M_i(k)\nu_i(k), \\ y(k) &= C_i(k)x_i(k) + D_i(k)u(k) + d_i(k) \\ &\quad + N_i(k)\nu_i(k), \end{aligned} \tag{1}$$

for $k = 0, \ldots, N-1$, where $i = s$ and $i = f$ correspond respectively to the normal and failed modes. Here u and y are respectively inputs and outputs which are measured on-line, and $\nu_i(k)$ are the (unknown) perturbations. A_i, B_i, C_i, D_i, M_i, N_i, b_i and d_i are matrices and vectors of appropriate dimensions which depend on v and for which the notation $A_i(k) = A_i(v(k))$, etc., has been used, where $v = v(k), k \in [0, N-1]$ is a test signal.

The (known) inputs $u(k)$ and the (unknown) perturbations $\nu_i(k)$ are both supposed to satisfy

$$\begin{aligned} R_{\nu i}(k)\nu_i(k) &\le p_{\nu i}(k), \\ R_{ui}(k)u(k) &\le p_{ui}(k), \end{aligned} \tag{2}$$

where $x_i(k)$, $u(k)$, $y(k)$, $\nu_i(k)$, $p_{\nu i}(k)$, $p_{ui}(k)$ are real vectors, and $R_{\nu i}(k)$, $R_{ui}(k)$ are given matrices of appropriate dimensions. The vectors $p_{\nu i}(k)$, $p_{ui}(k)$ and the matrices $R_{\nu i}(k)$, $R_{ui}(k)$ also depend on v, i.e., $R_{\nu i}(k) = R_{\nu i}(v(k))$, etc. No assumption is made on $R_{\nu i}$ and R_{ui} except that the inequalities (2) are consistent. Here the inequalities should be interpreted as element-wise.

The matrices and vectors do not necessarily have the same dimensions in both systems. The systems have in common only the input $u(k)$ and the output $y(k)$.

The basic assumption is that the normal and failed mode of the system can be modeled as in (1) and (2). But the system matrices can be (and hopefully are) different for different operating modes.

Note that unlike most other approaches to uncertainty modeling in dynamical systems for the purpose of failure detection, ν is not a stochastic white noise sequence, but rather an arbitrary inequality bounded discrete sequence.

A fundamental, and reasonable, assumption here is that, during the test period, the system is either in the normal mode or the failed mode; no transition occurs during the test period.

Let w be the vector defined by

$$w(k)=\begin{cases}\left[x_s(k)^T, x_f(k)^T, y(k)^T, u(k)^T, \nu_s(k)^T, \nu_f(k)^T\right]^T \\ \qquad\qquad \text{if } \ k\in[0,N-1], \\ \left[x_s(N)^T, x_f(N)^T\right]^T \quad \text{if } \ k=N.\end{cases} \tag{3}$$

The equations and inequalities (1), (2) for the normal system ($i=s$) and the failed system ($i=f$) can be written in the matrix form as

$$\begin{aligned} F_i(v)w &= p_i(v), \\ E_i(v)w &\le q_i(v), \end{aligned} \tag{4}$$

or, equivalently, as

$$\begin{aligned} F(v)w &= p(v), \\ E(v)w &\le q(v), \end{aligned} \tag{5}$$

with

$$F(v)=\begin{bmatrix} F_s(v) \\ F_f(v)\end{bmatrix}, \qquad E(v)=\begin{bmatrix} E_s(v) \\ E_f(v)\end{bmatrix},$$
$$p(v)=\begin{bmatrix} p_s(v) \\ p_f(v)\end{bmatrix}, \qquad q(v)=\begin{bmatrix} q_s(v) \\ q_f(v)\end{bmatrix}. \tag{6}$$

The following polyhedra can then be defined:

$$S_s(v)=\{w \mid w \text{ satisfies } (4) \text{ with } i=s\}, \tag{7}$$

$$S_f(v)=\{w \mid w \text{ satisfies } (4) \text{ with } i=f\}, \tag{8}$$

$$S(v)=\{w \mid w \text{ satisfies } (5)\ \}=S_s(v)\cap S_f(v). \tag{9}$$

3. Implementation of the Detection Filter

In this section it is assumed that the test signal $v=\{v(k), k\in[0,N-1]\}$ is given. The test signal v is a sequence of vectors, as short as possible, such that the constraints on the normal system (4) with $i=s$ and the constraints on the failed system (4) with $i=f$ are inconsistent, i.e., $S(v)=\emptyset$. In Section 4, a method for constructing a test signal v is given.

Let $v=v(k),\ k\in[0,N-1]$ be a test signal and $w=w(k),\ k\in[0,N]$ be an observation of the system state. The detection problem consists now in deciding whether this vector w is compatible with $i=s$ or with $i=f$ in (4).

Since the inequalities (4) define two disjoint convex polyhedra, the problem is reduced to knowing to which polyhedron the vector w belongs. If a hyperplane can be found that separates S_s and S_f, it is sufficient to find on which side of the hyperplane the vector w lies. Our work is limited to finding such a hyperplane. Its existence is guaranteed by the classical convexity theory.

The following lemma shows how to obtain constraints involving only inputs u and outputs y for testing a failure without calculating the state variables x_s and x_f of the systems (1).

Lemma 1. *Let S_s and S_f be two nonempty disjoint convex cylindrical polyhedra defined by (7) and (8), respectively. Then the equation of any separating hyperplane is of the form $h_y y+h_u u=d$.*

Proof. Suppose that the hyperplane is defined by

$$h_s\tilde{w}_s+h_f\tilde{w}_f+h_y y+h_u u=d, \tag{10}$$

where $\tilde{w}_s=(x_s,\nu_s)$, $\tilde{w}_f=(x_f,\nu_f)$, and h_s, h_f, h_y, h_y are not all zero.

Let $(\tilde{w}_s^o,\tilde{w}_f^o,y^o,u^o)\in S_s$. Then for all $\tilde{w}_f$, the point $(\tilde{w}_s^o,\tilde{w}_f,y^o,u^o)\in S_s$. It follows that

$$h_s\tilde{w}_s^o+h_f\tilde{w}_f+h_y y^o+h_u u^o\ge d. \tag{11}$$

Since $h_s\tilde{w}_s^o+h_y y^o+h_u u^o$ is fixed, and $h_f\tilde{w}_f$ can take any value because $\tilde{w}_f$ is arbitrary, the expression (11) can take values less than d, which contradicts the assumption that the hyperplane separates S_s and S_f. Thus $h_f=0$. Analogously, h_s must be zero. ■

The following lemma (see Rockafellar, 1972, p. 98, Thm. 11.4) and its corollary show that it is possible to convert the problem of separating two polyhedra into an equivalent problem, separating a polyhedron from the origin.

Lemma 2. *Let S_1 and S_2 be two nonempty convex polyhedra. There exists a hyperplane separating S_1 and S_2 if and only if the convex polyhedron S_1-S_2 does not contain 0, i.e., if and only if there exists a hyperplane separating 0 and the convex polyhedron S_1-S_2.*

Corollary 1. *Let S_1 and S_2 be two nonempty convex polyhedra. Then the hyperplane $hz = d$ separates S_1 and S_2 if and only if the hyperplane $hz_d = d_d$ separates 0 and the convex polyhedron $S_1 - S_2$, i.e., if and only if a hyperplane separating S_1 and S_2 can be chosen parallel to a hyperplane separating 0 and $S_1 - S_2$.*

3.1. Construction of a Difference Polyhedron

Owing to Corollary 1, the problem is now to find a hyperplane that separates a polyhedron from the origin. It will be solved by linear programming and taking into account the geometric property of the convex polyhedra given in Lemma 1.

Let w_s and w_f be such that $w_s \in S_s$ and $w_f \in S_f$, where

$$w_i(k) = \begin{cases} \left[x_s(k)^T, x_f(k)^T, y_i(k)^T, u_i(k)^T, \nu_s(k)^T, \nu_f(k)^T\right]^T \\ \qquad \text{if } \ k \in [0, N-1], \\ \left[x_s(N)^T, x_f(N)^T\right]^T \quad \text{if } \ k = N, \end{cases} \tag{12}$$

with $i = s, f$.

For a normal system, Eqn. (1) can be rewritten with $i = s$, so that the input-output pair (y, u) is set to (y_s, u_s). Analogously, for a failed system, Eqn. (1) can be rewritten with $i = f$, and in this case the pair (y, u) is set to (y_f, u_f).

The difference between input-output pairs for normal and failed systems can be defined as follows:

$$e(k) = \begin{bmatrix} e_y(k) \\ e_u(k) \end{bmatrix} = \begin{bmatrix} y_s(k) \\ u_s(k) \end{bmatrix} - \begin{bmatrix} y_f(k) \\ u_f(k) \end{bmatrix}. \tag{13}$$

Let $\bar{w} = [w_s^T, e^T]^T$, $\bar{F} = [F, F_e]$ and $\bar{E} = [E, E_e]$. Using (5) and (13), the following equation for $\bar{w}$ is obtained:

$$\begin{aligned} \bar{F}\bar{w} &= p, \\ \bar{E}\bar{w} &\le q. \end{aligned} \tag{14}$$

The difference polyhedron is defined as

$$T = \{\bar{w} \mid \bar{w} \text{ satisfies } (14)\}. \tag{15}$$

From the definitions of S_s, S_f and T, we get the following result:

Lemma 3. *The projection of $S_s - S_f$ onto the e coordinate is equal to the projection of T onto the same coordinate.*

The properties of S_s, S_f and $S_s - S_f$ given by Corollary 1 and Lemma 3 are schematically depicted in Fig. 2.

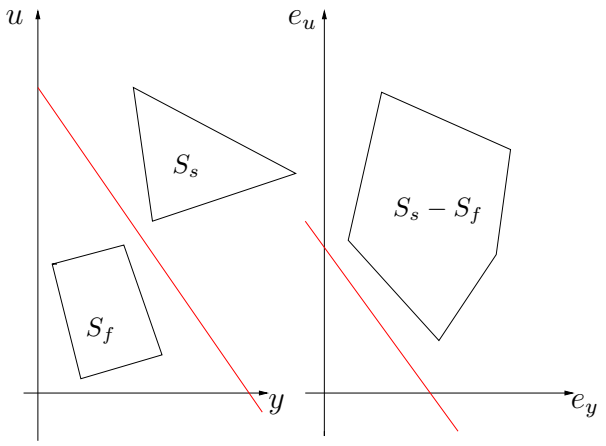

Fig. 2. Properties of S_s and S_f.

3.2. Construction of a Separating Hyperplane

It is assumed that the solution sets of the systems (4) for $i = s, f$ have no intersection, i.e., the system (5) has no solution. Then (14) has no solution of the form $[\ w^T \quad 0\ ]^T$ because (14) becomes (5) if $e = 0$.

Introducing a relaxation parameter

$$\delta = \big[\delta_1(0), \delta_2(0), \delta_3(0), \ldots, \delta_1(N-1), \delta_2(N-1), \delta_3(N-1)\big]^T \tag{16}$$

in (14) yields

$$\begin{aligned} Fw + F_e e &\le p + \delta_1, \\ -Fw - F_e e &\le -p + \delta_2, \\ Ew + E_e e &\le q + \delta_3. \end{aligned} \tag{17}$$

Let us define the following polyhedra:

$$\mathrm{P}_\delta = \{e \mid \exists w, (w, e) \text{ satisfies } (17)\}, \tag{18}$$

and note their properties:

- For $\delta = 0$, $0 \notin \mathrm{P}_\delta$.
- If $\delta^1 \ge \delta^2$, then $\mathrm{P}_{\delta^1} \supseteq \mathrm{P}_{\delta^2}$.
- For $\delta = 0$, the projection of $S_s - S_f$ onto the e coordinate is P_δ.

Choosing an appropriate δ, the polyhedron P_δ can be enlarged until $e = 0$ belongs to it, see Fig. 3.

Taking into account the polyhedron (17), the following linear programming problem is solved:

$$\min_{w,\delta} \sum_{j=1}^{3} \sum_{k=0}^{N-1} \delta_j(k) \tag{19}$$

subject to (17), $e = 0$ and $\delta \ge 0$.

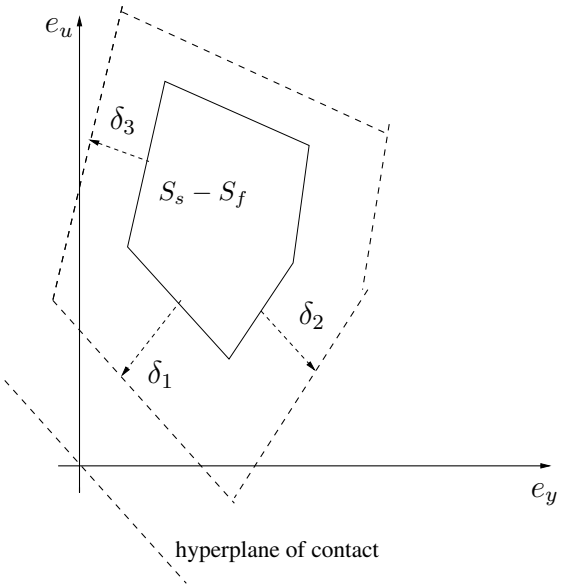

Fig. 3. Linear programming problem.

Let (w^o, δ^o) be a solution to (19), where $\delta^o = [\delta_1^{oT}, \delta_2^{oT}, \delta_3^{oT}]^T$. Then permuting the rows of (17) (evaluated at $(w^o, 0)$), in such a way that the equality constraints appear first, a permutation matrix P_1 is obtained, such that

$$P_1 [F, F_e] = \begin{bmatrix} F_{q,1} & F_{e,q,1} \\ F_{i,1} & F_{e,i,1} \end{bmatrix},$$
$$P_1 p = \begin{bmatrix} p_{q,1} \\ p_{i,1} \end{bmatrix}, \quad P_1 \delta_1^o = \begin{bmatrix} \delta_{q,1}^o \\ \delta_{i,1}^o \end{bmatrix}. \tag{20}$$

Equation (17) with $e = 0$ can be rewritten as follows:

$$\begin{aligned} F_{q,1} w^o + F_{e,q,1} 0 &= p_{q,1} + \delta_{q,1}^o, \\ F_{i,1} w^o + F_{e,i,1} 0 &< p_{i,1} + \delta_{i,1}^o. \end{aligned} \tag{21}$$

Analogously, P_2 and P_3 can be defined as

$$P_2 [F, F_e] = \begin{bmatrix} F_{q,2} & F_{e,q,2} \\ F_{i,2} & F_{e,i,2} \end{bmatrix},$$
$$P_3 [E, E_e] = \begin{bmatrix} E_q & E_{e,q} \\ E_i & E_{e,i} \end{bmatrix}, \tag{22}$$

so that Eqn. (17) can be rewritten as

$$\begin{aligned} -F_{q,2} w^o - F_{e,q,2} 0 &= p_{q,2} + \delta_{q,2}^o, \\ -F_{i,2} w^o - F_{e,i,2} 0 &< p_{i,2} + \delta_{i,2}^o, \end{aligned} \tag{23}$$

$$\begin{aligned} E_q w^o + E_{e,q} 0 &= q_q + \delta_{q,3}^o, \\ E_i w^o + E_{e,i} 0 &< q_i + \delta_{i,3}^o. \end{aligned} \tag{24}$$

The active constraints in (17) become

$$\begin{aligned} F_{q,1} w^o + F_{e,q,1} 0 &= p_{q,1} + \delta_{q,1}^o, \\ -F_{q,2} w^o - F_{e,q,2} 0 &= -p_{q,2} + \delta_{q,2}^o, \\ E_q w^o + E_{e,q} 0 &= q_q + \delta_{q,3}^o. \end{aligned} \tag{25}$$

The general equations for the hyperplanes defined by these active constraints are

$$\begin{aligned} F_{q,1} w + F_{e,q,1} e &= p_{q,1} + \delta_{q,1}^o, \\ -F_{q,2} w - F_{e,q,2} e &= -p_{q,2} + \delta_{q,2}^o, \\ E_q w + E_{e,q} e &= q_q + \delta_{q,3}^o. \end{aligned} \tag{26}$$

Equation (26) can be written down as

$$\phi w + \Psi e = b + \delta_q^o, \tag{27}$$

where

$$\phi = \begin{bmatrix} F_{q,1} \\ -F_{q,2} \\ E_q \end{bmatrix}, \quad \Psi = \begin{bmatrix} F_{e,q,1} \\ -F_{e,q,2} \\ E_{e,q} \end{bmatrix},$$
$$b = \begin{bmatrix} p_{q,1} \\ -p_{q,2} \\ q_q \end{bmatrix}, \quad \delta_q^o = \begin{bmatrix} \delta_{q,1}^o \\ \delta_{q,2}^o \\ \delta_{q,3}^o \end{bmatrix}. \tag{28}$$

In (26), the hyperplane is defined as a function of the variables (x, y, u, ν) and (x_f, y, u, ν_f). Now, in accordance with Lemma 1, the variables x, x_f, ν, ν_f do not appear in the hyperplane definition (26) because, as it is defined by (x, y, u, ν) and (x_f, y, u, ν_f), the common variables in the two sets are (y, u).

The following lemma and theorem show how to obtain a separating hyperplane direction from (27).

Lemma 4. *Let K be a full rank matrix whose columns span $\ker(\phi^T)$. If e and w satisfy the constraints $\phi w + \Psi e = b + \delta_q^o$, then e satisfies $He = 0$, where $H = K^T \Psi$.*

Proof. Multiplying both sides of (27) by K^T gives $K^T \phi w + K^T \Psi e = K^T (b + \delta_q^o)$. Now, Eqn. (27) for e = 0 is $\phi w = b + \delta_q^o$. Also, it can be easily proved that $\xi \in \text{Im}(\phi) \iff (\ker(\phi^T))^T \xi = 0$, so that $K^T (b + \delta_q^o) = 0$, i.e., $He + 0 = 0$. ■

Multiplying both sides of (27) by K^T defines a set of hyperplanes $He = 0$ (cf. Lemma 4). By (13) and Corollary 1, $H(y, u) = d$ defines a set of hyperplanes. There exists at least one hyperplane that separates the polyhedra S_s and S_f, which is denoted by $H_i(y, u) = d_i$. It remains to determine d_i. To do that,

the hyperplanes tangent to the two convex polyhedra S_s and S_f are calculated:

$$\begin{aligned} m_i^1 &= \min H_i(y,u), \quad M_i^1 = \max H_i(y,u), \\ &\text{s.t. } w \in S_f, \qquad\quad \text{s.t. } w \in S_f, \\ m_i^2 &= \min H_i(y,u), \quad M_i^2 = \max H_i(y,u), \\ &\text{s.t. } w \in S_s, \qquad\quad \text{s.t. } w \in S_s. \end{aligned} \tag{29}$$

If $M_i^1 < M_i^2$ then $d_i = (M_i^1 + m_i^2)/2$, or, if $M_i^1 > M_i^2$, then $d_i = (M_i^2 + m_i^1)/2$ with $i = 1, \ldots, n_p$.

Then the hyperplane equation will be

$$H_i(y,u) = d_i. \tag{30}$$

The following theorem shows that there always exists $i \in [1, n_p]$ such that the hyperplane in (30) separates the two polyhedra.

Theorem 1. *There exists $i \in [1, n_p]$ such that $H_i(y,u) = d_i$ separates the convex polyhedra S_s and S_f.*

Proof. Note that

$$\text{if } \left[w^T, e^T\right]^T \in T \text{ then } He \neq 0. \tag{31}$$

Let us suppose that

$$\left[w^T, e^T\right]^T \in T \text{ and } He = 0. \tag{32}$$

If $\left[w^T, e^T\right]^T \in T$, then $\Psi e + \phi w \leq b$. Let b_1 be such that

$$\Psi e + \phi w = b_1 \leq b. \tag{33}$$

Multiplying both sides of (33) by K^T yields

$$\begin{aligned} K^T \Psi e + K^T \phi z &= K^T b_1, \\ He &= K^T b_1. \end{aligned} \tag{34}$$

It follows that $K^T \ b_1 = 0$, i.e., $b_1 \in \mathrm{Im}\phi$. Then there exists w_1 such that

$$\phi w_1 = b_1 \leq b. \tag{35}$$

Let δ_q^o be a solution to the linear programming problem (19), i.e., $\delta_q^o \geq 0$ is the minimal l_1 norm such that $\phi w \leq b + \delta_q^o$ has a solution. We have $\delta_q^o \neq 0$ since (5) has no solution. It follows that the inequality $\phi w \leq b$ has no solution, i.e., (35) does not hold, (32) is false and (31) holds.

It follows that there exists $i \in [1, n_p]$ such that $H_i(y_s - y_f, u_s - u_f) \neq 0$ and $H_i(y_s, u_s) \neq H_i(y_f, u_f), \ \ \forall\, w_s \in S_s, \ \ w_f \in S_f$, i.e.,

$$H_i(S_s) \cap H_i(S_f) = \emptyset. \tag{36}$$

Since S_s and S_f are convex, so are $H_i(S_s)$ and $H_i(S_f)$, and it follows that $H_i(y_s, u_s) < H_i(y_f, u_f)$ or $H_i(y_s, u_s) > H_i(y_f, u_f)$ holds.

If $H_i(y_s, u_s) < H_i(y_f, u_f)$, then

$$\begin{aligned} H_i(y_s, u_s) &\leq \max_{w_s \in S_s} H_i(y_s, u_s) < d_i \\ &< \min_{w_f \in S_f} H_i(y_f, u_f) \leq H_i(y_f, u_f). \end{aligned} \tag{37}$$

If $H_i(y_s, u_s) > H_i(y_f, u_f)$, then

$$\begin{aligned} H_i(y_s, u_s) &\geq \min_{w_s \in S_s} H_i(y_s, u_s) > d_i \\ &> \max_{w_f \in S_f} H_i(y_f, u_f) \geq H_i(y_f, u_f). \end{aligned} \tag{38}$$

This shows that the hyperplane defined by the equation $H_i(y,u) = d_i$ separates the convex polyhedra S_s and S_f. ■

4. Test Signal Design

In this section it is shown how a detection horizon N can be found and how a test signal $v = \{v(k), k \in [0, N-1]\}$, which is as short as possible and such that (1) and (2) for $i = s, f$ are mutually exclusive, i.e., $S_f \cap S_s = \emptyset$, can be constructed.

The solution to this problem is given only for the case where the test signal enters the system linearly. This problem can be considered to be the counter-part of the off-line auxiliary signal design problem of (Zhang, 1989). In this paper, it is shown how a test signal can be designed for the special class of Model (1), (2). In particular, it is assumed that the matrices $A_i(k)$, $B_i(k)$, $C_i(k)$, $D_i(k)$, $M_i(k)$ and $N_i(k)$, for $i = s, f$ do not depend on v and that

$$\begin{aligned} b_i\big(v(k)\big) &= b_{i,1}(k)v(k) + b_{i,o}(k), \\ d_i\big(v(k)\big) &= d_{i,1}(k)v(k) + d_{i,o}(k), \end{aligned} \tag{39}$$

where $b_{i,1}(k)$ and $d_{i,1}(k)$ are matrices of appropriate dimensions, and $b_{i,o}(k)$ and $d_{i,o}(k)$ are vectors, $i = s, f$. On these assumptions, (1) and (2) can be rewritten as

$$\begin{aligned} Fw + Rv &= p_o, \\ Ew &\leq q. \end{aligned} \tag{40}$$

The problem is then to find v such that (40) is not satisfied for any w. The construction makes use of classical convexity results (Rockafellar, 1972).

Consider the polyhedron

$$S_h = \{v \mid \exists\, w, \ (w, v) \text{ satisfies } (40)\}. \tag{41}$$

S_h is a convex polyhedron and can be expressed by means of inequality constraints in v (this results from the fact that the projection of a convex polyhedron is a convex polyhedron).

Equation (40) can be rewritten as

$$\begin{aligned} Fw + Rv &= p_o, \\ Ew + \delta &= q, \\ \delta &\geq 0. \end{aligned} \tag{42}$$

The following lemma provides an equivalent formulation to the above-mentioned problem of finding v such that (42) is not satisfied.

Lemma 5. *Let K be a full rank matrix whose columns span $\ker(\left[F^T, E^T\right])$. If v and δ satisfy (42), then v and δ satisfy*

$$\begin{aligned} Lv + G\delta &= h, \\ \delta &\geq 0, \end{aligned} \tag{43}$$

where

$$L = K^T \begin{bmatrix} R \\ 0 \end{bmatrix}, \quad G = K^T \begin{bmatrix} O \\ I \end{bmatrix}, \quad h = K^T \begin{bmatrix} p_o \\ q \end{bmatrix}. \tag{44}$$

If $\ker(\left[F^T, E^T\right]) = 0$, then for all v (40) has a solution.

Proof. The proof is straighforward and therefore it is omitted here. ■

Now the problem is to find v which does not satisfy (43). Let L_i be the i-th row of matrix L. Consider the following linear programming problem:

$$\max_{v,\delta} L_i v \tag{45}$$

subject to (43).

Let (v^o, δ^o) be a solution to (45) (note that $h_i^o = L_i v^o$). Then

$$L_i v = h_i^o \tag{46}$$

is the equation of the tangent hyperplane to the polyhedron S_h. Then the test signal v can be chosen such that

$$L_i v > h_i^o. \tag{47}$$

4.1. Additional Criteria for the Selection of the Test Signal

It is possible to consider many criteria for choosing a test signal v among the v's for which (40) has no solution. Here, the following constraints will be considered:

$$Q(k)v(k) \leq q_v(k) \tag{48}$$

for $k = 0, \ldots, N-1$. These can be rewritten as

$$Qv \leq q_v, \tag{49}$$

where $Q = \text{diag}\,\{Q(0), Q(1), \ldots, Q(N-1)\}$.

Define the following polyhedron:

$$S_q = \{v \mid v \text{ satisfies } (49)\}. \tag{50}$$

Note that there is no v satisfying (49) and not satisfying (40) if and only if

$$S_q \subseteq S_h. \tag{51}$$

The situation in order for v to exist is represented in Fig. 4. Using the convexity of S_q, the following result can be established:

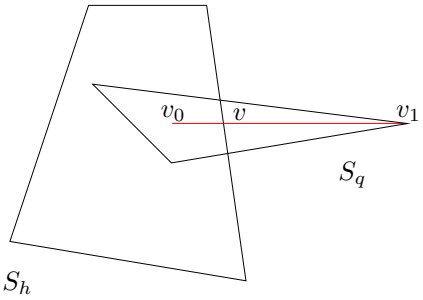

Fig. 4. Test signal design.

Lemma 6. *Let V_q be the set of the vertices of S_q. There is no v satisfying (49) and not satisfying (40) if and only if*

$$V_q \subseteq S_h. \tag{52}$$

It follows that the test signal satisfying the conditions of Lemma 6 is such that $v \in V_q$ and $v \notin S_h$. In other words, a vertex in V_q which does not belong to S_h can be chosen as the signal v (e.g., v_1 in Fig. 4).

Testing (52) is easier than testing (51) because V_q is a finite set, and the test can be performed by element-wise checking if some element v_1 exists such that $v_1 \in V_q$ and $v_1 \notin S_h$. This vertex can then be used as a test signal.

Even though v_1 satisfies the conditions in Lemma 6 for the test signal, it is an extreme solution, since it is on the boundary of the conditions. To improve this situation, a point near the boundary of S_h can be chosen (point v in Fig. 1). Let $v_0 \in S_q \cap S_h$ (without loss of generality, it can be assumed that $v_0 = 0$), and let v be the intersection point of the segment $v_0 v_1$ with the boundary of S_h. Then $v = \lambda v_1$, where λ is the solution to the following problem:

$$\max_{w,\lambda} \lambda \tag{53}$$

subject to

$$\begin{aligned} Fw + \lambda R v_1 &= p_0, \\ Ew &\leq q. \end{aligned} \tag{54}$$

To find v over interval $[0, N-1]$ with N as small as possible, the following algorithm is proposed:

Algorithm 1:

Step 1. Set $N = 1$.

Step 2. Find V_q.

Step 3. If $v_1 \in V_q$ exists such that $v_1 \notin S_h$, go to Step 4, otherwise set $N = N + 1$ and go to Step 1.

Step 4. Calculate λ according to (53), (54), $v = \lambda v_1$.

To calculate the set of vertices V_q, the block diagonal property of matrix Q is exploited. The polyhedron vertices defined for each block will be found and then the corresponding permutations to find all the elements of V_q will be performed.

In Step 3 of Algorithm 1, v_1 is sought among the elements of set V_q. If an element is found, the search ends. In addition, the search is finite because V_q has a finite number of elements.

Owing to Lemma 5, the linear programming problem (53), (54) used by Algorithm 1 can be replaced by an equivalent linear programming problem with fewer variables. This equivalent problem can be formulated as follows: Let $v_o \in V_q$. If $0 \in S_q$, then $v = v_o\lambda \in S_q$, $\forall \lambda \in [0, 1]$. Consider the following linear programming problem:

$$\max_{\delta, \lambda} \lambda \tag{55}$$

subject to

$$\begin{aligned} Lv_o\lambda + G\delta &= h, \\ \delta &\geq 0, \\ 1 \geq \lambda &\geq 0. \end{aligned} \tag{56}$$

Let (δ^o, λ^o) be a solution to (55) and (56). If $\lambda > \lambda^o$ and $\lambda \in [0, 1]$, then $v = v_o\lambda$ does not satisfy (43), i.e. $v \in S_q$ is such that $v \notin S_h$.

Algorithm 1 can be rewritten as follows:

Algorithm 2.

Step 1. Set N = 1.

Step 2. Find V_q.

Step 3. If $v_o \in V_q$ exists such that a solution (λ^o, δ^0) can be found to (55) and (56), go to Step 4, otherwise set $N = N + 1$ and go to Step 1.

Step 4. Set $v = v_1\lambda$ with $1 \leq \lambda > \lambda^o$.

The number of variables in the problem (55), (56) is lower than the number of variables in the problem (53), (54): in (55) and (56) the number of variables is equal to the number of inequality constraints in (54) incremented by one.

4.2. Special Cases

There is some flexibility in the choice of the test signal, and in some special cases it could be interesting to select specific v's. Three different test signals are considered in what follows.

Case 1. If $Q(k) = Q_o$ and $q_v(k) = q_o$ in (49) for $k \in [0, N-1]$, the following polyhedron can be defined:

$$P_o = \{v \mid Q_o v \leq q_o\}. \tag{57}$$

Let $V_o = \{w_1, w_2, \ldots, w_n\}$ be the set of the vertices of the polyhedron P_o. Then the vertices of S_q are permutations of the elements of V_o.

The selection of v can be restricted to the subsets V_1 and V_2 of the set

$$V_q = \{v \mid v \text{ is a vertex of } S_q\}, \tag{58}$$

where V_1 is the set of v's such that

$$v(k) = \begin{cases} w_i & \text{if} \quad k \in [0, k_o], \\ w_j & \text{if} \quad k \in [(k_o + 1), (N-1)] \end{cases} \tag{59}$$

with $k_o = 0, \ldots, N-1$ and $w_i, w_j \in V_o$, and V_2 is the set of v's such that

$$v(k) = \begin{cases} w_i & \text{if} \quad k \in [0, k_o], \\ w_j & \text{if} \quad k \in [(k_o + 1), k_1], \\ w_l & \text{if} \quad k \in [(k_1 + 1), (N-1)] \end{cases} \tag{60}$$

with $k_o, k_1 \in [0, N-1]$ and $w_i, w_j, w_l \in V_o$.

This case was successfully applied, e.g., to the problem of an automatic control system (autopilot) for hydrofoil boats (Clark, 1978; Clark and Setzer, 1980; Clark and Walton, 1975). The results can be found in (Rubio Scola, 2000a; Rubio Scola *et al.*, 2000).

Case 2. Another type of the test signal of interest can be defined by

$$v(k+1) = v(k) + z(k), \quad k \in [0, N-1], \tag{61}$$

$$n(k) \leq v(k) \leq m(k)$$

with $k \in [0, N-1]$ and $z \in V_q$.

In order to reduce the problem to its original formulation, a state variable should be added to the system. Equations (1) are then transformed into

$$\begin{bmatrix} x_i(k+1) \\ v(k+1) \end{bmatrix} = \begin{bmatrix} A_i & b_i \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x_i(k) \\ v(k) \end{bmatrix} + \begin{bmatrix} B_i \\ 0 \end{bmatrix} u(k) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} z(k) + \begin{bmatrix} M_i \\ 0 \end{bmatrix} \nu_i(k), \quad (62)$$

$$y(k) = \begin{bmatrix} C_i & d_i \end{bmatrix} \begin{bmatrix} x_i(k) \\ v(k) \end{bmatrix} + D_i u(k) + N_i \nu_i(k) \quad (63)$$

with $i = s, f$.

Given z, a test signal constructed for the extended systems, the signal v for the original system is completely defined by (61) once $v(0)$ (or any $v(k)$) is fixed. Then, to design the test signal, initial conditions should be set to solve the linear programming problem (53), (54) with additional bounds on some components of v.

This case is suitable for optimizing the amplitude in those problems in which the amplitude is a relevant variable for exposing the failure modes.

Case 3. It is possible to consider other criteria for choosing a test signal z, e.g., imposing that $z \in V_q$ minimizes the following function:

$$\Big\| \sum_{k=0}^{N-1} z(k) \Big\|. \quad (64)$$

5. Exploiting Sparsity

The proposed algorithms have been successfully applied to some real problems like gas pressure in (Nikoukhah, 1998) and the automatic control system (autopilot) for hydrofoil boats (Clark and Setzer, 1980). The results can be seen in (Rubio Scola, 2000a; Rubio Scola *et al.*, 2000). In this section a simulation example is provided to illustrate the sparsity of the linear programming problem.

The results presented in what follows are taken from a random system having five state variables and two outputs. The following criterion is used to choose the test signal v:

$$-300 \leq v_k \leq 300. \quad (65)$$

The values of the test signal v_1 and λ are obtained by the method proposed in (59) and Algorithm 2.

Table 1. Size of the LP problem as a function of the length of the test signal N. Notation: nr is the number of equations, nc is the number of variables, nnz is the number of nonzero entries in the matrix of constraints (56) of the LP problem and nlp is the number of LP problems (55), (56) to solve (19) in this case.

N	1	2	3	4	5	6	7	8	9	10
nr	10	20	30	40	50	60	70	80	90	101
nc	21	41	61	81	101	121	141	161	181	201
nnz	20	40	60	80	100	120	140	160	180	293
nlp	2	4	6	8	10	12	14	16	18	1

For each test signal of the form (59), a linear programming problem (55), (56) has to be solved. In Table 1, the size of the linear problem to be solved as a function of the length of the test signal N is shown.

A feasible direction h for a separating hyperplane (30) has to be found. To calculate the hyperplane coefficients, the linear programming problem (19) is solved. The problem constraints (17) have 420 equations and 210 variables. The density of the matrix (17) is 0.0231293.

To determine the term d of the separating hyperplane (30), the four linear programming problems (29) have to be solved. The two linear programming problems associated with the failed system have 160 equations and 115 variables. The density of the associated matrix in (4) is 0.0277174. Analogously, the linear programming problems associated with the normal system have 150 equations and 104 variables. The matrix density is 0.0256410.

All the calculations were made using LIPSOL in Scilab. LIPSOL (Linear programming Interior-Point SOLvers) (Zhang, 1995), is a package that uses sparse-matrix data structure utilities to achieve both the program simplicity and computational efficiency.

The objective of using sparse matrix techniques for solving linear systems is to reduce computational costs by exploiting sparsity. It is possible to achieve drastic reductions in storage and arithmetic requirements when compared with the solutions of dense systems.

6. Example

An example of a very simple system is presented here. Equations (1) and (2) for this example are as follows:

6.1. System Model

For the normal system, with $i = s$, we have

$$x_s(k+1) = x_s(k) + 4v(k) + \nu_s(k),$$

$$\begin{aligned} y(k) &= x_s(k) + \nu_s(k), \\ \nu_s(k) &\leq 0.02, \\ -\nu_s(k) &\leq 0.02. \end{aligned} \tag{66}$$

For the failed system, with $i = f$, we define

$$\begin{aligned} x_f(k+1) &= x_f(k) + 4v(k) + \nu_f(k), \\ y(k) &= \nu_f(k), \\ \nu_f(k) &\leq 0.02, \\ -\nu_f(k) &\leq 0.02. \end{aligned} \tag{67}$$

Let w be the vector defined according to (3) by

$$\begin{aligned} w = \big[&x_s(0), x_f(0), y(0), \nu_s(0), \nu_f(0), x_s(1), x_f(1), \\ &y(1), \nu_s(1), \nu_f(1), x_s(2), x_f(2)\big]^T. \end{aligned} \tag{68}$$

From (5) it follows that

$$F = \begin{bmatrix} -1 & 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 & 0 \end{bmatrix},$$

$$p = \begin{bmatrix} -4 \\ 0 \\ -4 \\ 0 \\ 4 \\ 0 \\ 4 \\ 0 \end{bmatrix}, \tag{69}$$

$$E = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \end{bmatrix},$$

$$q = \begin{bmatrix} 0.02 \\ 0.02 \\ 0.02 \\ 0.02 \\ 0.02 \\ 0.02 \\ 0.02 \\ 0.02 \end{bmatrix}. \tag{70}$$

6.2. Test Signal Design

The first part of the problem consists in finding a signal v such that the set of possible input-output pairs for normal systems is disjoint from the set of possible input-output pairs for failed systems.

From Eqn. (40) we get

$$\begin{aligned} R &= \begin{bmatrix} 4. & 0. & 4. & 0. & 4. & 0. & 4. & 0. \end{bmatrix}, \\ p_0 &= \begin{bmatrix} 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \end{bmatrix}. \end{aligned} \tag{71}$$

According to (49), we have

$$Q = \text{diag}\{Q(0), Q(1), Q(2)\}, \tag{72}$$

where $Q(i) = [-1, 1]^T$, $i = 0, 1, 2$, and

$$q_v = [-1, 1, -1, 1, -1.1]^T. \tag{73}$$

Equation (48) defines the vertex set V_q

$$\begin{aligned} L &= \begin{bmatrix} -4. & 0. \\ 0. & 0. \\ 0. & 0. \\ 0. & 0. \\ 0. & 0. \end{bmatrix}, \quad h = \begin{bmatrix} 0.02 \\ 0.04 \\ 0.04 \\ 0.04 \\ 0.04 \end{bmatrix}, \\ G &= \begin{bmatrix} 0. & 0. & 1. & 0. & 0. & -1. & 0. & 1. \\ 0. & 0. & 0. & 0. & 1. & 1. & 0. & 0. \\ 1. & 1. & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 1. & 1. \\ 0. & 0. & 1. & 1. & 0. & 0. & 0. & 0. \end{bmatrix}. \end{aligned} \tag{74}$$

The linear programming problem (55), subject to the conditions (56), is now solved for each element of V_s until a vertex v_1 is found such that the problem has a solution. The value $\lambda = 0.015$ is obtained, with $v_1 = [-1, 1]$; $v = v_1$ is adopted.

6.3. Construction of a Separating Hyperplane

In the second part of the problem, for the test signal v found, and given an input-output pair, it must be recognized whether it belongs to the normal system's or to the failed system's input-output set.

The linear programming problem (19), subject to (17), is now solved with the matrices F, E, p, q previously defined and the matrices F_e, E_e defined as follows:

$$F_e = \begin{bmatrix} 0. & 0. \\ 0. & 0. \\ 0. & 0. \\ -1. & 0. \\ 0. & 0. \\ 0. & 0. \\ 0. & 0. \\ 0. & -1. \end{bmatrix}, \quad E_e = \begin{bmatrix} 0. & 0. \\ 0. & 0. \\ 0. & 0. \\ 0. & 0. \\ 0. & 0. \\ 0. & 0. \\ 0. & 0. \\ 0. & 0. \end{bmatrix}. \tag{75}$$

Once the solution (w^o, δ^o) is found, (27) can be written in terms of the active constraints given by the following lines of (17):

$$\begin{bmatrix} 1 & 4 & 6 & 10 & 16 & 19 & 21 & 24 \end{bmatrix}. \tag{76}$$

The matrix H in Lemma 4 is

$$H = \begin{bmatrix} -1 & 1 \end{bmatrix}. \tag{77}$$

Solving problems (29) yields

$$\begin{aligned} m^1 &= -3.98, \quad M^1 = -3.98, \\ m^2 &= -0.04, \quad M^2 = 0. \end{aligned} \tag{78}$$

The value $d = -2.01$ is selected, so that the hyperplane equation is given by

$$-x_1 + x_2 = -2. \tag{79}$$

7. Conclusions

The problem of active fault detection in linear systems subject to inequality bounded perturbations has been considered. Under certain conditions, there exist test signals v that can completely expose various failure modes of the system. A method of designing a filter for detecting and isolating failures in systems excited by such test signals has been presented.

The method proposed in this work involves injecting a test signal v into the system for a finite period of time, at some given time during the normal operation of the system. For the found test signal v, the detection filter determines whether or not the measured input-output is in the set of the normal systems or in the set of the failed systems, using a separating hyperplane test.

The complexity of both filter computation and test signal design can be important, since all the operations needed for this method are implemented by solving large linear optimization problems. To perform this computation, a Scilab Toolbox was developed by the first author. Details of the implementation and real application examples are provided in (Rubio Scola, 2000b). A numerical example for a simple system was given in this work in order to illustrate the proposed procedure.

The method presented here can also be applied to continuous-time linear dynamic systems with discrete-time measurements. Also, it can be applied to very large systems because it works with sparse matrices and linear optimization problems taking advantage of sparsity.

References

Basseville M. and Benveniste A., (Eds.) (1985): *Detection of Abrupt Changes in Signals and Dynamical Systems.* — Berlin: Sprinter.

Basseville M., Benveniste A., Moustakids G. and Rougée A. (1987): *Detection and diagnosis of changes in the eigenstructure of nonstationary multivariable system.* — Automatica, Vol. 23, No. 4, pp. 479–489.

Clark R.N. (1978): *Instrument fault detection.* — IEEE Trans. Aerospace Electron. Syst., Vol. AES–14, pp. 456–465.

Clark R.N. and Setzer W. (1980): *Sensor fault detection in a system with random disturbances.* — IEEE Trans. Aerospace Electron. Syst., Vol. AES–16, pp. 468–473.

Clark R.N. and Walton V.M. (1975): *Detecting instrument malfunctions in control systems.* — IEEE Trans. Aerospace Electron. Syst., Vol. AES-11, pp. 465–473.

Gertler J. (1998) *Fault Detection and Diagnosis in Engineering Systems.* — New York: Marcel Dekker.

Gondzio J. (1996): *Multiple centrality corrections in a primal-dual method for linear programming.* — Comput. Optim. Appl., Vol. 6, No. 2, pp. 137–156.

Isermann R. (1984): *Process fault detection based on modeling and estimation methods. A survey.* — Automatica, Vol. 20, No. 4, pp. 387–404.

Kerestecioğlu F. (1993): *Change Detection and Input Design in Dynamical Systems.* — Taunton: Research Studies Press.

Mangoubi R.S. (1998): *Robust Estimation and Failure Detection, A Concise Treatment.* — Berlin: Springer.

Mehrotra S. (1992): *On the implementation of a primal-dual interior point method.* — SIAM J. Optim., Vol. 2, No. 4, pp. 575–601.

Mironovski L.A. (1980): *Functional diagnosis of dynamic systems: A survey*. — Autom. Remote Contr., Vol. 41, pp. 1122–1143.

Nikoukhah R. (1998): *Guaranteed active failure detection and isolation for linear dynamical systems.* — Automatica, Vol. 34, No. 11, pp. 1345–1358.

Rockafellar R.T. (1972): *Convex Analysis.* — Princeton: Princeton University Press.

Rubio Scola H. (1999): *Solving large linear optimization problems with Scilab: Application to multicommodity problems.* — Techn. Rep. No. 0227, INRIA, Rocquencourt.

Rubio Scola H. (2000a): *Detection signal design for failure detection and isolation for linear dynamic systems.* — Techn. Rep. No. 3935, INRIA, Rocquencourt.

Rubio Scola H. (2000b): *Toolbox Scilab: Detection signal design for failure detection and isolation for linear dynamic systems. User's guide.* — Techn. Rep. No. 0241, INRIA, Rocquencourt.

Rubio Scola H., Nikoukhah R. and Delebecque F. (2000): *Detection signal design for failure detection: A linear linear programming approach.* — Proc. 4th Symp. *Fault Detection, Supervision and Safety for Technical Processes, Safeprocess'2000*, Budapest, Hungary, Vol. 1, pp. 593–598.

Uosaki K., Tanaka I. and Sugiyama H. (1984): *Optimal input design for autoregressive model discrimination with constrained output variance.* — IEEE Trans. Automat. Contr., Vol.AC–29, No. 4, pp. 384–350.

Willsky A.S. (1976): *A survey of design methods for failure detection in dynamic systems.* — Automatica, Vol. 12, No. 4, pp. 601–611.

Zhang X.J. (1989): *Auxiliary Signal Design in Fault Detection and Diagnosis.* — Heidelberg: Springer.

Zhang Y. (1995): *User's guide to LIPSOL.* — Dept. Mathematics and Statistics, University of Maryland, Baltimore County, USA.

Received: 20 August 2002
Revised: 2 June 2003