Detection Signal Design for Failure Detection and Isolation For Linear Dynamic System

Héctor E. Rubio Scola

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Detection signal design for failure detection and isolation for linear dynamic systems

Héctor E. Rubio Scola

No 3935

Mai 2000

THÈME 4

ISSN 0249-6399 ISRN INRIA/RR--3935--FR+ENG

Rapport de recherche

Detection signal design for failure detection and isolation for linear dynamic systems

Héctor E. Rubio Scola

Thème 4 — Simulation et optimisation
de systèmes complexes
Projet Meta2

Rapport de recherche n˚3935 — Mai 2000 — 51 pages

Abstract: We present a new methodology to calculate a filter that permits failure detection and isolation in a dynamic system. Assuming that the normal and the failed behaviors of a process can be modeled by two linear systems subject to inequality bounded perturbations, a method for on line implementation of a detection signal, guaranteeing detection of failure, is presented. To make the failures detectable, the injection of the test signal that improves the detectability property of failure in the dynamic process is proposed which achieves detectability on line. All the operations needed for our method are implemented by using large linear optimization problems. Examples are shown.

Key-words: failure detection, large scale programming, failure isolation, bounded perturbations, active detection.

(Résumé : tsvp)

CIUNR - FCEIA - Universidad Nacional de Rosario, Argentina.

Unité de recherche INRIA Rocquencourt
Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
Téléphone : 01 39 63 55 11 - International : +33 1 39 63 55 11
Télécopie : (33) 01 39 63 53 30 - International : +33 1 39 63 53 30

Conception de signal de détection et d'isolation de panne pour les systèmes dynamiques linéaires

Résumé : Nous présentons une nouvelle méthodologie pour calculer un filtre qui permet la détection et l'isolation de pannes dans un système dynamique. En supposant que le comportement du système normal et du système en panne d'un processus peut être modélisé par deux systèmes linéaires avec des contraintes bornées, une méthode pour trouver le signal de detection est presentée. Cette méthode, sous une condition de détectabilité, peut garantir la détection et l'isolation des pannes. Pour rendre les pannes discernables, on propose l'injection d'un signal de test qui réalise la détectabilité en ligne. Toutes les opérations nécessaires pour notre méthode sont résolues en utilisant la programmation linéaire creuse à grande dimension. Plusieurs exemples sont traités.

Mots-clé : détection de pannes, programmation grande échelle, isolement de pannes, perturbations bornées, détection

Contents

1 Introduction

The conception of failure detection systems entails the consideration of several points. Conceiving a system that will react rapidly when a failure occurs is what usually matters most interesting; yet, in systems with a high performance, one cannot generally tolerate important degradation in performance during normal system operation. That is why these two considerations are often conflicting. In other words, a system which is conceived to react quickly to sudden changes needs to be sensitive to some high frequency effects, which in turn will tend to increase the system's sensitivity to noise, through the occurrence of false alarms signals by the failure detection system. The difference between these conception issues is best studied if we take a precise example in which we can evaluate the costs of the various differences. For instance, false alarms will be better tolerated in a highly redundant system configuration than in a system deprived of significant backup capacities.

There are two ways of tackling the problem of failure detection and isolation. In first place, a passive approach: the detector monitors the inputs and the outputs of a system to know whether a failure has occurred and, if possible, what kind of failure. To achieve this, the measured input-output 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 of detecting changes in dynamical systems has been carefully studied (Willsky, 1976; Mironovrski, 1980; Isermann, 1984; Basseville and Benveniste, 1986; Clark 1986) in many field applications, to achieve failure detection in controlled systems or signal segmentation for recognition. Most of the time domain model-based methods use all the known or estimated model parameters to solve the two fundamental steps of change detection, that is to say residual generation and choice of the statistical decision function (Willsky, 1976).

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

The active approach to failure detection consists in acting upon the system on a periodic basis or at critical times using a test signal so as to show abnormal behaviors which would otherwise remain undetected during normal operation (Nikoukhah,1998). The detector can act either by taking over the usual inputs of the system or through a special input channel, perhaps modifying the structure of the system. The structure of the failure detection method considered in this paper is depicted in Figure 1.

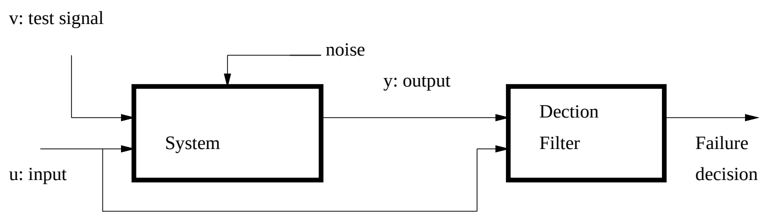

Figure 1: Active failure detection.

At some given time during 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 conception of test signals has been an important question in system identification, but their use to detect failures has been introduced in (Zhang, 1989, Kerestecioğlu, 1993, Uosaki et al., 1984). The test signals (called auxiliary inputs) in these works are regarded as linear inputs of stochastic models, and their objective is to optimize some statistical properties of the detector.

This work is oriented to the detection and isolation of faults in an active approach. We present a methodology to calculate a detector (filter) that allow us to make the detection and isolation of faults.

The selection of the test signal, as well as the calculation of the filter is computed off-line. The complexity of filter computation and design of the test signal is important when dealing with medium systems and long test signals. All the operations needed for our method are implemented by solving large linear optimization problems.

The mathematical operations that must make the filter to decide the good or bad operation of the system reduce to a multiplication and a sum of each input and each output, which is very easy to implement in real time. Examples are given in Section 4.

Structure of the failure detection and isolation

A structure of the failure detection and isolation method is proposed in this work at some given time during normal operation of the system, a test signal is injected into the system for a finite period of time. This signal is supposed to expose the failure modes of the system which are then detected by the detection filter.

The problem considered here has two parts:

Part 1: Find a signal v such that the possible input-output set of the normal system be disjoint to the possible input-output set of the failed system.

Part 2: For a found test signal v, given an input-output, recognize if this input-output is in the set of the normal system or in the set of the failed system.

The outline of the paper is as follows. In Section 2, basic assumptions are presented and the model is introduced. The solution to Part 2 is characterized in Section 3. This solution is computed off-line over a finite horizon, see for example Nikoukhah (1999). 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 real problems.

In this paper, we propose computationally implementable solutions, using algorithms for sparse matrices. We obtain a reduced time of computation for very large systems and long test signals, and we can use the proposed method to real problems.

The examples, one proposed by Nikoukhah (1998) and other proposed by Clark (1980) are given in Section 4.

The solution to Part 1 is only given in the case where the test signal enters the system linearly, the solution is considered in Section 5 and in Section 6 two examples are presented.

2 System model

We consider systems which can be modeled as follows:

$$\begin{array}{rcl} x_s(k+1) & = & A_s(k)x_s(k) + B_s(k)u(k) + b_s(k) + M_s(k)\nu_s(k) \\ y(k) & = & C_s(k)x_s(k) + D_s(k)u(k) + d_s(k) + N_s(k)\nu_s(k) \end{array} \tag{1}$$

for $k = 0, ..., N-1$, where $v = v(k), k \in [0, N-1]$ is a test signal , u and y are inputs and outputs which are measured on-line. A_s, B_s, C_s, D_s, M_s, N_s, b_s and d_s are matrices and vectors of appropriate dimensions which depend on v and we use the notation $A_s(k) = A_s(v(k))$, etc.

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

$$\begin{array}{rcl} R_{\nu s}(k)\nu_s(k) & \leq & p_{\nu s}(k) \\ R_{us}(k)u(k) & \leq & p_{us}(k) \end{array} \tag{2}$$

where $x_s(k) \in \Re^{n(k)}$, $u(k) \in \Re^{m(k)}$, $y(k) \in \Re^{l(k)}$, $\nu_s(k) \in \Re^{h(k)}$, $p_{\nu s}(k) \in \Re^{q(k)}$, $p_{us}(k) \in \Re^{j(k)}$, and $R_{\nu s}(k)$, $R_{us}(k)$ are given matrices of appropriate dimensions. The vectors $p_{\nu s}(k)$, $p_{us}(k)$ and the matrices $R_{\nu s}(k)$, $R_{us}(k)$ also depend on v, i.e. $R_{\nu s}(k) = R_{\nu s}(v(k))$, etc. No assumption is made on $R_{\nu s}$, and R_{us} except that (2) are consistent.

In the same way, for the failed system, we choose a similar model in the variables x_f, u, y, ν_f, and considered the system with failure as follows:

$$\begin{array}{rcl} x_f(k+1) & = & A_f(k)x_f(k) + B_f(k)u(k) + b_f(k) + M_f(k)\nu_f(k) \\ y(k) & = & C_f(k)x_f(k) + D_f(k)u(k) + d_f(k) + N_f(k)\nu_f(k) \end{array} \tag{3}$$

for $k = 0, ..., N-1$,

Perturbation ν_f and input u satisfy the inequality constraints:

$$\begin{array}{rcl} R_{\nu f}(k)\nu_F(k) & \leq & p_{\nu f}(k) \\ R_{uf}(k)u(k) & \leq & p_{uf}(k) \end{array} \tag{4}$$

where $x_f(k) \in \Re^{n_f(k)}$, $u(k) \in \Re^{m(k)}$, $y(k) \in \Re^{l(k)}$, $\nu_f(k) \in \Re^{h_f(k)}$, $p_{\nu f}(k) \in \Re^{q_f(k)}$, $p_{uf}(k) \in \Re^{j_f(k)}$ and $R_{\nu f}(k)$, $R_{uf}(k)$, are given matrices of appropriate dimensions. The vectors $p_{\nu f}(k)$, $p_{uf}(k)$ and the matrices $R_{\nu f}(k)$, $R_{uf}(k)$ also depend on v. No assumption is made on $R_{\nu f}$, R_{uf} except that (4) are consistent, for $k = 0, ..., N-1$.

The matrices and the vectors have not necessarily 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 mode, and the failed mode of the system can be modeled as in (1), (2), (3) and (4). But the system matrices can (and hopefully are) different for different 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 normal mode or failed mode; no transition occurs during the test period.

We assume 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 operating system (1),(2) and the constraints on the failed system (3), (4) are inconsistent. In Section 5, we show how to construct a signal v.

We introduce the vectors w, p, q, defined by

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

$$p(k) = \begin{bmatrix} b_s(k) \\ d_s(k) \\ b_f(k) \\ d_f(k) \end{bmatrix}, \quad q(k) = \begin{bmatrix} p_{\nu s}(k) \\ p_{us}(k) \\ p_{\nu f}(k) \\ p_{uf}(k) \end{bmatrix}$$

for $k = 0, ..., N-1$, and the following matrix F defined by the following scheme:

$$F = \begin{bmatrix} \boxed{F(0)} & & & & \\ & \boxed{F(1)} & & & \\ 0 & & \boxed{F(2)} & 0 & \\ & & & \ddots & \\ & & & & \boxed{F(N-1)} \end{bmatrix} \tag{5}$$

In the matrix F, the submatrices $F(k)$ and $F(k+1)$ share the columns corresponding to the variables $x_s(k+1)$ and $x_f(k+1)$, and

$$F(k) = \begin{bmatrix} -A_s(k) & 0 & 0 & -B_s(k) & -M_s(k) & 0 & I & 0 \\ -C_s(k) & 0 & I & -D_s(k) & -N_s(k) & 0 & 0 & 0 \\ 0 & -A_f(k) & 0 & -B_f(k) & 0 & -M_f(k) & 0 & I \\ 0 & -C_f(k) & I & -D_f(k) & 0 & -N_f(k) & 0 & 0 \end{bmatrix}$$

for $k = 0, 1, \ldots, (N-1)$.

Analogously we define E with a similar scheme:

$$E = \begin{bmatrix} \boxed{E(0)} & & & & \\ & \boxed{B(1)} & & & \\ 0 & & \boxed{E(2)} & & 0 \\ & & & \ddots & \\ & & & & \boxed{E(N-1)} \end{bmatrix} \tag{6}$$

where

$$E(k) = \begin{bmatrix} 0 & 0 & 0 & 0 & R_{\nu s}(k) & 0 & 0 & 0 \\ 0 & 0 & 0 & R_{us}(k) & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & R_{\nu f}(k) & 0 & 0 \\ 0 & 0 & 0 & R_{uf}(k) & 0 & 0 & 0 & 0 \end{bmatrix}$$

for $k = 0, 1, \ldots, (N-1)$.

(1), (2), (3) and (4) can be rewritten as:

$$\begin{array}{ccc} F & w & = & p \\ E & w & \leq & q \end{array} \tag{7}$$

We write in matrix form the equations and inequations for the operating system (1), (2) and the failed system (3), (4) separately:

- For the operating system :

$$\begin{array}{ccc} F_s & w & = & p_s \\ E_s & w & \leq & q_s \end{array} \tag{8}$$

We define F_s and E_s with a scheme similar to F and E, where

$$F_s(k) = \left[\begin{array}{cccccccc} -A_s k) & 0 & 0 & -B_s(k) & -M_s(k) & 0 & I & 0 \\ -C_s(k) & 0 & I & -D_s(k) & -N_s(k) & 0 & 0 & 0 \end{array} \right]$$

$$E_s(k) = \left[\begin{array}{cccccccc} 0 & 0 & 0 & 0 & R_{\nu s}(k) & 0 & 0 & 0 \\ 0 & 0 & 0 & R_{us}(k) & 0 & 0 & 0 & 0 \end{array} \right]$$

$$p_s(k) = \left[\begin{array}{c} b_s(k) \\ d_s(k) \end{array} \right], \quad q_s(k) = \left[\begin{array}{c} p_{\nu s}(k) \\ p_{us}(k) \end{array} \right].$$

- For the failed system :

$$\begin{array}{ccc} F_f & w & = & p_f \\ E_f & w & \leq & q_f \end{array} \tag{9}$$

We define analogously F_f and E_f, where

$$F_f(k) = \left[\begin{array}{cccccccc} 0 & -A_f(k) & 0 & -B_f(k) & -M_f(k) & 0 & I & 0 \\ 0 & -C_f(k) & I & -D_f(k) & -N_f(k) & 0 & 0 & 0 \end{array} \right]$$

$$E_f(k) = \left[\begin{array}{cccccccc} 0 & 0 & 0 & 0 & R_{\nu f}(k) & 0 & 0 & 0 \\ 0 & 0 & 0 & R_{uf}(k) & 0 & 0 & 0 & 0 \end{array} \right]$$

$$p_f(k) = \left[\begin{array}{c} b_f(k) \\ d_f(k) \end{array} \right], \quad q_f(k) = \left[\begin{array}{c} p_{\nu f}(k) \\ p_{uf}(k) \end{array} \right].$$

We define the following polyhedrons

$$S_s = \{w \ / \ w \text{ verifies } (8)\}, \quad S_f = \{w \ / \ w \text{ verifies } (9)\}.$$

3 Detection filter implementation

Let $v = v(k), k \in [0, N-1]$ be a test signal such that the solution sets of systems (8) and (9) have no intersection and let $w = w(k), k \in [0, N]$ be an observation of the state of the system. The detection problem consists now to decide whether this vector w is compatible with (8) or with (9).

Since inequalities (8) and (9) define two disjoint convex polyhedrons, our problem is reduced to know in which polyhedron the vector w stays. If a hyperplane can be found that separates the polyhedrons, it is sufficient to find at which side of the hyperplane the vector w lies. Our work is limited to find such an hyperplane. Its existence is guaranteed by the classical convexity theory.

The following theorem shows that we can obtain directly constraints involving only inputs u and outputs y for testing failure. Therefore we do not need to know the states variables x_s and x_f of the systems (1) and (3).

Theorem 1 *Let S and S_o be two non empty disjoint convex cylindrical polyhedrons defined by*

$$S = \{(x, x_o, y, u, \nu, \nu_o) \ / \ A_1 x + A_3 y + A_4 u + A_5 \nu \leq b\}$$

$$S_o = \{(x, x_o, y, u, \nu, \nu_o) \ / \ A_{o,2} x_o + A_{o,3} y + A_{o,4} u + A_{o,6} \nu_o \leq b_o\}$$

If we note $h_1 x + h_2 x_o + h_3 y + h_4 u + h_5 \nu + h_6 \nu_o = d$ an hyperplane which separates S and S_o, then $h_1 = h_2 = h_5 = h_6 = 0$, i.e. the hyperplane equation is

$$h_3 y + h_4 u = d \tag{10}$$

Proof:

Let us suppose that the hyperplane is defined by

$$h_1 x + h_2 x_o + h_3 y + h_4 u + h_5 \nu + h_6 \nu_o = d$$

where h_1, h_2, h_5, h_6 are not all zero. Let $(x^o, x^o_o, y^o, u^o, \nu^o, \nu^o_o) \in S_o$ then $\forall (x, \nu)$ the point $(x, x^o_o, y^o, u^o, \nu, \nu^o_o) \in S_o$.

It follows that:

$$h_1 x + h_2 x^o_o + h_3 y^o + h_4 u^o + h_5 \nu + h_6 \nu^o_o \geq d \tag{11}$$

Since $h_2x^o_o + h_3y^o + h_4u^o + h_6\nu^o_o$ is fixed, and $h_1x + h_5\nu$ can take any value, because x, ν are arbitrary, the expression (11) can take values $< d$, which contradicts the assumption that the hyperplane separates the two convex polyhedrons S and S_o. Thus $h_1 = 0$ and $h_5 = 0$. Analogously h_2 and h_6 must be zero.

□

Construction of separating hyperplane

The following lemma and its corollary show that it is possible to convert the problem separating two polyhedrons into an equivalent, problem separating of a polyhedron from the origin of coordinates.

Lemma 1 *Let S and S_o be two non-empty convex polyhedrons. There exists an hyperplane separating S and S_o, if and only if the convex polyhedron $S - S_o$ does not contain 0, i.e.,if and only if there exists an hyperplane separating 0 and the convex polyhedron $S - S_o$.*

Proof:

See Rockafellar [18] p.98 (Theorem 11.4)

Corollary 1 *Let S and S_o be two non-empty convex polyhedrons. Then, the hyperplane $hz = d$ separates S and S_o if and only if the hyperplane $hz_d = d_d$ separates 0 and the convex polyhedron $S - S_o$. i.e. a hyperplane separating S and S_o can be chosen parallel to a hyperplane separating 0 and $S - S_o$.*

Thanks to corollary 1, the problem that we are going to solve is to find an hyperplane that separates a polyhedron from the origin of coordinates. We will solve it by linear programming and taking into account the geometric property of the convex polyhedrons given in theorem 1.

The solution sets of systems (8) and (9) have no intersection then $S_s \cap S_f = \emptyset$. Let w_s and w_f such that $w_s \in S_s$ and $w_f \in S_f$, where

$$w_s(k) = \begin{cases} \left[x_s(k)^T, x_f(k)^T, y_s(k)^T, u_s(k)^T, \nu_s(k)^T, \nu_f(k)^T\right]^T & \text{if} \quad k \in [0, N-1] \\ \left[x_s(N)^T, x_f(N)^T\right]^T & \text{if} \qquad k = N \end{cases}$$

and

$$w_f(k) = \begin{cases} \left[x_s(k)^T, x_f(k)^T, y_f(k)^T, u_f(k)^T, \nu_s(k)^T, \nu_f(k)^T\right]^T & \text{if} \quad k \in [0, N-1] \\ \left[x_s(N)^T, x_f(N)^T\right]^T & \text{if} \qquad k = N \end{cases}$$

We note (y_s, u_s) as operating system output-input pair, i.e. in the equation (1) (of the operating system) we change the pair (y, u) by (y_s, u_s) obtaining :

$$\begin{array}{rcl} x_s(k+1) & = & A_s(k)x_s(k) + B_s(k)u_s(k) + b_s(k) + M_s(k)\nu_s(k) \\ y_s(k) & = & C_s(k)x_s(k) + D_s(k)u_s(k) + d_s(k) + N_s(k)\nu_s(k) \end{array} \tag{12}$$

for $k = 0, ..., N-1$.

Analogously (y_f, u_f) as failed system output-input pair, i.e. in the equation (3) (of the failed system) we change the pair (y, u) by (y_f, u_f) obtaining :

$$\begin{array}{rcl} x_f(k+1) & = & A_f(k)x_f(k) + B_f(k)u_f(k) + b_f(k) + M_f(k)\nu_f(k) \\ y_f(k) & = & C_f(k)x_f(k) + D_f(k)u_f(k) + d_f(k) + N_f(k)\nu_f(k) \end{array} \tag{13}$$

for $k = 0, ..., N-1$.

We also introduce the deviation between input-output pairs for normal and failed systems as follows :

$$e(k) = \begin{bmatrix} e_y(k) \\ e_u(k) \end{bmatrix} = \begin{bmatrix} y_s(k) \\ u_s(k) \end{bmatrix} - \begin{bmatrix} y_f(k) \\ u_f(k) \end{bmatrix} \tag{14}$$

We introduce the vector $\bar{w}$ defined by $\bar{w} = \begin{bmatrix} w \\ e \end{bmatrix}$

and the following matrices $\bar{F}$ and $\bar{E}$ defined by $\bar{F} = [F, F_e]$, $\quad \bar{E} = [E, E_e]$
where

$F_e = \text{diag}\{F_e(0), F_e(1), ..., F_e(N-1)\}$, and

$E_e = \text{diag}\{E_e(0), E_e(1), ..., E_e(N-1)\}$

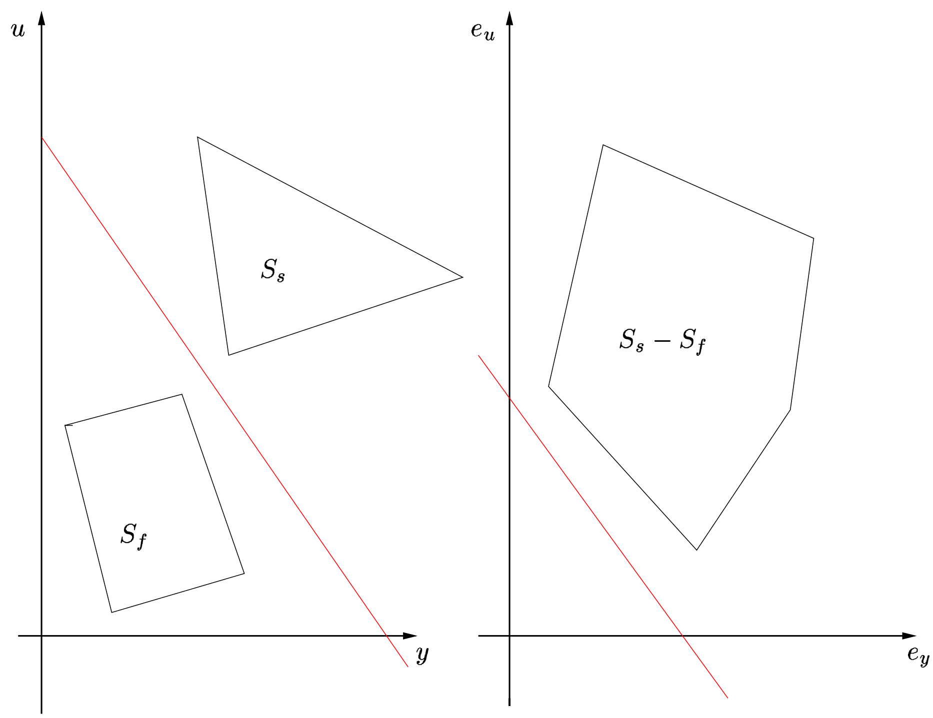

Figure 2: S_s and S_f properties

$$F_e(k) = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & B_f(k) \\ -I & 0 \end{bmatrix}, \quad E_e(k) = \begin{bmatrix} 0 & R_{\nu s}(k) \\ 0 & 0 \\ 0 & 0 \\ -R_{uf}(k) & 0 \end{bmatrix}$$

Using now (12), (2), (13), (4) and (14) we obtain :

$$\begin{array}{rcl} \bar{F}\bar{w} & = & p \\ \bar{E}\bar{w} & \leq & q \end{array} \tag{15}$$

We define

$$T = \{\bar{w}/\bar{w} \text{ verifies } (15)\}$$

We assume that the solution sets of systems (8) and (9) have no intersection, i.e. system (7) has no solution. Then (15) has no solution of the form $\begin{bmatrix} w \\ 0 \end{bmatrix}$ because (15) becomes (7) if $\bar{w} = \begin{bmatrix} w \\ 0 \end{bmatrix}$.

We introduce a relaxation parameter

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

in the equations (15), we obtain :

$$\begin{array}{rcl} Fw + F_e e & \leq & p + \delta_1 \\ -Fw - F_e e & \leq & -p + \delta_2 \\ Ew + E_e e & \leq & q + \delta_3 \end{array} \tag{17}$$

We define the following polyhedrons :

$$T_\delta = \{e/\bar{w} \text{ verifies } (17)\}$$

with following properties :

- $0 \notin T_\delta$ for $\delta = 0$.

- If $\delta^1 \geq \delta^2$ then $T_{\delta^1} \supseteq T_{\delta^2}$.
- The projection of $S_s - S_f$ on the e coordinate is T_δ for $\delta = 0$.

Chosing an adequate δ we can enlarge the polyhedron T_δ until $e = 0$ belongs to it. See figure 3.

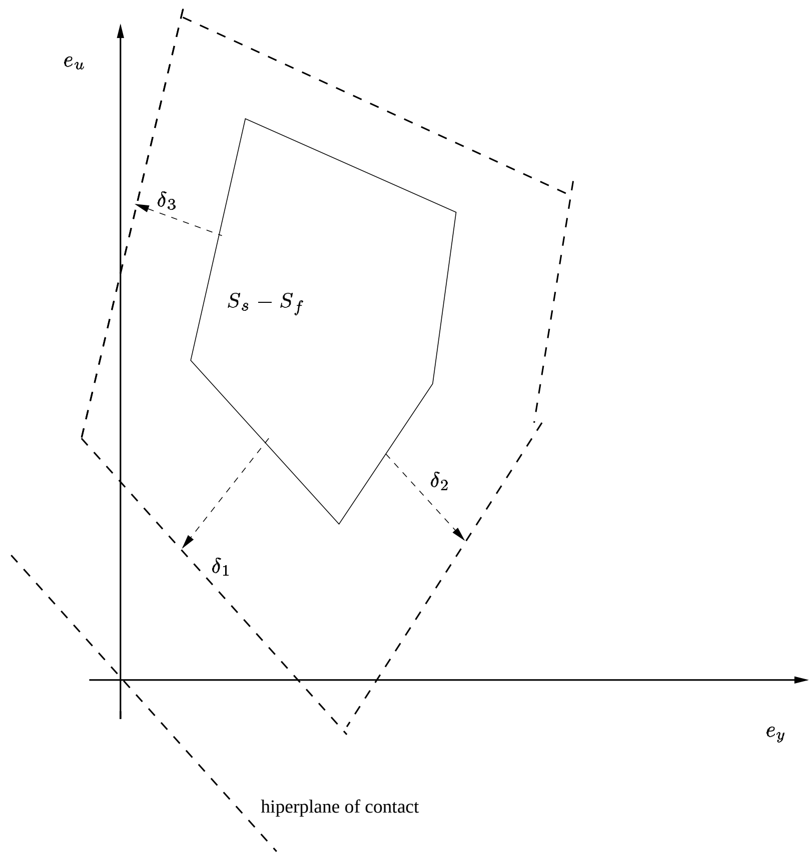

Figure 3: linear programming problem

Taking into account the polyhedron (17) we solve the following linear programming problem:

$$\min_{w,\delta} \sum_{j=1}^{3} \sum_{k=0}^{N-1} \delta_j(k) \tag{18}$$

subject to

$$\begin{aligned} Fw + F_e e &\leq p + \delta_1 && (19)\\ -Fw - F_e e &\leq -p + \delta_2 && (20)\\ Ew + E_e e &\leq q + \delta_3 && (21)\\ e &= 0 && (22)\\ \delta &\geq 0 && (23) \end{aligned}$$

Let $\delta^o = \left[\delta_1^{oT}, \delta_2^{oT}, \delta_3^{oT}\right]^T$ and w^o a solution to (18). Then permuting the rows of (19), (20) and (21) (evaluated at $(w^o, 0)$), in such a way that equality constraints appear first. We introduce the permutation matrix P_1, P_2, P_3 such that

$$P_1\left[F, F_e\right] = \begin{bmatrix} F_{q,1} & F_{e,q,1} \\ F_{i,1} & F_{e,i,1} \end{bmatrix}, \quad P_1 p = \begin{bmatrix} p_{q,1} \\ p_{i,1} \end{bmatrix}, \quad P_1 \delta_1^o = \begin{bmatrix} \delta_{q,1}^o \\ \delta_{i,1}^o \end{bmatrix} \tag{24}$$

We rewrite the equation (19) using $e = 0$ as follows :

$$\begin{array}{ccccccccc} F_{q,1}w^o & + & F_{e,q,1}0 & = & p_{q,1} & + & \delta_{q,1}^o \\ F_{i,1}w^o & + & F_{e,i,1}0 & < & p_{i,1} & + & \delta_{i,1}^o \end{array} \tag{25}$$

Analogously, we define P_2 and P_3 for the equations (20) and (21):

$$P_2\left[F, F_e\right] = \begin{bmatrix} F_{q,2} & F_{e,q,2} \\ F_{i,2} & F_{e,i,2} \end{bmatrix}, \quad P_3\left[E, E_e\right] = \begin{bmatrix} E_q & E_{e,q} \\ E_i & E_{e,i} \end{bmatrix} \tag{26}$$

$$\begin{array}{ccccccccc} -F_{q,2}w^o & - & F_{e,q,2}0 & = & p_{q,2} & + & \delta_{q,2}^o \\ -F_{i,2}w^o & - & F_{e,i,2}0 & < & p_{i,2} & + & \delta_{i,2}^o \end{array} \tag{27}$$

$$\begin{array}{ccccccccc} E_q w^o & + & E_{e,q}0 & = & q_q & + & \delta_{q,3}^o \\ E_i w^o & + & E_{e,i}0 & < & q_i & + & \delta_{i,3}^o \end{array} \tag{28}$$

We write the active constraints of (19), (20) and (21)

$$\begin{array}{rcrcrcl} F_{q,1}w^o & + & F_{e,q,1}0 & = & p_{q,1} & + & \delta^o_{q,1} \\ -F_{q,2}w^o & - & F_{e,q,2}0 & = & -p_{q,2} & + & \delta^o_{q,2} \\ E_q w^o & + & E_{e,q}0 & = & q_q & + & \delta^o_{q,3} \end{array} \tag{29}$$

The general equation for hyperplanes defined by these active constraints are :

$$\begin{array}{rcrcrcl} F_{q,1}w & + & F_{e,q,1}e & = & p_{q,1} & + & \delta^o_{q,1} \\ -F_{q,2}w & - & F_{e,q,2}e & = & -p_{q,2} & + & \delta^o_{q,2} \\ E_q w & + & E_{e,q}e & = & q_q & + & \delta^o_{q,3} \end{array} \tag{30}$$

We rewrite (30) as:

$$\phi w + \beta e = b + \delta^o_q \tag{31}$$

where

$$\phi = \begin{bmatrix} F_{q,1} \\ -F_{q,2} \\ E_q \end{bmatrix}, \quad \beta = \begin{bmatrix} F_{e,q,1} \\ -F_{e,q,2} \\ E_{e,q} \end{bmatrix}, \quad b = \begin{bmatrix} p_{q,1} \\ -p_{q,2} \\ q_q \end{bmatrix}, \quad \delta^o_q = \begin{bmatrix} \delta^o_{q,1} \\ \delta^o_{q,2} \\ \delta^o_{q,3} \end{bmatrix}$$

As we said before, the variables x, x_f, ν, ν_f do not appear in the hyperplane definition, because it is defined by (x, y, u, ν) and (x_f, y, u, ν_f), so the variables which participate in the two sets are (y, u) (Theorem 1).

The following theorem will show how transforming (31) to obtain an expression of the hyperplanes according to Theorem 1.

Lemma 2 *Let K be a full rank matrix whose columns span $\ker(\phi^T)$. If e and w satisfy the constraints $\beta e + \phi w = b + \delta^o_q$, then e satisfies $He = 0$, where $H = K^T \beta$.*

Proof:

We have $\xi \in Im(\phi) \iff (\ker(\phi^T)^T \xi = 0$. Multiplying by K^T both sides of (31) we get $K^T\ \beta e + K^T\ \phi w = K^T\ (b + \delta^o_q)$

The equation (31) for e = 0 is $\phi w = b + \delta^o_q$, then $K^T\ (b + \delta^o_q) = 0$, i.e. $He + 0 = 0$

□

Multiplying by K^T both members of equation (31) defines a set of hyperplanes $He = 0$ (Lemma 2). Considering (14) and Corollary 1, $H(y,u) = d$ defines a set of hyperplanes. It exists at least one that separates the two polyhedrons (8) and (9), which we note $H_i(y,u) = d_i$. Simply we have left to determine d_i. To do that, we calculate the tangent hyperplanes to the two convex polyhedrons S_s and S_f.

$$\begin{array}{ll} m_i^1 = \min\limits_{s.t. 9} H_i(y,u) & M_i^1 = \max\limits_{s.t. 9} H_i(y,u) \\ m_i^2 = \min\limits_{s.t. 8} H(y,u) & M_i^2 = \max\limits_{s.t. 8} H(y,u) \end{array} \tag{32}$$

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 $i = 1, ..., n_p$.

Then the hyperplane equation will be

$$H_i(y,u) = d_i. \tag{33}$$

Theorem 2 *There exists $i \in [1, n_p]$ such that $H_i(y,u) = d_i$ separates the two convex the polyhedrons S_s and S_f.*

Proof:

We remark that

$$\text{if } \begin{bmatrix} w \\ e \end{bmatrix} \in T \text{ then } He \neq 0 \tag{34}$$

Indeed, let us suppose that

$$\begin{bmatrix} w \\ e \end{bmatrix} \in T \text{ and } He = 0 \tag{35}$$

If $\begin{bmatrix} w \\ e \end{bmatrix} \in T$ then $\beta e + \phi w \leq b$

Let b_1 such that

$$\beta e + \phi w = b_1 \leq b \tag{36}$$

Multiplying by K^T both sides of (36), we get

$$\begin{array}{rcl} K^T\beta e + K^T\phi z & = & K^T b_1 \\ He & = & K^T b_1 \end{array}$$

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

$$\phi w_1 = b_1 \leq b \tag{37}$$

δ_q^o is solution of the linear programming problem (18), i.e. $\delta_q^o \geq 0$ is the minimal l_1 norm such that $\phi w \leq b + \delta_q^o$ has solution. $\delta_q^o \neq 0$ since (7) has no solution. It follows that the inequality $\phi w \leq b$ has no solution, i.e. (37) does not hold, i.e. (35) is false. We can conclude that (34) holds.

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

$$H_i(S_s) \cap H_i(S_f) = \emptyset$$

Since S_s and S_f are convex, $H_i(S_s)$ and $H_i(S_f)$ are convex, and we have $H_i(y_s, u_s) < H_i(y_f, u_f)$ or $H_i(y_s, u_s) > H_i(y_f, u_f)$

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

$$H_i(y_s, u_s) \leq \underset{(y_s,u_s)\in S_s}{max H_i(y_s, u_s)} < d_i < \underset{(y_f,u_f)\in S_f}{min H_i(y_f, u_f)} \leq H_i(y_f, u_f)$$

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

$$H_i(y_s, u_s) \geq \underset{(y_s,u_s\in S_s}{min H_i(y, u)} > d_i > \underset{(y_f,u_f)\in S_f}{max H_i(y_f, u_f)} \geq H_i(y_f, u_f)$$

It follows that the hyperplane defined by $H_i(y, u) = d_i$, separates the two convex polyhedrons S_s and S_f.

□

4 Examples

4.1 Example 1

The model considered here represents a gas chamber [15]. We will indicate with f_i the gas flow in , and with f_o the gas flow out. The chamber is supposed to have a uniform pressure P_i. A sensor is placed inside the chamber for measuring this pressure.

Based on this measurement, a central computer may decide to open the emergency valve which releases gas into the atmosphere. This happens in emergency situation, for example if P_i goes above a critical threshold.

The problem here is that emergency situations are rare and the emergency valve which stays closed over long periods of time may not be functioning when needed. For this reason, on a periodic basis, the emergency valve should be tested. The central computer does that by periodically sending a signal to open the emergency valve and monitoring the pressure P_i to decide whether or not the valve has effectively been opened.

The following assumptions are made. Every Δt seconds, the central computer can either read the output of the pressure sensor, or send a signal to open (or keep open if already open) to the valve mechanism. During the test period the valve mechanism is either broken or functional; we exclude the case were the mechanism can brake down or be repaired in middle of the test. The gas temperature is constant during the test period and the gas can be modeled as an ideal gas. The flow f_i is bounded above and below, i.e., $0 < a < f_i < b$, and the output flow is given by $f_o = \alpha(P_i - P_o)$, where P_o (the pressure outsize the chamber) is supposed to be constant but unknown during the test period.

The state vector is

$$x(k) = \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} = \begin{bmatrix} P_i(k\Delta t) \\ P_o(k\Delta t) \end{bmatrix} = \begin{bmatrix} \text{pressure inside the chamber} \\ \text{pressure outside the chamber} \end{bmatrix}$$

and the output vector is

$$y(k) = \begin{cases} \begin{bmatrix} \text{pressure in chamber} \end{bmatrix} & \text{if } v(k) = 0 \\ [\,]^{1} & \text{if } v(k) \neq 0 \end{cases}$$

$v(k)$ denotes the test signal wich can take the values 0 and 1. When $v(k) = 0$, the pressure is measured and the valve is closed. When $v(k) = 1$, a signal to open the emergency valve (or to keep open in case it is already open) is sent but pressure is not measured.
The sample period is $\Delta t = 0.1$ s. The resulting matrices and vectors of equations (1) and (2) are given by

$$A_s(k) = \begin{bmatrix} O(k) & \alpha/(\alpha+\beta)(1-O(k)) \\ 0 & 1 \end{bmatrix} \quad B_s(k) = [\,]$$

$$M_s(k) = \begin{cases} \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} & \text{if } v(k) = 0 \\ \begin{bmatrix} 1 \\ 0 \end{bmatrix} & \text{if } v(k) \neq 0 \end{cases} \qquad b_s(k) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$C_s(k) = \begin{cases} \begin{bmatrix} 1 & 0 \end{bmatrix} & \text{if } v(k) = 0 \\ [\,] & \text{if } v(k) \neq 0 \end{cases} \qquad D_s(k) = [\,]$$

$$N_s(k) = \begin{cases} [0, 1] & \text{if } v(k) = 0 \\ [\,] & \text{if } v(k) \neq 0 \end{cases} \qquad d_s(k) = \begin{cases} 0 & \text{if } v(k) = 0 \\ [\,] & \text{if } v(k) \neq 0 \end{cases}$$

$$R_{us}(k) = [\,] \qquad p_{us}(k) = [\,]$$

$$R_{\nu s}(k) = \begin{cases} \begin{bmatrix} 1 & 0 \\ -1 & 0 \\ 0 & 1 \\ 0 & -1 \end{bmatrix} & \text{if } v(k) = 0 \\ \begin{bmatrix} 1 \\ -1 \end{bmatrix} & \text{if } v(k) \neq 0 \end{cases}$$

[1]We note with [ ] the empty matrix

$$p_{\nu s}(k) = \begin{cases} \begin{bmatrix} 20/(\alpha(1-O(k))) \\ -a/(\alpha(1-O(k))) \\ c \\ -c \end{bmatrix} & \text{if } v(k)=0 \\ \begin{bmatrix} 20/((\alpha+v(k)\beta)(1-O(k))) \\ -a/((\alpha+v(k)\beta)(1-O(k))) \\ 1 \\ -1 \end{bmatrix} & \text{if } v(k)\neq 0 \end{cases}$$

The function $O(k)$ is defined by:

$O(k) = exp(-(v(k)\beta + \alpha)\Delta t)$

and

$a = 12.5,\ \alpha = 3; c = 1,\ \beta = 10.$

The constraints on initial conditions are $x_2(0) > 0$ (pressure cannot be negative) and $x_1(0) > P_{i,m}$ (corresponds to the lowest steady state pressure P_i with the emergency valve closed). This clearly corresponds to the situation where $f_i = a$, and can be obtained by noting that in steady state $f_i = f_o$, which implies that

$$\alpha(P_{i,m} - P_o) = a$$

$$\begin{cases} x_1(0) - x_2(0) & > & a/\alpha \\ x_2(0) & > & 0 \end{cases}$$

For the system with fault whose equations are (3)and (4), the matrices and vectors are given by

$$A_f = \begin{bmatrix} O(0) & (1-O(0)) \\ 0 & 1 \end{bmatrix} \qquad B_f(k) = [\,]$$

$$M_f(k) = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \qquad b_f(k) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$C_f(k) = \begin{cases} \begin{bmatrix} 1 & 0 \end{bmatrix} & \text{if } v = 0 \\ [\,] & \text{if } v \neq 0 \end{cases} \qquad D_f(k) = [\,]$$

$$N_f(k) = \begin{cases} [0,1] & \text{if } v(k) = 0 \\ [\,] & \text{if } v(k) \neq 0 \end{cases} \qquad d_f(k) = \begin{cases} 0 & \text{if } v(k) = 0 \\ [\,] & \text{if } v(k) \neq 0 \end{cases}$$

$$R_{uf}(k) = [\,] \qquad p_{uf}(k) = [\,]$$

$$R_{\nu f}(k) = \begin{cases} \begin{bmatrix} 1 & 0 \\ -1 & 0 \\ 0 & 1 \\ 0 & -1 \end{bmatrix} & \text{if } v(k) = 0 \\ \begin{bmatrix} 1 & 0 \\ -1 & 0 \end{bmatrix} & \text{if } v(k) \neq 0 \end{cases}$$

$$p_{\nu f}(k) = \begin{cases} \begin{bmatrix} 20/(\alpha(1-O(k))) \\ -a/(\alpha(1-O(k))) \\ c \\ -c \end{bmatrix} & \text{if } v(k) = 0 \\ \begin{bmatrix} 20/((\alpha+v(k)\beta)(1-O(k))) \\ -a/((\alpha+v(k)\beta)(1-O(k))) \\ 1 \\ -1 \end{bmatrix} & \text{if } v(k) \neq 0 \end{cases}$$

The constraints on initial conditions if the valve has failed are :

$$\begin{cases} x_{f1}(0) - x_{f2}(0) & > & a/\alpha \\ x_{f2}(0) & > & 0 \end{cases}$$

The simple test signal $v = [0, 1, 1, 1, 0]$, is such that the above constraints are mutually exclusive.

Solving the linear programming problem (18) and applying the theorem 2 we obtain:

$$d = -0.5959724y(0) + 2.4596031y(4) \tag{38}$$

The value of d can be determined by solving the linear programming problems:

$$\begin{array}{lll} d_{max} = & \max & \{-0.5959724y(0) + 2.4596031y(4)\} \\ & \text{s.t.}(9) & \end{array}$$

and

$$\begin{array}{lll} d_{min} = & \min & \{-0.5959724y(0) + 2.4596031y(4)\} \\ & \text{s.t. } (8) & \end{array}$$

We obtain $d_{max} = 4.7095532$ and $d_{min} = 4.5194724$. So any d satisfying $d_{min} \leq d \leq d_{max}$ can be used.

Then the hyperplane equation is:

$$-0.5959724y(0) + 2.4596031y(4) = 4.6145128 \tag{39}$$

Solving (32), with the output y of a system that works correctly we obtain from the equation (38) the values d between -559.21481 and 4.5194724. For the abnormal system the values of d will be placed between 4.7095532 and 1865.8638.

4.2 Example 2

In this example we report the automatic control system (autopilot) for a hydrofoil boat. A sketch of the basic boat, which employs submerged hydrofoils, is given in Figure 4. A detailed description of this boat and its autopilot appears in [4], [5] and [6].

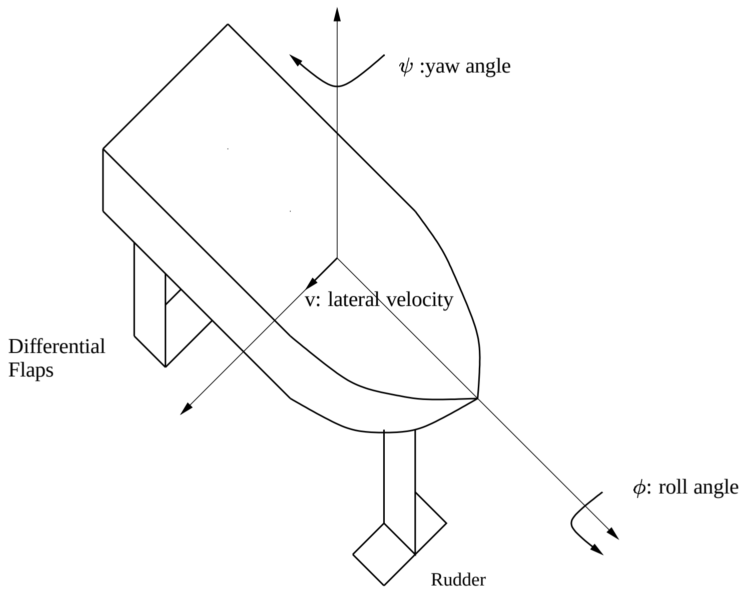

Figure 4: Boat with submerged hydrofoils

The model considered here is the hydrofoil boat in its nominal cruise condition of 45 knots speed and 6 foot foil depth. The state vector is

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} P \\ \phi \\ R \\ v \end{bmatrix} = \begin{bmatrix} \text{roll rate, } deg/sec \\ \text{roll angle, } deg \\ \text{yaw rate, } deg/sec \\ \text{lateral velocity, } ft/sec \end{bmatrix} \tag{40}$$

where $P(t) = \dot{\phi}(t)$, $R(t) = \dot{\psi}(t)$ and $v(t)$ is the component of the velocity of the center of mass (with respect to the Earth) along the lateral axis. The yaw, roll, and sway motions of the boat are controlled by the forward rudder and the differentially operated flaps at the rear ones. We assume for simplicity that we have only single flaps on each side with deflection $\delta_p(t)$ for the port flap and $\delta_s(t)$ for the starboard flap. We further assume that the actuators for these flaps are exactly coordinated so that $\delta_p(t) = -\delta_s(t)$. The control vector is:

$$u = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \begin{bmatrix} \delta_A \\ \delta_r \end{bmatrix} = \begin{bmatrix} \text{horizontal flag deflection, } deg \\ \text{rudder deflection, } deg \end{bmatrix} \tag{41}$$

There are four sensors providing feedback signals and they are monitored. The four instrument signals constitute the instrument output vector

$$y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} = \begin{bmatrix} a_{PI} \\ \phi_I \\ R_I \\ a_{LI} \end{bmatrix} = \begin{bmatrix} \text{indicated port acceleration , } ft/s^2 \\ \text{indicated roll angle , } deg \\ \text{indicated yaw rate , } deg/sec \\ \text{indicated lateral acceleration , } ft/s^2 \end{bmatrix} \tag{42}$$

The difference equations for the system model are:

$$\begin{array}{rcl} x_s(k+1) & = & Ax_s(k) + Bu(k) + M\nu_s(k) \\ y(k) & = & Cx_s(k) + Du(k) + N\nu_s(k) \end{array} \tag{43}$$

$$R_{\nu s}\nu_s(k) \quad \leq \quad p_{\nu s} \tag{44}$$

In (43) the terms $M\nu_s(k)$ and $N\nu_s(k)$ represents the disturbance bounded by (44).

The autopilot (control laws and actuators) generates the control input u from the four feedback signals and the input command θ_h (helm command) in a conventional manner [5].

It is assumed that the autopilot and instrument failure detection will be realized with digital computers, so a discrete time model of autopilot system is used here. The difference equation for the autopilot model are:

$$\begin{array}{rcl} u(k) & = & Jq(k) \\ q(k+1) & = & Fq(k) + Gy(k) + H\theta_h(k) \end{array} \tag{45}$$

where the matrices A, B, C, D, M, N, R_ν and p_ν are listed in the Appendix. The sample period is $\Delta t = 0.02$ s.

It is assumed that two sensors consisting of roll axis gyro and yaw rate gyro are used to measure the state of the system, and the initial state of the system is taken to be

$$x^T(0) = \begin{bmatrix} 0. & 0. & 0. & 0. \end{bmatrix}$$

4.2.1 Case 1

We shall consider failed systems with can be modeled as follows:

$$\begin{array}{rcl} x_s(k+1) & = & Ax_s(k) + Bu(k) + M\nu_s(k) \\ y(k) & = & Cx_s(k) + Du(k) + N\nu_s(k) \quad + \text{[fault vector]} \end{array} \tag{46}$$

$$R_{\nu s}\nu_s(k) \;\; \leq \;\; p_{\nu s} \tag{47}$$

The “fault vector” represents the effect of the instrument failures which are to be detected and identified.

We choice a test signal the control vector u, therefore in (46) if $v = u$ then $B(v(k)) = Bv(k)$, $d_s(v(k)) = dv(k)$ and $d_f(v(k)) = Dv(k) + \text{[fault vector]}$. Now “fault vector” was chosen to system with failure:

$$\text{[fault vector]} = \begin{bmatrix} 0. & 0. & 0.6 & 0. \end{bmatrix}^T$$

The respective two candidate models are:

Failed system model

$$\begin{array}{rcl} x_f(k+1) & = & Ax_f(k) + b(v(k)) + M\nu_f(k) \\ y(k) & = & Cx_f(k) + d_f(v(k)) + N\nu_f(k) \end{array} \tag{48}$$

$$R_\nu \nu_f(k) \;\; \leq \;\; p_\nu \tag{49}$$

Operating system model

$$\begin{array}{rcl} x_s(k+1) & = & Ax_s(k) + b(v(k)) + M\nu_s(k) \\ y(k) & = & Cx_s(k) + d_s(v(k)) + N\nu_s(k) \end{array} \tag{50}$$

$$R_\nu \nu_s(k) \;\; \leq \;\; p_\nu \tag{51}$$

The fault is detected in $\Delta t = 0.02s.$, the simple test signal would be

$$V = \begin{bmatrix} 0. \\ 0. \end{bmatrix} \tag{52}$$

So to decide whether or not a failure has occured, a possible test is

$$d = -0.0599478y_1(0) + y_3(0) + 0.3772902y_4(0) \tag{53}$$

The value of d can be determined by solving the linear programming problems (32), it follows that:

$$\begin{array}{lcccl} \text{If} & -0.0011959 & \leq d \leq & 0.0027537 & \text{The system works normally} \\ \text{If} & 0.5988041 & \leq d \leq & 0.6027537 & \text{The system works with anomalies} \end{array}$$

Then the hyperplane equation will be

$$-0.0599478y_1(0) + y_3(0) + 0.3772902y_4(0) = 0.3007789 \tag{54}$$

The autopilot generates the control input u (figure 5) from the four feedback signals and the input command θ_h. The figure 6 shows the dynamic response of the boat (four output), curve A is the response to $\theta_h = 17$ deg. step input and the disturbance null $w(k) = 0$ and no instruments faults. The curve B is same as A, but with disturbance $w(k) \neq 0$, the curve C is for instruments faults and disturbance $w(k) \neq 0$.

The random sequence $w(k)$ was chosen to represent Gaussian white stationary noise with zero mean value and of such a intensity as to cause noticeable random fluctuations in the state variable responses to the standard $\theta_h = 17deg$ command.

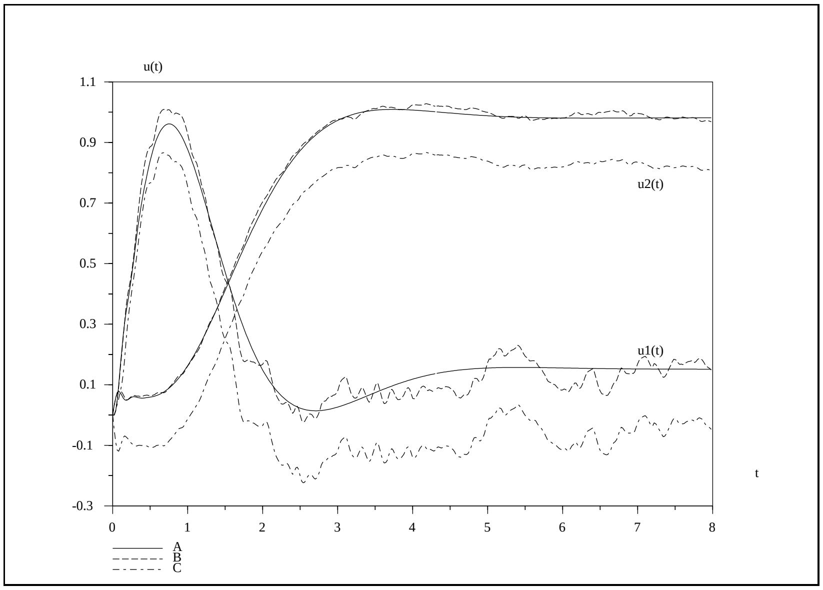

Figure 5: Control variables u to a step of 17 deg.

4.2.2 Case 2

Failed system model

$$\begin{array}{rcl} x_s(k+1) & = & Ax_s(k) + b(v(k)) + M\nu_s(k) \\ y(k) & = & C_f x_s(k) + d(v(k)) + N\nu_s(k) \end{array} \tag{55}$$

$$R_\nu \nu_f(k) \quad \leq \quad p_\nu \tag{56}$$

where

$$C_f = \begin{bmatrix} 2.8 & 0.149 & 1.01 & -7.03 \\ 0. & 1. & 0. & 0. \\ 0. & 0. & 1. & 0. \\ 0. & 0. & 0. & 0. \end{bmatrix}$$

The null line of C_f represents the effect of disconnect of the lateral indicated acceleration.

The fault is detected in $4\Delta t = 0.08s.$, the simple test signal would be

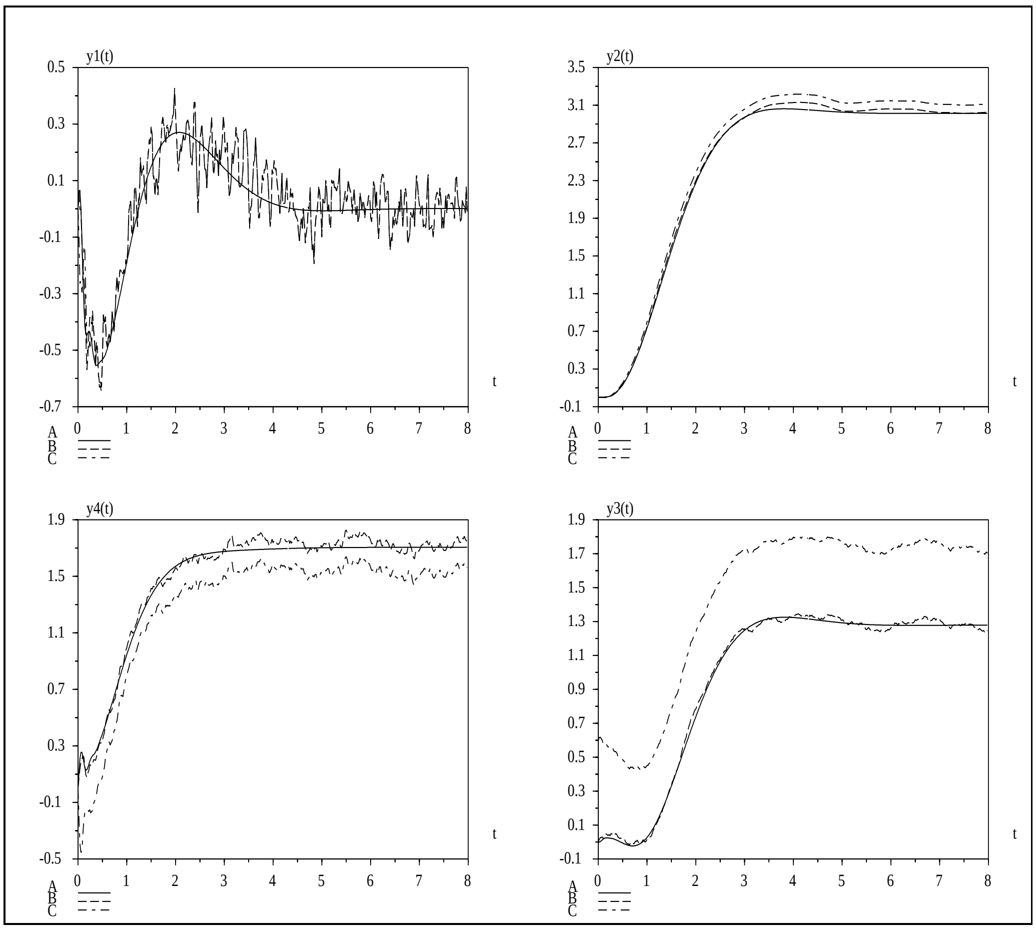

Figure 6: Dynamic responses of the boat to a step of 17 deg.

$$V = \begin{bmatrix} 0. & .2 & .2 & 0. \\ 0. & 1. & 1. & 0. \end{bmatrix} \tag{57}$$

Then the hyperplane equation will be

$$0.0004474 y_4(0) - 0.8358254 y_4(1) + 0.9789195 y_4(3) = -3.1432313 \tag{58}$$

The autopilot generates the control input u (figure 7) from the four feedback signals and the input command $\theta_h = 17deg.$. The figure 8 shows the dynamic response of the boat (y_1, y_2, y_3, y_4). The curves A are response with disturbance $w(k) = 0$ and no instruments faults. the curves B are same as A, but with $w(k) \neq 0$, the curves C are the response with disturbance no null and instrument faults.

The random sequence $w(k)$ was chosen to represent Gaussian white stationary noise with zero mean value and of such an intensity as to cause noticeable random fluctuations in the state variable responses to the standard $\theta_h = 17deg$ command.

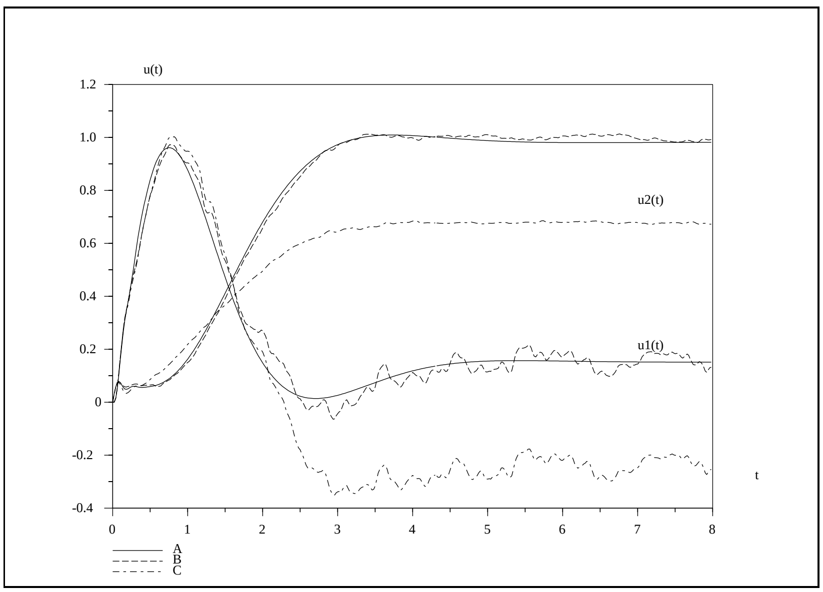

Figure 7: Control variables u to a step of 17 deg.

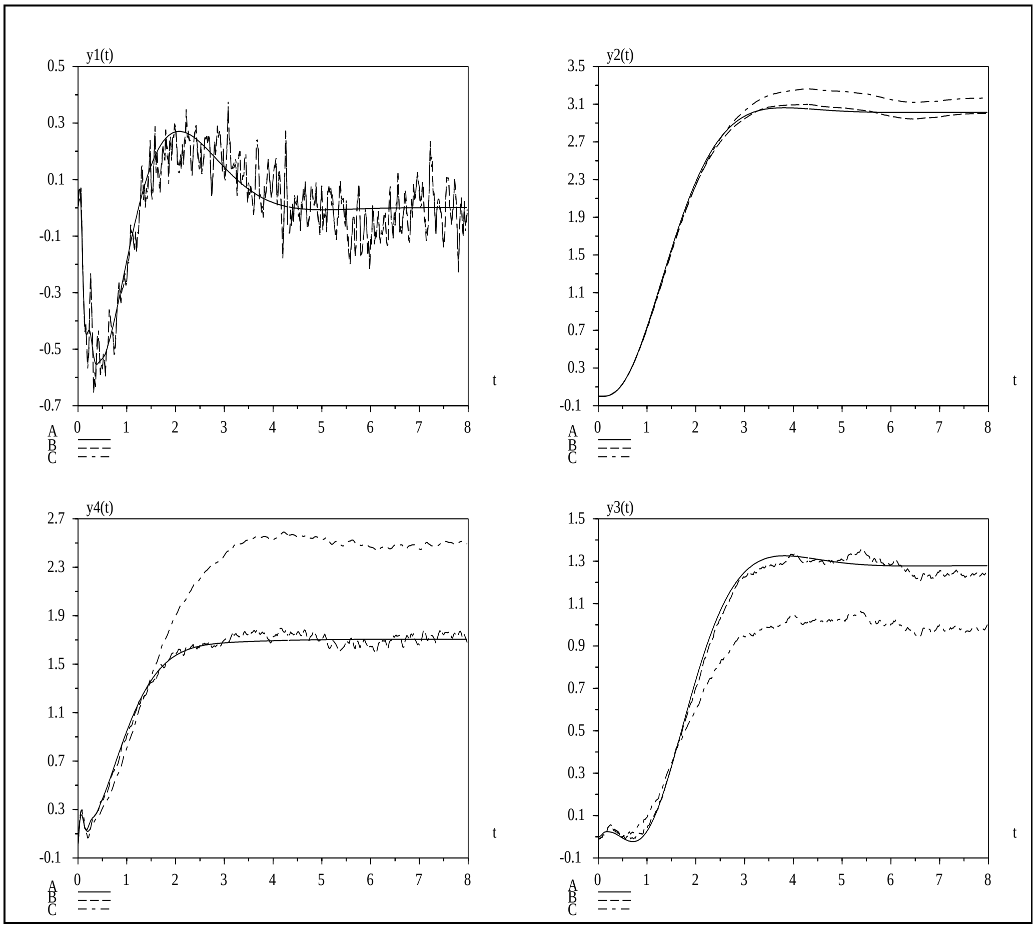

Figure 8: Dynamic responses of the boat to a step of 17 deg.

5 Test signal design

In this section, we show how can we find a detection horizon N and construct a test signal $v = \{v(k), k \in [0, N-1]\}$, as short as possible, such that (1), (2) and (3), (4) are mutually exclusive, i.e. $S_f \cap S_s = \emptyset$.

The solution to this problem is only given in 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 [27]. We show how a test signal can be designed for a special class of Model (1), (2), (3) and (4). We assume 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

$$\begin{array}{rcl} b_s(v(k)) & = & b_{s,1}(k)v(k) + b_{s,o}(k) \\ d_s(v(k)) & = & d_{s,1}(k)v(k) + d_{s,o}(k) \end{array} \tag{59}$$

$$\begin{array}{rcl} b_f(v(k)) & = & b_{f,1}(k)v(k) + b_{f,o}(k) \\ d_f(v(k)) & = & d_{f,1}(k)v(k) + d_{f,o}(k) \end{array} \tag{60}$$

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, for $i = s, f$. Then (1), (2), (3) and (4) can be rewritten as:

$$\begin{array}{ccccc} Fw & + & Rv & = & p_o \\ Ew & & & \leq & q \end{array} \tag{61}$$

where

$R = \text{diag}\{r(0), r(1), ..., r(N-1)\}$,

$$r(k) = \begin{bmatrix} -b_{s,1}(k) \\ -d_{s,1}(k) \\ -b_{f,1}(k) \\ -d_{f,1}(k) \end{bmatrix} \quad p_0(k) = \begin{bmatrix} b_{s,o}(k) \\ d_{s,o}(k) \\ b_{f,o}(k) \\ d_{f,o}(k) \end{bmatrix}$$

for $k = 0, ..., N-1$.

The problem is then to find v such that (61) is not satisfied. To solve it, we use the classical convexity theory.

Consider the polyhedron:

$$S_h = \{v \;/\; \exists w \;/\; (w, v) \text{ verifies } (61)\}$$

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).

We rewrite (61) as

$$\begin{array}{ccccc} Fw & + & Rv & = & p_o \\ Ew & + & \delta & = & q \\ & & \delta & \geq & 0 \end{array} \tag{62}$$

Lemma 3 *Let K be a full rank matrix whose columns span $\ker([F^T, E^T])$. If v and δ satisfy (62) then v and δ satisfy :*

$$\begin{array}{ccccc} Lv & + & G\delta & = & h \\ & & \delta & \geq & 0 \end{array} \tag{63}$$

where

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

If $\ker([F^T, E^T]) = 0$ *then* $\forall\ v$ *(61) has solution.*

The problem that we are going to solve is to find a v that does not satisfy (63). Let L_i be the i-th row of matrix L, we introduce the following linear programming problem :

$$\max_{v,\delta} L_i v \tag{64}$$

subject to

$$\begin{array}{ccccc} Lv & + & G\delta & = & h \\ & & \delta & \geq & 0 \end{array} \tag{65}$$

Let (v^o, δ^o) be a solution to (64) and we note $h_i^o = L_i v^o$, then

$$L_i v = h_i^o$$

is a equation of the tangent hyperplane to the polyhedron S_h. Then we can chose v such that

$$L_i v > h_i^o$$

It is possible to consider many criteria for choosing a detection signal v , among the v 's for which (61) has no solution : for example, we can choose a minimal norm v. We will consider here the following constraints :

$$Q(k)v(k) \quad \leq \quad q_v(k) \tag{66}$$

for $k = 0, ..., N-1$, which can be rewritten as :

$$Qv \quad \leq \quad q_v \tag{67}$$

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

We define the polyhedron :

$$S_q = \{v \;/\; v \text{ verifies } (67)\}\,.$$

We note that there exists no v satisfying (67) and not satisfying (61) if and only if

$$S_q \subseteq S_h. \tag{68}$$

Using the convexity of S_q, we have :

Lemma 4 *Let V_q be the set of vertices of S_q. There exists no v satisfying (67) and not satisfying (61) if and only if*

$$V_q \subseteq S_h. \tag{69}$$

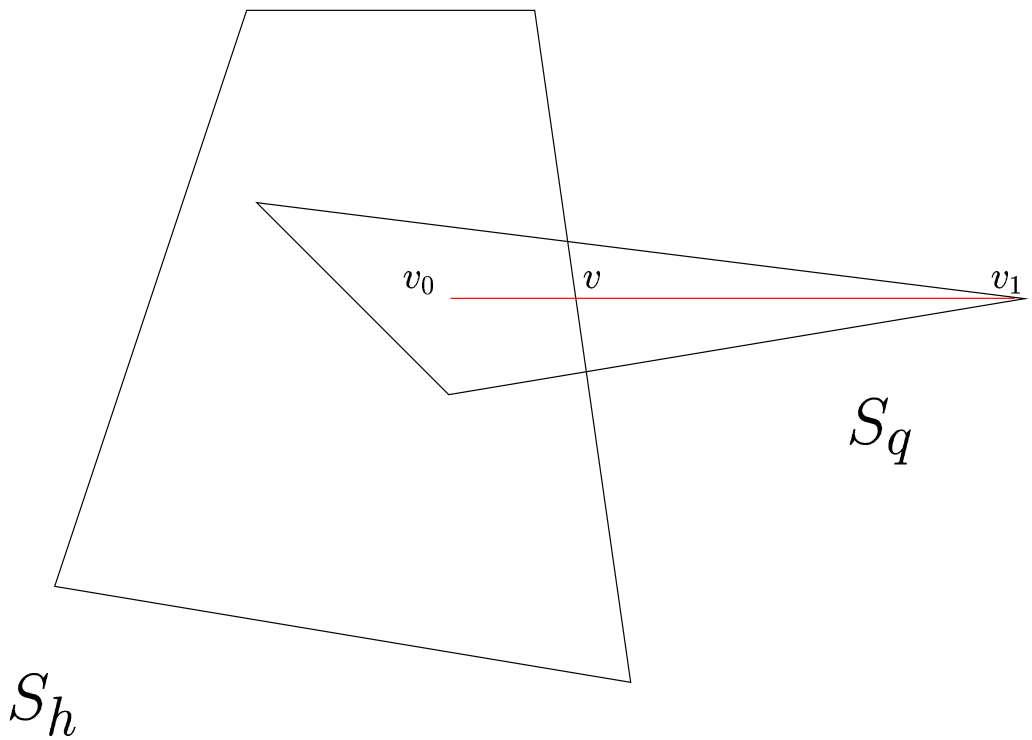

Figure 9: Test signal design

It follows that the test signal that we are looking for should be such that $v \in V_q$ and $v \notin S_h$. Our construction of the test signal v is equivalent to find v, a particular vertex in V_q, which does not belong to S_h.

To test (69) is easier than to test (68) because V_q is a finite set and we can verify element by element, if there exits some element v_1 such that $v_1 \in V_q$ and $v_1 \notin S_h$. We can use this vertex like a test signal.

Even though v_1 verifies the conditions that we are searching for our test signal, it is an extreme solution, therefore we will try to do better, choosing a point near the boundary of S_h. Let $v_0 \in S_q \cap S_h$, (without loss of generality we can assume that $v_0 = 0$) and v the intersection point of the segment v_0v_1 with the boundary of S_h then $v = \lambda v_1$, where λ is the solution of the following problem:

$$\max_{w,\lambda} \lambda \tag{70}$$

subject to

$$\begin{array}{ccccc} Fw & + & \lambda R v_1 & = & p_0 \\ Ew & & & \leq & q \end{array} \tag{71}$$

To find v, over interval $[0, N-1]$, as short as possible, we propose the following algorithm:

Algorithm 1

Step 1 : Let N = 1

Step 2 : Find V_q

Step 3 : If exits $v_1 \in V_q$ such that $v_1 \notin S_h$ go to step 4, else let N = N + 1 and go to step 1

Step 4 : Calculate λ according to (70), (71), $v = \lambda v_1$

Remark 1 *To calculate the set of vertices V_q, we exploit the property of the block diagonal matrix Q. We will find the polyhedron vertices defined for each block, then we do the corresponding permutations to find all the elements of V_q.*

Let $v_o \in V_q$, if $0 \in S_q$, then $v = v_o\lambda \in S_q, \quad \forall\lambda \in [0, 1]$

$$\max_{\delta,\lambda} \lambda \tag{72}$$

subject to

$$\begin{array}{ccccc} Lv_o\lambda & + & G\delta & = & h \\ & & \delta & \geq & 0 \\ 1 & \geq & \lambda & \geq & 0 \end{array} \tag{73}$$

Let (δ^o, λ^o) a solution to (72), (73), if $\lambda > \lambda^o$ and $\lambda \in [0, 1]$, then $v = v_o\lambda$ does not satisfy (63), i.e. $v \in V_q$ is such that $v \notin S_h$.

The algorithm 1 can be rewritten as follows :

Algorithm 2

Step 1 : Let N = 1

Step 2 : Find V_q

Step 3 : If $v_o \in V_q$, such that exist (λ^o, δ^0) solution of (72), (73), goto set 4, else let N = N + 1 and go to step 1

Step 4 : $v = v_1\lambda$, with $1 \leq \lambda > \lambda^o$

Remark 2 *The number of linear programming problem variables (72), (73), is lower than the number of variables in the problem (70), (71) : in (72), (73) the number of variables is equal to the number of inequality constraints in (71) plus one.*

There is some flexibility in the choice of the signal test and in some particular cases, it can be interesting to select specific v's.

Particular cases of the signal test

Case 1

If in (67) $Q(k) = Q_o$ and $q_v(k) = q_o$ for $k \in [0, N-1]$, we define the polyhedron

$$P_o = \{v \ / \ Q_o v \leq q_o\} \tag{74}$$

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

Among all the vertices of S_q we can restrict the selection of v in the following subsets of $V_q = \{v \ / \ v \text{ vertices of } S_q\}$

- V_1 is the set of v such that

$$v(k) = \begin{cases} w_i & \text{if} \quad k \in [0, k_o] \\ w_j & \text{if} \quad k \in [(k_o+1), (N-1)] \end{cases} \tag{75}$$

  with $k_o = 0, ..., N-1$ and $w_i, w_j \in V_o$.

- V_2 is the set of v such that

$$v(k) = \begin{cases} w_i & \text{if} \quad k \in [0, k_o] \\ w_j & \text{if} \quad k \in [(k_o+1), k_1] \\ w_l & \text{if} \quad k \in [(k_1+1), (N-1)] \end{cases} \tag{76}$$

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

Case 2

Another type of signal test of interest can defined by :

$$v(k+1) = v(k) + z(k) \qquad k \in [0, N-1] \tag{77}$$
$$n(k) \leq v(k) \leq m(k)$$

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

We can see this problem as a way of aggregating one state variable in our system. Equations (1), (4) are transformed as :

$$\left[\begin{array}{c} x_i(k+1) \\ v(k+1) \end{array}\right] = \left[\begin{array}{cc} A_i & b_i \\ 0 & 1 \end{array}\right] \left[\begin{array}{c} x_i(k) \\ v(k) \end{array}\right] + \left[\begin{array}{c} B_i \\ 0 \end{array}\right] u(k) + \left[\begin{array}{c} 0 \\ 1 \end{array}\right] z(k) + \left[\begin{array}{c} M_i \\ 0 \end{array}\right] \nu_i(k)$$

$$y(k) = \left[\begin{array}{cc} C_i & d_i \end{array}\right] \left[\begin{array}{c} x_i(k) \\ v(k) \end{array}\right] + D_i u(k) + N_i \nu_i(k)$$

with $i = s, f$.

Given z, a test signal constructed for the extended systems, the signal v for the original systems is completely defined by (77) once $v(0)$ (or any $v(k)$) is fixed.

Then to design the signal test, we can set initial conditions or linear programming problem (70), (71) with additional bounds on some components of v.

It is possible to consider many criteria for choosing a detection signal z, for instance that $z \in V_q$ minimizes the following function :

$$\|\sum_{k=0}^{N-1} z(k)\| \tag{78}$$

.

6 Examples

6.1 Example 3

The following results are taken from a random system having five state variables and two outputs. For choosing a detection signal v we will consider here the following criteria :

$$-300 \leq v \leq 300$$

The values of v_1 and λ are determined by Algorithm 2 and typical signal tests obtained by the method proposed in (75) which gives :
$v_1 = [-300, 300, 300, 300, 300]$ and $\lambda = 0.219$.
Then we choose for detection signal :

$$v = 0.3 * v_1 = [-90, 90, 90, 90, 90, 90]$$

Based on the algorithm described in the section 3, the failure detection test can be based on the hyperplane:

$$d = 0.0016y_1(0) - 0.0097y_2(0) + 0.059y_1(1) - 0.022y_2(1) - 0.080y_1(2)$$
$$+0.024y_2(2) + 0.041y_1(3) + 0.095y_2(3) - 0.35y_1(4) + 0.14y_1(5) - 0.026y_2(5)$$

The value of d is determined by solving the linear program (32) which gives $d = 10.34$ and we obtain here the following test:
For $11.047299 \leq d \leq 15.106376$: the system works normally
For $6.1376013 \leq d \leq 9.6385558$: the system works with anomalies
The following figures show typical signal tests obtained and dynamic responses of system.The curve A are for the operating system and curve B for the failed system.

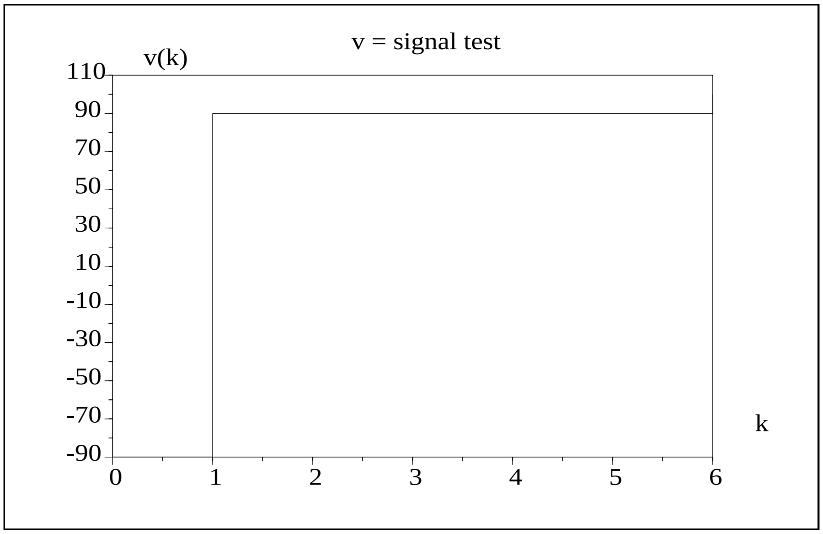

Figure 10: Signal test

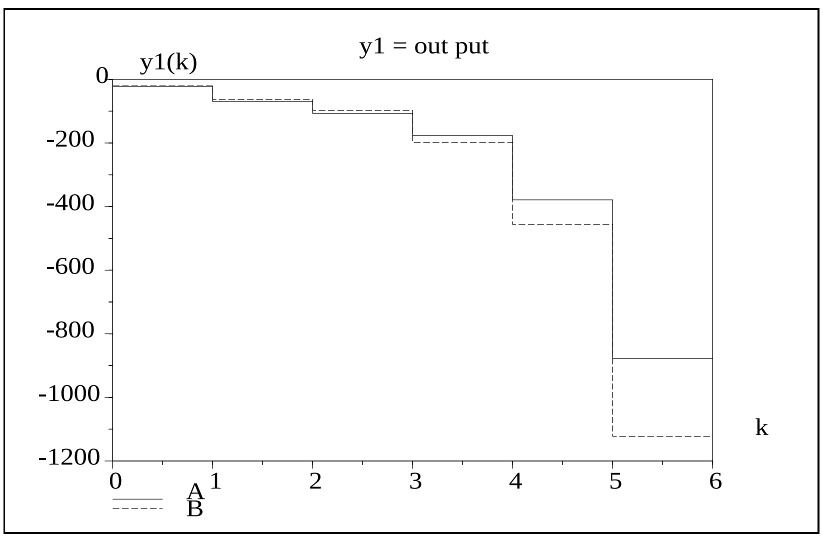

Figure 11: Output y_1

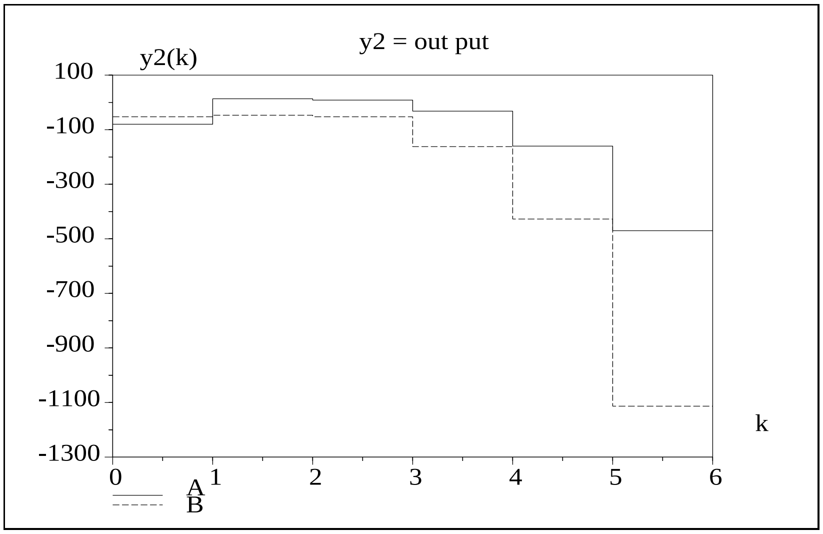

Figure 12: Output y_2

6.2 Example 4

This example show typical signal tests obtained by the method proposed in (77). The results are taken from a random system having four state variables and two outputs.

Test signal:

$$V = [0,\ 1.1,\ 2.2,\ 1.1,\ 2.2,\ 1.1,\ 0,\ 1.1,\ 0]$$

The hyperplane equation found by the algorithm is:

$$\begin{aligned} 22.83y_1(1) - 34.09y_1(2) + 27.20y_2(1) - 40.12y_2(2) - 0.348y_2(3) \\ +5.572y_1(5) - 6.512y_2(5) + 2.656y_1(6) - 2.818y_2(6) + 0.345y_1(7) \\ -0.325y_2(7) = -16.21 \end{aligned}$$

The figures display the signal test v and the output vector (y_1, y_2) as a function of time. The curve A are for the operating system and curve B for the failed system.

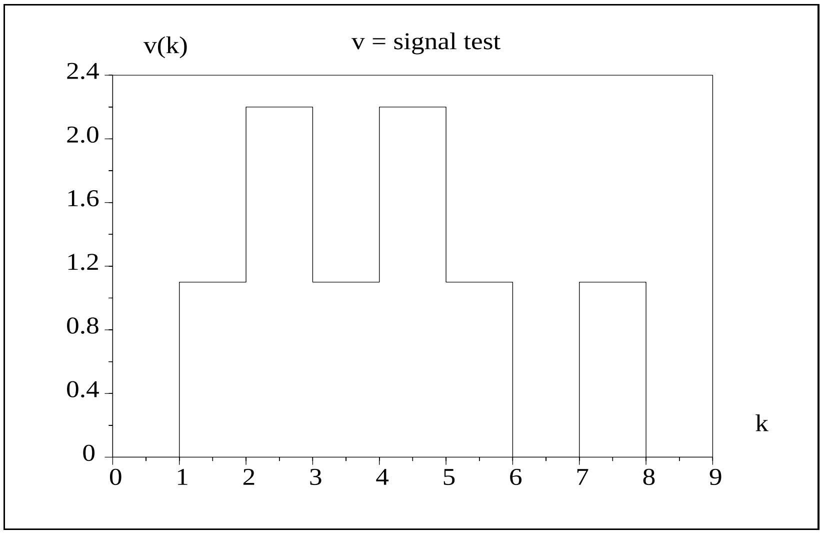

Figure 13: Signal test

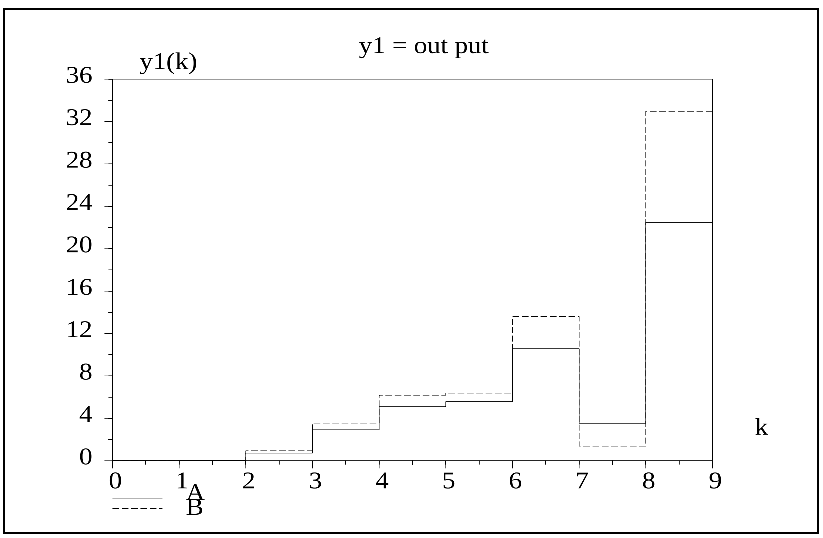

Figure 14: Output y_1

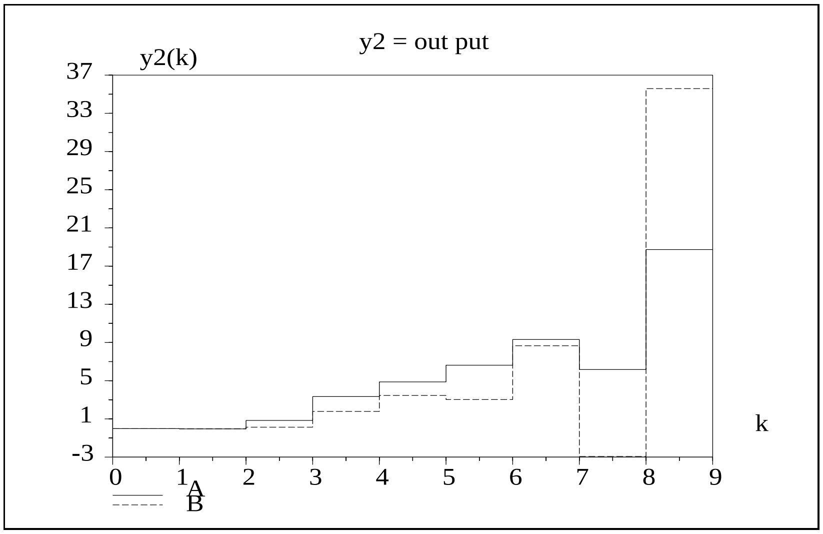

Figure 15: Output y_2

7 Appendix

The matrices A, B, C, D, M, N, R_ν and p_ν from (43) and (44) define the dynamics of both the plant in discrete time from.

$$A = \begin{bmatrix} 0.7951 & -0.008282 & -0.07689 & 0.5078 \\ 0.01785 & 0.9999 & -0.0007949 & 0.005345 \\ -0.01486 & 0.001104 & 0.9215 & 0.05294 \\ 0.01694 & 0.01073 & -0.01560 & 0.9389 \end{bmatrix}$$

$$B = \begin{bmatrix} 0.1786 & -0.1495 \\ 0.001854 & -0.001560 \\ -0.02738 & 0.07352 \\ 0.001782 & 0.01689 \end{bmatrix}$$

$$M = \begin{bmatrix} 0.0181 & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 0.000187 & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & 0.00566 & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0.00207 & 0. & 0. & 0. & 0. \end{bmatrix}$$

$$C = \begin{bmatrix} 2.8 & 0.149 & 1.01 & -7.03 \\ 0. & 1. & 0. & 0. \\ 0. & 0. & 1. & 0. \\ 0.307 & 0.588 & -2.49 & -1.117 \end{bmatrix}$$

$$D = \begin{bmatrix} -2.37 & 2.02 \\ 0 & 0. \\ 0 & 0. \\ -0.0442 & 3.727 \end{bmatrix}$$

$$N = \begin{bmatrix} 0. & 0. & 0. & 0. & 0. & 0.0001 & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 0.0001 & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0.0001 \\ 0. & 0. & 0. & 0. & 0.0001 & 0. & 0. & 0. \end{bmatrix}$$

$$R_\nu = \begin{bmatrix} 1. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\ -1. & 0. & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 0.8 & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & -0.8 & 0. & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & 0.9 & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & -0.9 & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0.7 & 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & -0.7 & 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. & 1. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. & -1. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & 0.8 & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & -0.8 & 0. & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 0.9 & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & -0.9 & 0. \\ 0. & 0. & 0. & 0. & 0. & 0. & 0. & 0.7 \\ 0. & 0. & 0. & 0. & 0. & 0. & 0. & -0.7 \end{bmatrix}$$

$$p_\nu = [15, 6, 15, 6, 15, 6, 15, 6, 15, 6, 15, 6, 15, 6, 15, 6]^T$$

The matrices J, F, G and H from (45) define the dynamics of autopilot in discrete-time form.

$$J = \begin{bmatrix} 0. & 0. & 0. & 0. & 0. & 0. & 1. \\ 0. & 0. & 0. & 0. & 0. & 1. & 0. \end{bmatrix}$$

$$H = \begin{bmatrix} 0. & 0. & 0. & 0. & 0.039 & 0.00183 & 0. \end{bmatrix}^T$$

$$F = \begin{bmatrix} 0.7837 & 0.008316 & 0. & 0. & 0. & 0. & 0. \\ -15.8 & 0.05189 & 0. & 0. & 0. & 0. & 0. \\ 0. & 0. & 0.549 & 0. & 0. & 0. & 0. \\ 0. & 0. & 0. & 0.549 & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. & 0.961 & 0. & 0. \\ 0. & 0. & -0.131 & 0. & 0. & 0.869 & 0. \\ -0.128 & 0. & 0. & 0.174 & 0.0232 & 0. & 0.942 \end{bmatrix}$$

$$G = \begin{bmatrix} 0. & 16.02 & 0. & 0. \\ 0. & -1786. & 0. & 0. \\ 0. & 0. & 0. & 0.451 \\ 0.451 & 0. & 0. & 0. \\ 0. & 0. & 0. & 0. \\ 0. & 0.162 & -0.131 & 0. \\ 0. & 0. & 0. & 0. \end{bmatrix}$$

8 Conclusion

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

The complexity of filter computation and design of the test signal can be important , all the operations needed for our method are implemented by solving large linear optimization problems.

The method presented here can sometimes be applied to continuous-time linear dynamical systems with discrete-time measurements.

This method does not have difficulties with very large systems because it works with sparse matrices and solving large linear optimization problem taking advantage of sparse matrix properties.

The efficiency of the proposed approach has been shown by two worked out examples.

References

[1] Basseville, M. and Benveniste, A., (eds) (1985) Detection of abrupt changes in signals and dynamical systems. Lecture notes in Control and Information Science, Vol 77, Sprinter Verlag.

[2] Basseville, M. and all (1987) Detection and Diagnosis of Changes in the Eigenstructure of Nonstationary Multivariable System, Automatica, Vol 23, No 4, pp. 479-489.

[3] Basseville, M. (1988) Detecting Changes in Signals and Systems - A Survey Automatica, Vol. 24 No 3, pp. 309-326.

[4] Clark, R. N. and V.M. Walton (1975) Detecting instrument malfunctions in control systems. IEEE Trans. Aerospace Electron. Syst., AES-11, pp. 465-473.

[5] Clark, R. N. (1978) Instrument fault detection, IEEE Trans. Aerospace Electron. Syst., AES-14, pp. 456-465.

[6] Clark, R. N. and W. Setzer (1980) Sensor fault detection in a system with random disturbances. IEEE Trans. Aerospace Electron. Syst., AES-16, pp. 468-473.

[7] Dakin, R. J. (1965) A Tree Search Algorithm for MILP Problems, Comput. J., 8 pp. 250-255.

[8] Gondzio, J. (1996) Multiple centrality corrections in a primal-dual method for lineal programming, Computational Optimization and Application, 6, pp. 137-156.

[9] Isermann, R. (1984) Process fault detection based on modeling and estimation methods. A survey. Automatica, Vol 20, pp. 387-404.

[10] Kerestecioğlu, F. (1993) Change detection and input design in dynamical systems. Research Studies Press, Taunton, U.K.

[11] Kerestecioğlu, F. and Zarrop(1994) M. B. Input design for detection of abrupt changes in dynamical systems. Int. J. Control 59, no. 4, pp. 1063–1084.

[12] Mehrotha, S. (1992) On the implementation of a primal-dual interior point method, SIAM J. Optimization 2., pp. 575-601.

[13] Milanese, M. and Vicino, A. (1991) Optimal estimation theory for dynamic systems with set membership uncertainty: An overview. Automatica, Vol 27, No 6, pp. 997-1009

[14] Mironovki, L. A. (1979) Functional diagnosis of linear dynamic systems. Automn Remote Control, 40, pp. 120-128

[15] Nikoukhah, R. (1998) Guaranteed Active Failure Detection and Isolation for Linear Dynamical Systems. Automatica, Vol 34, No 11, pp. 1345-1358

[16] Orchard-Hays, W. (1968) Advanced Linear Programming. Computing Techniques, McGraw-Hill

[17] Piet-Lahanier, H. and Walter, E. (1990) Exact recursive characterization of feasible parameter sets in the linear case. Math. and Computers in Simulation, 32, pp. 495-504.

[18] Rockafellar, R. T. (1972) Convex Analysis. Princeton, New Jersey. Princeton University Press.

[19] Rubin, D. (1975) Vertex generation and cardinality constrained linear programs. Operations Research, 23(3), PP. 555-565.

[20] Rubio Scola, H. (1997) Implementation of Lipsol in Scilab Rapport INRIA No 0215.

[21] Rubio Scola, H. (1999) Solving large linear optimization problems with Scilab: application to multicommodity problems. Rapport Technique No 0227, INRIA.

[22] Thie, P. R. (1988) An Introduction to Linear Programming and Game Theory, second edition, John Wiley and Sons.

[23] Uosaki, K., Tanaka, I. and Sugiyama, H. (1984) Optimal input design for autoregressive model discrimination with constrained output variance. IEEE Trans. Auto. Contr. , vol. AC-29, no 4, pp. 384-350.

[24] Watanabe, K. and M. Iwasaki (1983) A fast computational approach in optimal distributed-parameter state estimation. Trans. ASME J. Dynam. Syst. Meas. Control 105, 1-10.

[25] Wilde, E. (1993) A library for doing polyhedral operations, IRSA Report, no 785.

[26] Willsky, A. S. (1976) A survey of design methods for failure detection in dynamics systems. Automatica, Vol 12, No 4, pp. 601-611

[27] Zhang, X. J. (1989) Auxiliary signal design in fault detection and diagnosis. Spring-Verlag, Heidelberg.

[28] Zhang, Y. (1995) User's Guide to LIPSOL, Department of Mathematics and Statistics. University of Maryland Baltimore County. USA.

Unit´e de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifi que,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unit´e de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unit´e de recherche INRIA Rhône-Alpes, 655, avenue de l'Europe, 38330 MONTBONNOT ST MARTIN
Unit´e de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unit´e de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
http://www.inria.fr
ISSN 0249-6399