Abstract

CHOE, DONGKYOUNG. Digital Signal Design for Fault Detection in Linear Continuous Dynamical Systems. (Under the direction of Dr. Stephen L. Campbell.)

A systematic approach to detect underlying undesirable states of a physical system is to compare its observed behaviors to several competing models and to identify the model that best describes the observation. This model selection process can be enhanced by applying a specially designed auxiliary input signal to the system.

This dissertation applies the auxiliary-signal-based model selection approach for recognizing faulty behaviors of systems whose dynamic behaviors can be described by linear differential equations. Using an existing analog–signal–based algorithm, the effect of the modeling error on this particular type of detection approach is examined and a geometrical explanation is provided. We also present a variation of the analog–signal–based algorithm which produces signals that are more practical for certain types of applications. In addition, an alternative auxiliary signal design algorithm is developed, producing digital signals that minimally disturb regular system operation and guarantee fault detection for a given amount of the gap between physical system and the corresponding model. The algorithm implements the analytical solution steps derived mostly by the optimal control problem solution technique while converting a nested optimization problem into an equivalent eigenvalue problem. The algorithm provides an option to optimize the duration of each digital piece to yield even more

"plant-friendly" auxiliary detection signals at the cost of a moderate increase in computational time.

DIGITAL SIGNAL DESIGN FOR FAULT DETECTION IN LINEAR CONTINUOUS DYNAMICAL SYSTEMS

BY

Dongkyoung Choe

A DISSERTATION SUBMITTED TO THE GRADUATE FACULTY OF

NORTH CAROLINA STATE UNIVERSITY

IN PARTIAL FULFILLMENT OF THE

REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

OPERATIONS RESEARCH

RALEIGH, NORTH CAROLINA

April 2007

APPROVED BY:

Dr. Stephen L. Campbell	Dr. Robert Buche
CHAIR OF ADVISORY COMMITTEE	
Dr. Yahya Fathi	Dr. Negash Medhin

For my beloved parents, Insoo and Mija

Biography

Dongkyoung Choe grew up in a beautiful port city, Pusan, South Korea until she graduated Haksan Girls' High School in February 1998.

She continued her study at Hankuk Aviation University, South Korea majoring in Air Transportation. In her junior year, she was selected to participate in the Exchange Student Program and attended Airline Business and Computer Science courses at Embry-Riddle Aeronautical University, Florida for a year. In 2002, Dongkyoung Choe earned her B.S. from Hankuk Aviation University with the title of the Best Student of the graduating class of 2002.

She arrived at North Carolina State University to pursue a Ph.D. in Operations Research. During her graduate years, she interned at INRIA-Rocquencourt with Ramine Nikoukhah in Le Chesnay Cedex, France. In addition, She received the Salah E. Elmagraby Award and the Winton–Rose Scholarship for outstanding achievement in and contributions to the field of Operations Research and Mathematics. After graduating with a Ph.D. in Operations Research, she plans to secure a solution design and decision support tool development position.

Acknowledgements

I owe a great amount of gratitude to my advisor and mentor, Dr. Stephen Campbell. Without his guidance, expertise, and encouragement, I would not have been able to complete this research project.

I am also grateful to Dr. Robert Buche, Dr. Yahya Fathi, Dr. Negash Medhin and Dr. Hien Tran for serving on my committee and for guiding me in and outside of class throughout my graduate study.

I would like to thank Dr. Ramine Nikoukhah and INRIA-Rocquencourt for giving me an opportunity to work with them at their respected institutes. It was a true learning experience in so many aspects.

A big thank to you, my dearest sisters and friends. From the opposite side of the Earth or right next to me, you uplift my spirit for everything I do.

Finally, thanks mom and dad for your love and support. This and everything else are worthwhile doing because of you.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

This work is concerned with the problem of fault detection. Fault detection is an important part of the operations of many different systems such as mechanical devices, electrical circuits, and chemical reaction processes ranging from industrial-level processes to household appliances. Nowadays systems are more than ever complex and automated due to the technologies that have grown sophisticated and the increasing number of replacement of human operators by computers and microprocessors.

In our society where humans are directly or indirectly influenced by the elaborate and highly-automated systems, the smooth operation of systems is tied to their safety, economic growth, environment-friendly life and so on. Naturally, the condition of the systems has been one of the main concerns.

A fault is an undesirable state of the system which may permanently or temporarily interrupt the systems while they perform acceptable, required, expected functions [23]. Therefore, detection of faults is an absolute must in any practical system [18]. The primary goal of fault detection is early recognition of a non-usual, problem-prone behavior of observed system in order to prevent the system from shutting down and causing catastrophic disasters. Fault detection increases system availability, reliability, and safety.

In the rest of this chapter, the goals and contents of this dissertation are further described and defined. In section 1.1, an application example is provided to motivate the study of the fault detection problem. Section 1.2 summarizes key fault detection methods, and then the mathematical background related to this dissertation is reviewed in section 1.3. A recently developed active fault detection algorithm, which served as a starting point of this research, is described in section 1.4.

1.1 Motivating Application Example of Fault Detection

Because of the essential role of fault detection in various system operations, it has been the subject of many studies. From navigation systems of planes, boats, rockets and other moving objects [5] to biomedical systems such as a drug infusion system [1], we see various applications of fault detection. In this section, we draw a real-life example from [4]. Its inclusion here will hopefully motivate the topic of this dissertation.

Spacecraft Monitoring [4]

The system, called the Spacecraft Health Automated Reasoning Prototype (SHARP), is designed at Jet Propulsion Laboratory (JPL) in California to automate monitoring and diagnosing the status of spacecraft and the telecommunication between the craft and ground systems. SHARP was deployed into the Voyager spacecraft and was demonstrated during the Voyager 2 encounter with the planet Neptune in August 1989. During the encounter, it accurately detected, analyzed, and tracked the antenna drift and pointing conditions in a matter of seconds which would have taken human operators minutes or hours. It also detected a data error which appeared in the telemetry and efficiently identified the cause of the problem.

For fault detection and diagnosis, SHARP is designed to use Artificial Intelligence techniques including procedural reasoning, blackboards, and spontaneous computation daemons. SHARP begins to check spacecraft condition by determining whether received engineering data values are within acceptable limits. Alarm limits are predetermined by the domain experts for each spacecraft state. After identifying the spacecraft's current state, SHARP retrieves the table of alarm limits corresponding to the state and compares it with newly received data. If abnormal performance is detected and it needs to be further analyzed, SHARP sends the data and analysis results to a fault classification module where the data and analysis reports are interpreted to identify the pattern of the problem and locate the source of the abnormality. In the cases where no clear interpretation is available, SHARP tries to diagnose the fault by conducting several hypotheses tests, if not one, in parallel. At the end of the tests, SHARP presents a single or multiple recovery recommendation depending on the degree of conflict between accepted hypothesis.

SHARP's real-time fault detection and diagnosis can prevent the loss of the telecommunications link due to the spacecraft's antenna drift. It was one of the stated causes of the loss of the Phobos spacecraft. The fault detection and diagnosis function of SHARP supports safe and reliable operation of the spacecraft, and hence contributes to the success of the spacecraft mission.

1.2 Fault Detection Methods

The first, naturally developed and still widely used, method of fault detection is to use human senses [18]. They can distinguish change in color, irritating fumes, unusual sound and inconsistent texture. Over time, systems has grown complicated, and accordingly more systematic techniques are developed to assist or replace biological senses in discriminating the unhealthy states of such systems. In this section, we

review some of these advanced fault detection methods.

1.2.1 Passive vs. Active approach

In general, the approaches to the problem of fault detection can be grouped into two categories, passive and active. The main difference of the two approaches is whether or not one involves generating and applying an auxiliary input for the specific purpose of fault detection.

The passive approach, which does not use such an auxiliary signal, measures and/or estimates system states and looks for its deviation from known fault-free condition.

On the other hand, the active approach acts upon the system using an auxiliary signal to improve detection performance. The auxiliary signals are designed in advance based on known system behaviors and specifications, and then are applied to the system periodically or at critical times to reveal abnormal behavior if it exists.

Passive detection is useful, in particular, for the systems that are restrictive to auxiliary signals for material, safety, and sensitivity reasons. A drawback with this approach is that it can miss an existing fault that stays hidden during regular operation. For example, a truck driver would not notice a malfunction in the brake system of his truck while he drives on level ground within smooth traffic flow [39]. However, a little system excitement which can be done in the active approach can make such hidden faults surface before the system fails. In the brake system example, tapping the breaks is equivalent to the application of an auxiliary signal. In this research, the active fault detection approach is adopted. What follows is the review of some of the basic techniques that are used for a fault detection test and signal design.

1.2.2 Basic fault detection test techniques

There are essentially two groups of techniques that are used for fault detection: Model-free and model-based. When the true condition is well-indicated by system behavior measurements, model-free techniques may be used. On the other hand, some inner states may be hard to gauge, but it may be important to know their dynamics to maintain smooth operation. A mathematical model provides a way to describe the internal states by relating measurable input and outputs. In this dissertation, a model-based detection technique is utilized. As the model-free techniques are often combined with a model-based approach, we first briefly review them below.

Model-free approach [18]

- *Comparison between sensor measurements.* At least three identical instruments (sensors) are installed to measure the same physical activity. Any discrepancy between measurements indicates a fault. This approach is often referred as physical (hardware) redundancy test.
- *Comparison between measurements and preset quantities.* System measurements are compared with a predefined threshold (limit), time trend and/or frequency spectrum that indicate healthy condition of the system.
- *Logic reasoning.* The simplest logic reasoning method is to evaluate system measurements by following a tree of logical rules of the "IF-symptoms-THEN-conclusion" type. After successive evaluations, a final analysis of the measurement is obtained.

[1] applied a threshold test and a logic reasoning method to detect faults that may occur while infusing a drug into a patient.

Model-based approach

The physical systems of interest of this dissertation are the systems which are dynamic in nature. We are particularly interested in their continuous-time operation near an equilibrium condition. Such systems can be modeled by linear differential equations. In what follows, some of the basic techniques that can be used with a linear differential equation model are reviewed.

- *Comparison between measurements and analytically computed values.* Relating input and output measurements that are collected during the non-faulty condition provides a model that estimates internal or non-measurable system states. The difference between the model estimates and measurements is never zero, even if there is no fault, as there are always present model errors, disturbance and noises. The amount of the discrepancy, called the residual, is analyzed by a model-free approach to test existence of a fault. This type of fault detection method is often divided into three subcategories, observers, filers, and parity equations based, depending on how the system states are estimated. Interested readers are referred to [23] for more on these methods. Some of recent articles [3, 38, 2] are relevant to this type of fault detection.
- *Comparison between analytically computed values.* A system model is identified by obtaining parameter values in the faulty-free condition and they become reference parameters. Then the parameters are repeatedly re-identified during regular operation. The parameter value's deviation from reference is tested by a model-free method to determine system states. [21] and [37] applied parameter estimation based fault detection to an electronic analog circuits and a brushless DC motor, respectively.
- *Comparision between models and measurements.* Nominal and faulty

behavior patterns are described by mathematical models. System measurements are compared to the models to determine the model best representing measurements.

When modeling system states, there are two ways of treating initial state uncertainty, system dynamic disturbances and measurement noises. When they are considered to be a random variable and stochastic processes, each model predicts system behaviors to follow a certain probability distribution function, and hence, the model selection can be done by comparing the likelihood of competing models, that is, the probability that measurements are predicted by a model [49, 29, 8].

An alternative approach is to consider the uncertainty, disturbance and noises as unknown but deterministic variables that belong to a given vector set [6].Under this circumstance, all information of system states in a particular mode (i.e. normal or faulty) is described by a family of sets. For fault detection, it is then required to identify the membership of system measurements [39, 22]. In this dissertation, we follow this approach.

1.2.3 Signal design for fault detection

A good choice of detection signal is an essential part of the active approach. Although the meaning of "good" can vary depending on the type of systems that are being investigated, the kind of detection techniques and other design considerations, on the whole designing a detection signal amounts to solving a constrained optimization problem. For example, [8] pursued auxiliary input signals that minimize the probability of selecting an incorrect model. Also, input signals that minimize the economical, environmental loss by making wrong decision were sought in [47]. [39] and [13] tried to find input signals that are as short as possible and that minimally

disturb system operation, respectively, while fault detection is guaranteed for a given level of uncertainty. The signals with minimal disruption characteristics are referred as "plant-friendly" signals i.e. signals that can be introduced while the plant is in its regular operation. This dissertation also discusses this type of signals.

1.3 Mathematical Background

In this section, we briefly review several of mathematical theories and techniques that play an essential role in solving and analyzing the main research problem of this dissertation.

1.3.1 Systems and Optimal control

The fault detection signal design problem assumes the ability to regulate the dynamic behavior of a given physical system. To adjust the behavior of the system in some desired way, an input is applied to the system and such input is called a control. Among controls, an optimal control is a control law that makes a given system achieve a certain optimal performance. Designing a fault detection signal involves finding such a control, and it is computed by solving a continuous optimal control problem.

In a continuous optimal control problem, the dynamics of a given system is described by first-order ordinary differential equations

$$\dot{x}(t) = f(x, \nu, t) \tag{1.1}$$

with system state history $x(t) \in R^n$ and control input $\nu(t) \in R^m$. Along with the system, system performance is often measured as follows:

$$J = \phi(x(t_b), t_b) + \phi_f(x(t_f), t_f) + \int_{t_b}^{t_f} L(x, \nu, t)\, dt \tag{1.2}$$

where $[t_b, t_f]$ is the time interval of interest. The functions ϕ, ϕ_f, and L evaluate system operation at initial, final, and intermediate times, respectively. An optimal control is the input $\nu^*(t)$ on the time interval $[t_b, t_f]$ that drives the system (1.1) to a state x^* such that the measure of undesirable performance (1.2) is minimized.

In a subproblem of FD signal design problem, the initial and final times t_b, t_f are fixed and the evaluation of the system at the final time is ignored, i.e. $\phi_f \equiv 0$. Then, the continuous optimal control problem of interest can be written as follows:

$$\min_{\nu} \quad \phi(x(t_b)) + \int_{t_b}^{t_f} L(x, \nu, t)\, dt \tag{1.3a}$$

$$\text{subject to} \quad \dot{x} = f(x, \nu, t) \tag{1.3b}$$

To solve (1.3), we shall use the variational approach. First, to include the constraint, we form an augmented functional

$$\bar{J} = \phi(x(t_b)) + \int_{t_b}^{t_f} L + \gamma^T (f - \dot{x})\, dt. \tag{1.4}$$

using a Lagrange multiplier γ. Note that since the constraint (1.3b) holds at each $t \in [t_b, t_f]$, the associated multiplier $\gamma \in R^n$ is a function of t.

The optimal control law $\nu^*(t)$ for the constrained optimization problem in (1.3), if it exists, must be such that $\bar{J}$ is insensitive to small perturbations in ν^* and the ν^*- corresponding optimal state x^* and optimal co-state γ^*. That is, the differential in $\bar{J}$, $d\bar{J}$, is equal to zero for all the independent increments $\delta\nu$, δx and $\delta\gamma$.

Let $H(x, \nu, t) = L(x, \nu, t) + \gamma^T f(x, \nu, t)$ and rewrite $\bar{J}$ in (1.4) as

$$\bar{J} = \phi(x(t_b)) + \int_{t_b}^{t_f} H(x, \nu, t) - \gamma^T \dot{x}\, dt. \tag{1.5}$$

Then, Leibniz's rule[1] provides

$$\begin{aligned} d\bar{J} &= \phi_x dx\Big|_{t_b} + (H - \gamma^T \dot{x})\, dt\Big|_{t_f} - (H - \gamma^T \dot{x})\, dt\Big|_{t_b} \\ &\quad + \int_{t_b}^{t_f} H_x \delta x + H_\nu \delta\nu - \gamma^T \delta\dot{x} + (H_\gamma - \dot{x})^T \delta\gamma \, dt. \end{aligned} \tag{1.6}$$

Note that if $x(t)$ is a continuous function of t, then the differentials $dx(t)$ and dt are not independent. If we let $\delta x(t)$ be the variation in $x(t)$, the incremental change in $x(t)$ while t being held fixed, we have, at a fixed point $t = \bar{t}$,

$$dx(\bar{t}) = \delta x(\bar{t}) + \dot{x}(\bar{t}) d\bar{t} \tag{1.7}$$

for small $d\bar{t}$. However, as our application has fixed initial and final time,

$$dt_b = 0, \qquad dx(t_b) = \delta x(t_b), \tag{1.8a}$$

$$dt_f = 0, \qquad dx(t_f) = \delta x(t_f). \tag{1.8b}$$

Also,

$$-\int_{t_b}^{t_f} \gamma^T \delta\dot{x}\, dt = -\gamma^T \delta x\Big|_{t_f} + \gamma^T \delta x\Big|_{t_b} + \int_{t_b}^{t_f} \dot{\gamma}^T \delta x \, dt. \tag{1.9}$$

Using (1.8) and (1.9), the equation (1.6) is simplified to

$$d\bar{J} = (\phi_x + \gamma^T) dx\Big|_{t_b} + \int_{t_b}^{t_f} (H_x + \dot{\gamma}^T)\delta x + H_\nu \delta\nu + (H_\gamma - \dot{x})^T \delta\gamma \, dt. \tag{1.10}$$

Then, by setting the coefficient of independent increments to zero, the conditions that

[1] *Leibniz's rule*: If $x(t) \in R^n$ is a function of t and $J(x) = \int_{t_b}^{t_f} h(x(t), t)\, dt$, where $J(\cdot)$ and $h(\cdot)$ are both real scalar functionals (i.e. functions of the function $x(t)$), then $dJ = h(x(t_f), t_f)\, dt_f - h(x(t_b), t_b)\, dt_b + \int_{t_b}^{t_f} [h_x^T(x(t), t)\delta x]\, dt$

the optimal solution of (1.3) must satisfy are obtained as follows:

$$\begin{aligned} \dot{x} &= H_\gamma = f && \text{(1.11a)} \\ 0 &= H_\nu = L_\nu + \gamma^T f_\nu && \text{(1.11b)} \\ -\dot{\gamma} &= H_x^T = L_x^T + f_x^T \gamma && \text{(1.11c)} \\ \gamma(t_b) &= -\phi_x^T(t_b) && \text{(1.11d)} \end{aligned}$$

Further information on optimal control can be found in [30] and [32].

1.3.2 Matrix eigenvalue problems

Later in this dissertation, we try to detect faults via signals that can be specified by only a sequence of quantities and time intervals for each. The signal is constructed by holding constant each quantity for the corresponding time interval, i.e. is a piecewise-constant signal or a digital signal. As we turn our attention from analog to digital signals, finite dimensional spaces become our main concern and the fault detection digital signal design problem boils down to a matrix eigenvalue problem. In this section we will briefly review several aspects of eigenvalue problem.

The eigenvalue and generalized eigenvalue problems

The matrix eigenvalue problem for a given square matrix A is to find a value λ and nonzero vectors x of A which satisfy

$$Ax = \lambda x. \tag{1.12}$$

The λ's and nonzero solutions x's are called eigenvalues and eigenvectors, respectively. We can describe them geometrically by saying that under the transformation of A,

an eigenvector keeps its direction parallel to itself and only experiences magnitude change by the quantity of the corresponding eigenvalue.

When A is a $n \times n$ real-symmetric matrix, its eigenvalues are real and, hence, they can be ordered as $\lambda_1 \geq \cdots \geq \lambda_n$. Then, the largest eigenvalue of A can be described as

$$\lambda_1 = \max_{\|z\|=1} z^T A z = \max_{z \neq 0} \frac{z^T A z}{z^T z} \tag{1.13}$$

where $\|z\|$ is the Euclidean vector norm, i.e. $\|z\| = \sqrt{z^T z}$. This can be proved by using the fact that A is orthogonally similar to a real-diagonal matrix which diagonal entries are its eigenvalues, i.e. $A = P^T D P$ for some orthogonal matrix P and that $\|z\| = \|P^T z\|$. The maximum is attained when z is the eigenvector of unit norm associate with λ_1 [33].

It is easy to see that, for a positive definite matrix Q, the value λ^* defined by

$$\max_{z \neq 0} \frac{z^T A z}{z^T Q z} \tag{1.14}$$

is the largest eigenvalue of $Q^{-\frac{1}{2}} A Q^{-\frac{1}{2}}$. In most situations of numerical computations, performing the inversion is not desirable. An alternative definition of λ^* is that it is the largest value of λ for which

$$A - \lambda Q \tag{1.15}$$

is singular. Such λ's are known as generalized eigenvalues or pencil eigenvalues and are the solutions of the generalized eigenvalue problem

$$Ax = \lambda Q x. \tag{1.16}$$

The z that attains the max in (1.14) is the nonzero solution x of (1.16) with $\lambda = \lambda^*$. The QZ algorithm introduced by Moler and Stewart [36] is one of the most well known

numerical algorithms that solve the generalized eigenvalue problem.

Perturbation theory for eigenvalue problems

When a digital signal is used for fault detection, the overall performance of the auxiliary signal is affected not only by the magnitude of each constant piece but also by the length of each piece. It turns out that seeking the best lengths of constant pieces that attain the "best" digital detection signals amounts to solving the matrix eigenvalue problem where the matrix of interest is now a function of several parameters.

Kato investigated how the eigenvalues and eigenvectors of a linear operator T change when T is subjected to a small perturbation [27]. We briefly review some facts regarding eigenvalues, in particular, when T is a real symmetric matrix operator and the perturbation is real. Rather than introducing multiple parameters, Kato considers the change of the eigenvalues of an operator T as a function of T, i.e. a function of the N^2 elements of T with respect to a fixed basis of a unitary space H. As the eigenvalues of T are all real, the repeated eigenvalues can be ordered as follows

$$\lambda_1(T) \leq \ldots \leq \lambda_N(T). \tag{1.17}$$

This defines N real-valued functions of T, T varying over all symmetric operators in H. All N functions are continuous functions of T in the sense that $\lambda_i(T+S) - \lambda_i(T)$ tends to zero as $\|S\| \to 0$. However, the $\lambda_i(T)$'s are not necessarily even partially differentiable, although they are piecewise smooth, since there are a finite number of crossing points in any bounded domain. It is known that the unordered N repeated eigenvalues are partially continuously differentiable with respect to T.

Exception occurs in the neighborhood of a T which has N distinct eigenvalues. Such operator T is called simple. Suppose $T = T_0$ is not only diagonal but also simple. Then, $T = T_0 + S$ also has N distinct eigenvalues for sufficiently small $\|S\|$ in

virtue of the continuity of eigenvalues with respect to entries of T. The eigenvalues of T can be expressed in a neighborhood of T_0 by N real-valued continuous functions $\lambda_i(T)$, $i = 1, \ldots, N$ without any coalescing points. Hence, the functions λ_i's are not only totally differentiable but smooth in a neighborhood of $T = T_0$.

For more information on perturbation theory for eigenvalue problems with general linear operators, see [27] and the references within.

1.4 An Active Fault Detection Approach

In this section, an active fault detection approach is summarized from [13]. This fault detection method is composed of two parts: designing an optimal analog detection signal and a detection test using the signal.

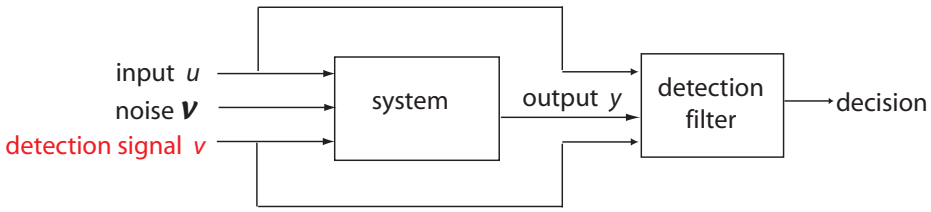

Figure 1.1: Active Fault Detection Approach

As an extension of this fault detection approach, this dissertation will answer several of practical questions for using the analog signal design algorithm and develop an algorithm that designs a digital counterpart of the analog signal. The digital signals will then use the detection test that is to be described later in this section.

We first describe the signal design algorithm.

1.4.1 The optimal analog signal design algorithm

The goal of this algorithm is to design an auxiliary signal that is highly effective in detecting abnormal behaviors without many false alarms and that minimally disrupt regular operations of the system of interest. According to this goal, the problem of designing a fault detection signal can be roughly defined as Table 1.1.

Table 1.1: Rough problem definition for designing optimal analog signal

Find auxiliary signal v that · minimally disturbs regular system operation · is analytically and numerically computable · and "guarantees" fault detection for system of interest

In the following, it is explained how the worded problem definition can be characterized mathematically, and how the problem can be solved analytically and numerically. Whenever it is helpful, we make use of the example of detecting faults in an aircraft's elevator actuator.

Problem Definition

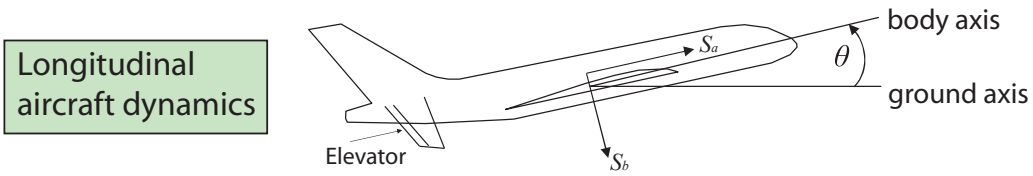

Figure 1.2: Longitudinal Aircraft Dynamics

Suppose the test period is $[0,\ T]$, which is assumed to be relatively short. To evaluate the performance of the aircraft's elevator actuator, the longitudinal dynamics are studied. The state of the dynamics can be composed of 4 elements as follows:

$$\text{state} \qquad x = \begin{pmatrix} S_a \\ S_b \\ \omega \\ \theta \end{pmatrix} = \begin{pmatrix} \text{velocity along body axis} \\ \text{velocity perpendicular to body axis} \\ \text{angular velocity } (= \dot{\theta}, \text{ pitch rate}) \\ \text{angle between body axis and ground axis} \end{pmatrix}.$$

Among the four, angular velocity $\dot{\theta}$ is supposed to be a good indicator of elevator actuator performance and, hence, its state is outputted for observation. Let y denote the output.

A system can be affected by, in general, three types of inputs during a fault detection period: regular inputs, detection signal inputs, and uncertainty inputs. In the case of aircraft longitudinal dynamics, regular input u may be thought of as a stick input commanded by a pilot for the purpose of maneuvering the plane. The detection signal inputs v are the ones that are inserted into the system by the pilot or the detection mechanism specifically for identifying the condition the system is in. It is also important to take account of the noise or disturbance added to the two inputs u, v and the output y. The wind that effects the aircraft's pitch rate can be considered such a disturbance along with input/output measurement noise. In the process of modeling a system, the possible error in the model itself can explain the gap between the physical system and the mathematical model of the system. All these noise, error and disturbances are considered as uncertainty inputs ν. More specifically, the latter is denoted as model uncertainty and the former as additive uncertainty.

Assuming no regular inputs during a detection test period, we suppose that the *normal* ($i = 0$) and *faulty* ($i = 1$) longitudinal dynamics of the aircraft linearlized at an equilibrium flight condition can be modeled as follows:

$$\dot{x}_i = A_i x_i + B_i v + M_i \nu_i, \tag{1.18a}$$

$$E_i y = C_i x_i + D_i v + N_i \nu_i \tag{1.18b}$$

Here the variables are deterministic and all the system matrices are determined in accordance with the aircraft's dynamic behavior; the only conditions are that N_i's have full row rank and E_i's have full column rank. Throughout this dissertation, it is assumed that there is only one possible type of fault. Although the basic idea of the detection signal design and detection tests of two-model approach naturally extends to the three or more models, the solution require different problem formulation. See [13] for further information on the fault identification.

Note that if we let the linear operator $\mathcal{L}(h)$ be the solution of $\dot{z} = Az + h$, $z(0) = 0$, and let $E_i = I$, the set of outputs for a given input signal v

$$\mathcal{Y}_i(v) = \mathcal{Y}_i(0) + [C_i \mathcal{L}_i B_i + D_i]v \tag{1.19}$$

where $\mathcal{Y}_i(0) = \left\{ [C_i \mathcal{L}_i M_i + N_i]\nu_i + C_i e^{A_i t} x_i(0) \right\}$. This shows that because of unrestricted initial condition of the state $x_i(0)$ and uncertainty variable ν_i, the normal and faulty models form an unbounded set of normal and faulty behaviors, respectively. Thus, it is reasonable to assume a bound on the allowable $x_i(0)$ and ν_i. We define the constraint on the amount of uncertainty in each model as follows:

$$\mathcal{S}_v^i(x_i(0), \nu_i, s) = x_i(0)^T P_{i,0}^{-1} x_i(0) + \int_0^s \nu_i^T J_i \nu_i \, dt < 1, \; \forall s \in [0, T], \tag{1.20}$$

where $P_{i,0}$'s are symmetric positive definite matrices and J_i's are diagonal matrices with 1 or -1 on their diagonal. When there is a greater uncertainty in the initial condition, a "larger" $P_{i,0}$ is to be selected. The diagonal matrix J_i is determined depending on the existence of model uncertainty, which is described in Chapter 2.1.1. When there is no model uncertainty, we have $J_i = I$, and thus we can just take $s = T$ in (1.20). The bound 1 is not restrictive as any positive real number can be scaled to one with some modification in the matrices M_i, N_i.

With (1.18),(1.20), it is implied that all possible normal and faulty angular velocities (output) can be described as the elements of two bounded sets.

Notice from (1.19) that the detection signal v moves the sets of output $\mathcal{Y}_i(0)$ in two different directions. Suppose that as a result of applying an elevator angle v, the normal and faulty output sets became disjoint as in Figure 1.3. Then, there is no ambiguity in determining from which model an observed pitch rate, which is obtained after injecting the aforementioned v, came from. In other words, if

$$\mathcal{Y}_0(v) \cap \mathcal{Y}_1(v) = \emptyset, \tag{1.21}$$

then v guarantees fault detection. Such a v is called a *proper signal*.

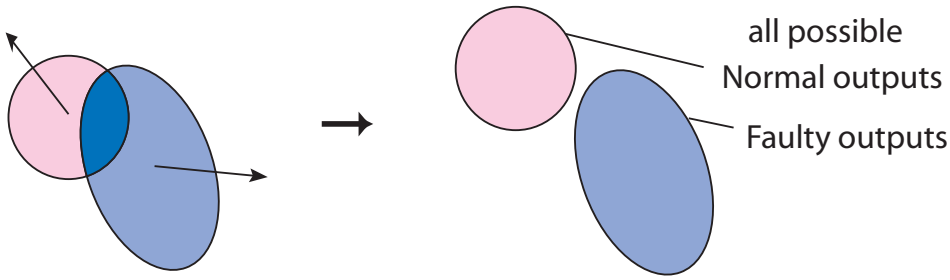

Figure 1.3: Signal v disjoints two output sets

The geometrical definition of a proper signal in (1.21) leads to a mathematical

problem definition that can be analytically and numerically solved. Suppose that a proper elevator angle v is inserted into an aircraft solely for fault detection. Then, the state of the aircraft at the moment of signal insertion $x(0)$ changes itself during the test period by the v and some other uncertain factors ν. And finally at the end of the test, a pitch rate y is yielded. This behavior change of the aircraft must be described by only one of the two models. Thus, the $x(0)$ and ν must satisfy only one of the two uncertainty constraint (1.20), which in turn implies that

$$\max\left\{ \mathcal{S}_v^0(x_0(0), \nu_0, s), \mathcal{S}_v^1(x_1(0), \nu_1, s) \right\} \geq 1, \forall s \in [0, T]. \tag{1.22}$$

Note that (1.22) holds for all the possible $(x_0(0), \nu_0, x_1(0), \nu_1)$ that satisfy the models (1.18a). Thus, proper signals, i.e. the signals guaranteeing fault detection, satisfy the following constraint:

$$\begin{aligned}
\inf \ \max\left\{ \mathcal{S}_v^0(x_0(0), \nu_0, s), \mathcal{S}_v^1(x_1(0), \nu_1, s) \right\} \ &\geq \ 1, \forall s \in [0, T] && \text{(1.23a)} \\
\text{for all } x_0, x_1, \nu_0, \nu_1 \text{ that satisfy} & && \text{(1.23b)} \\
\dot{x}_0 = A_0 x_0 + B_0 v + M_0 \nu_0, & && \text{(1.23c)} \\
E_0 y = C_0 x_0 + D_0 v + N_0 \nu_0, & && \text{(1.23d)} \\
\dot{x}_1 = A_1 x_1 + B_1 v + M_1 \nu_1, & && \text{(1.23e)} \\
E_1 y = C_1 x_1 + D_1 v + N_1 \nu_1. & && \text{(1.23f)}
\end{aligned}$$

As an auxiliary signal, the fault detection signal should be able to be introduced at any moment while the plant is in its usual operation. Unreasonably "large" signals are often proper, but cannot be applied in practice as they often interrupt too much

of the regular operation of systems. We consider cost functions on v of the form:

$$q(v) = \xi(T)^T W \xi(T) + \int_0^T \|v\|^2 + \xi^T U \xi \, dt, \tag{1.24a}$$

$$\dot{\xi} = F\xi + Gv, \quad \xi(0) = 0. \tag{1.24b}$$

where W, U are positive semi-definite matrices and F, G are chosen by design considerations. The usual Euclidian vector norm is denoted by $\|\cdot\|$. If it is not indicated otherwise, we keep this notation throughout this dissertation. In many applications, one takes $F = A_0$ and $G = B_0$ so that (1.24b) represents the normal behavior of the system without the uncertainties. Then $\xi(t)$ is an a-priori estimate of $x_0(t)$. Penalizing ξ amounts to penalizing the perturbation of the system during the test period assuming no fault has occurred. For simplicity, we will often use a simpler form such as a norm of the signal $\|v\|$ as the measure of "plant-hostility"and try to minimize $\|v\|$. The minimization is done in $L^2(R^m)$ space, the completion of the continuous functions with respect to the L^2-norm. Now we formally define the problem in Table 1.2.

Solving the problem

The monograph [13] discussed two different approaches for solving the problem defined in Table 1.2. One approach solves the inner optimization problem in the left-hand-side of (1.23) and obtains necessary conditions to reformulate the overall problem into an optimal control problem that can be solved by a software package called SOCS (Sparse Optimal Control Software) [7]. As a part of this solution approach will be used for solving a slightly different problem, which will be described in Chapter 3, here we do not review them. We now describe the second solution approach.

Table 1.2: Problem definition for designing optimal analog signal

Find auxiliary signal v that

$$\begin{aligned}
\text{minimize} \quad & \xi(T)^T W \xi(T) + \int_0^T \|v\|^2 + \xi^T U \xi \; dt \\
\text{subject to} \quad & v \in L^2(R^m) \\
& \dot{\xi} = F\xi + Gv, \quad \xi(0) = 0 \\
\text{such that} \quad & \inf \; \max \left\{ \mathcal{S}_v^0, \mathcal{S}_v^1 \right\} \geq 1, \\
& \text{for all } x_0, x_1, \nu_0, \nu_1 \text{ that satisfy} \\
& \dot{x}_0 = A_0 x_0 + B_0 v + M_0 \nu_0, \\
& E_0 y = C_0 x_0 + D_0 v + N_0 \nu_0, \\
& \dot{x}_1 = A_1 x_1 + B_1 v + M_1 \nu_1, \\
& E_1 y = C_1 x_1 + D_1 v + N_1 \nu_1.
\end{aligned}$$

First, the following equivalence is noted (see Theorem 2.1 in [20]):

$$\begin{aligned}
\inf_{[\ldots]} \; \max \left\{ \mathcal{S}_v^0(x_0(0), \nu_0, s), \mathcal{S}_v^1(x_1(0), \nu_1, s) \right\} &= \inf_{[\ldots]} \; \max_{0 \leq \beta \leq 1} \left\{ \beta \mathcal{S}_v^0 + (1-\beta) \mathcal{S}_v^1 \right\} \\
&= \max_{0 \leq \beta \leq 1} \; \inf_{[\ldots]} \left\{ \beta \mathcal{S}_v^0 + (1-\beta) \mathcal{S}_v^1 \right\} \quad (1.25)
\end{aligned}$$

where $[\ldots]$ is defined in (1.23b)-(1.23f). Denote

$$\begin{aligned}
\sigma(v, s) &= \max_{\beta \in [0,1]} \phi_\beta(v, s), \qquad \text{where} & (1.26a) \\
\phi_\beta(v, s) &= \inf_{[\ldots]} \left\{ \beta \mathcal{S}_v^0 + (1-\beta) \mathcal{S}_v^1 \right\}. & (1.26b)
\end{aligned}$$

Let $X^\perp$ and $X_\perp$ denote maximal rank right and left annihilators of matrix X. Thus $X^\perp X = 0$ and $X X_\perp = 0$. Also $\text{diag}(X, Y)$ denotes a block diagonal matrix with

diagonal entries X, Y. Let

$$\begin{pmatrix} F_0 & F_1 \end{pmatrix} = \begin{pmatrix} E_0 \\ E_1 \end{pmatrix}^{\perp}, x = \begin{pmatrix} x_0 \\ x_1 \end{pmatrix}, \nu = \begin{pmatrix} \nu_0 \\ \nu_1 \end{pmatrix}, M = \operatorname{diag}(M_0, M_1),$$

$$A = \operatorname{diag}(A_0, A_1), B = \begin{pmatrix} B_0 \\ B_1 \end{pmatrix}, N = \begin{pmatrix} F_0 N_0 & F_1 N_1 \end{pmatrix}, D = F_0 D_0 + F_1 D_1,$$

$$C = \begin{pmatrix} F_0 C_0 & F_1 C_1 \end{pmatrix}, P_\beta^{-1} = \operatorname{diag}(\beta P_{0,0}^{-1}, (1-\beta) P_{1,0}^{-1}), J_\beta = \operatorname{diag}(\beta J_0, (1-\beta) J_1).$$

We reformulate Problem (1.26b) as (1.27), (1.28):

$$\phi_\beta(v, s) = \inf_{\nu, x} x(0)^T P_\beta^{-1} x(0) + \int_0^s \nu^T J_\beta \nu \, dt \tag{1.27}$$

$$\dot{x} = Ax + Bv + M\nu, \tag{1.28a}$$

$$0 = Cx + Dv + N\nu. \tag{1.28b}$$

We assume that for some $\beta \in [0, 1]$, that $N_\perp^T J_\beta N_\perp > 0$ and the infinum exists. Then the problem is:

$$\min_v q(v) \text{ subject to } \max_{\substack{\beta \in \mathcal{B} \\ s \in [0,T]}} \phi_\beta(v, s) \geq 1.$$

Define

$$\lambda_{\beta,s} = \max_{v \neq 0} \frac{\phi_\beta(v, s)}{\xi(s)^T W \xi(s) + \int_0^s \|v\|^2 + \xi^T U \xi \, dt},$$

$$\lambda_\beta = \max_{s \leq T} \lambda_{\beta,s}, \tag{1.29}$$

so that we end up having to solve

$$\max_v \inf_{\nu, x} \left\{ x(0)^T P_\beta^{-1} x(0) - \lambda \xi(T)^T W \xi(T) + \int_0^s \nu^T J_\beta \nu - \lambda(\xi^T U \xi + \|v\|^2) \, dt \right\} \tag{1.30}$$

subject to (1.28). Later on we will see that λ in (1.30) and λ_β in (1.29) are essentially the same. The cost in (1.30) becomes

$$\mu(0)^T \mathrm{diag}(P_\beta^{-1}, 0)\mu(0) + \mu(s)^T \mathrm{diag}(0, -\lambda W)\mu(s) + \int_0^s \nu^T J_\beta \nu - \lambda \|v\|^2 - \mu^T \mathcal{Q}_\lambda \mu \, dt,$$

where $\mathcal{Q}_\lambda = \mathrm{diag}(0, \lambda U)$. A necessary condition for this minimization problem can be expressed as the following two-point boundary-value problem (TPBVP)

$$\frac{d}{dt}\begin{pmatrix}\mu \\ \zeta\end{pmatrix} = \begin{pmatrix}\bar{\Omega}_{11} & \bar{\Omega}_{12} \\ \bar{\Omega}_{21} & \bar{\Omega}_{22}\end{pmatrix}\begin{pmatrix}\mu \\ \zeta\end{pmatrix} = H\begin{pmatrix}\mu \\ \zeta\end{pmatrix}, \tag{1.31a}$$

$$V_0\begin{pmatrix}\mu(0) \\ \zeta(0)\end{pmatrix} + V_s\begin{pmatrix}\mu(s) \\ \zeta(s)\end{pmatrix} = 0. \tag{1.31b}$$

Here

$$V_0 = \begin{pmatrix}I & \mathrm{diag}(-P_\beta, 0) \\ 0 & 0\end{pmatrix}, V_s = \begin{pmatrix}0 & 0 \\ \mathrm{diag}(0, -\lambda W) & I\end{pmatrix},$$

$$\begin{pmatrix}\bar{Q}_{\lambda,\beta} & \bar{S}_{\lambda,\beta} \\ \bar{S}_{\lambda,\beta}^T & \bar{R}_{\lambda,\beta}\end{pmatrix} = \begin{pmatrix}\bar{B} \\ \bar{D}\end{pmatrix}\Gamma_{\lambda,\beta}^{-1}\begin{pmatrix}\bar{B} \\ \bar{D}\end{pmatrix}^T, \ \Gamma_{\lambda,\beta} = \mathrm{diag}(J_\beta, -\lambda I),$$

$$\bar{A} = \mathrm{diag}(A, F), \ \bar{B} = \begin{pmatrix}M & B \\ 0 & G\end{pmatrix}, \ \bar{C} = \begin{pmatrix}C & 0\end{pmatrix}, \ \bar{D} = \begin{pmatrix}N & D\end{pmatrix}$$

and $\bar{\Omega}_{11} = -\bar{\Omega}_{22}^T = \bar{A} - \bar{S}_{\lambda,\beta}\bar{R}_{\lambda,\beta}^{-1}\bar{C}$, $\bar{\Omega}_{12} = \bar{Q}_{\lambda,\beta} - \bar{S}_{\lambda,\beta}\bar{R}_{\lambda,\beta}^{-1}\bar{S}_{\lambda,\beta}^T$, $\bar{\Omega}_{21} = \bar{C}^T R_{\lambda,\beta}^{-1}\bar{C} + \mathcal{Q}_\lambda$.

The optimal v and ν satisfy

$$\begin{pmatrix} \nu \\ v \end{pmatrix} = \alpha \Gamma_{\lambda,\beta}^{-1} \begin{pmatrix} \bar{D}^T \bar{R}_{\lambda,\beta}^{-1} \bar{C} & \bar{D}^T \bar{R}_{\lambda,\beta}^{-1} \bar{S}_{\lambda,\beta}^T - \bar{B}^T \end{pmatrix} \begin{pmatrix} \mu \\ \zeta \end{pmatrix}, \tag{1.32}$$

where α is a to be determined scalar. We need to compute first λ_β which is the largest value of λ for which the TPBVP (1.31) is not well-posed for some $s \in [0, T]$.

However, a TPBVP

$$\dot{x} = Hx, \tag{1.33a}$$

$$0 = V_0 x(0) + V_s x(s), \tag{1.33b}$$

is well-posed if and only if $V_0 + V_s \Phi(s)$ is invertible where

$$\dot{\Phi} = H\Phi, \quad \Phi(0) = I. \tag{1.34}$$

Computation of Φ based on (1.34) is in general not practical, except on short intervals, since H is a Hamiltonian matrix and hence is not stable. When H is time-invariant, a simple and numerically efficient test of the well-posedness of (1.33) can be done by block diagonalizing H,

$$SHS^{-1} = \mathrm{diag}(A_f, -A_b), \tag{1.35}$$

where A_f and A_b do not have any eigenvalues with strictly positive real parts. Invertibility of $V_0 + V_s \Phi(s)$ is equivalent to the invertibility of the better conditioned

$$\Psi(s) \triangleq V_0 S^{-1} \mathrm{diag}(I, e^{A_b s}) + V_s S^{-1} \mathrm{diag}(e^{A_f s}, I).$$

Under our assumptions, $\lambda > \lambda_\beta$ if and only if $\Psi(s)$ is invertible for all $s \in [0, T]$. A λ-iteration scheme is implemented using an ODE solver with a root finder option. In particular, we solve

$$\dot{\Psi}_f = A_f \Psi_f, \quad \Psi_f(0) = I, \tag{1.36a}$$

$$\dot{\Psi}_b = A_b \Psi_b, \quad \Psi_b(0) = I, \tag{1.36b}$$

and test to see if the surface

$$0 = \det(\Psi(s)) \tag{1.37}$$

is crossed. Then λ_β is the infinum over the set of λ's for which the above system can be solved over the interval $[0, T]$ without any surface crossing. Then optimal λ^* and β^* are obtained by $\lambda^* = \max_{\beta \in \mathcal{B}} \lambda_\beta$. In the numerical computation, the maximum is taken by a simple search over a β–grid. Note that for $\lambda = \lambda_\beta$, the surface crossing may happen inside the interval $[0, T]$, say at T^*. This simply means that the optimal proper auxiliary signal can be defined over the interval $[0, T^*]$. Nothing is gained by increasing the test period and we can let $T = T^*$.

When $\lambda = \lambda^*$ and $\beta = \beta^*$, (1.31) has a non-zero solution. This solution allows us to compute the optimal proper auxiliary signal from (1.32). By computing a vector in the nullspace of

$$\begin{pmatrix} \operatorname{diag}(-\Psi_f(s), I) & \operatorname{diag}(I, -\Psi_b(s)) \\ V_0 S^{-1} & V_s S^{-1} \end{pmatrix},$$

we can find consistent values of $x_f(0)$ and $x_b(T)$. Then x_f and x_b satisfy

$$\dot{x}_f = A_f x_f, \tag{1.38a}$$

$$\dot{x}_b = -A_b x_b, \tag{1.38b}$$

where A_f and A_b do not have eigenvalues with positive real parts. Thus x_f and x_b can be computed from (1.38a) and (1.38b) respectively by forward and backward integration. Finally, the solution of the boundary value problem is

$$x = \begin{pmatrix} \mu \\ \zeta \end{pmatrix} = S^{-1} \begin{pmatrix} x_f \\ x_b \end{pmatrix}. \tag{1.39}$$

The optimal auxiliary signal is computed from (1.32) by choosing α such that $q(v)^{-1} = \sqrt{\lambda^*}$.

These procedures have been implemented in Scilab programs. Scilab is a software environment developed at INRIA [14]. It is used at a number of industries and has a large user base. It has a very similar syntax to MATLAB so that a Scilab program can be easily converted to MATLAB. However, Scilab has the advantage that it is publicly available and may be downloaded from `http://www.scilab.org/`. This algorithm was also implemented in MATLAB and tests have shown that the conversion did not alter relevant observations made in this dissertation. These are not industrial grade programs. They are, however, useful for constructing the solution to reasonable sized, well-posed problems.

The full mathematical description of the approach, along with algorithms and programs can be found in the monograph [13].

1.4.2 Fault detection tests

Applying an optimal proper detection signal v^* to the system of interest returns an output y^* that comes from one of the two disjoint output sets. We need a tool to test to which of the two models the output corresponds to. In this section we give main ideas of two different detection tests that were discussed in [13]. These tests can also

be applied for a new detection signal which will be designed later in this dissertation.

Standard test

This test is to determine the realizability of the input-output pair (v^*, y^*). That is, to examine the existence of the state variable x and uncertainty variable ν that, for the given (v^*, y^*), satisfy (1.18) and (1.20) at any time of the test period $[0, T]$. Then, the problem is to solve

$$\min x_i(0)^T P_{i,0}^{-1} x_i(0) + \int_0^s \nu_i^T J_i \nu_i \, dt, \forall s \in [0, T] \tag{1.40a}$$

subject to

$$\dot{x}_i = A_i x_i + B_i v^* + M_i \nu_i, \tag{1.40b}$$

$$E_i y^* = C_i x_i + D_i v^* + N_i \nu_i \tag{1.40c}$$

for each $i = 0, 1$. If the minimal objective value, the least amount of uncertainty required to make (v^*, y^*) consistent with model i, is equal to and greater than one, the model i is rejected. Theoretically, solving (1.40) for one of the i's should be enough for fault detection since two models are ensured not to produce any common output. However, using two realizability tests can lead to an early detection by rejecting a model before the end of the test. There are several other potential advantages to the filter test. Even if there is a different fault that v^* was designed for, it might still happen that we reject the normal model and detect that a fault exists. If the faulty model is also rejected that tells us the fault is of a different kind. Even if the normal model is not rejected, a value close to the rejection level could indicate that more frequent monitoring may be called for.

Separating hyperplane test

When both output sets $\mathcal{Y}_i(v)$ are convex (see Lemma 2.3.2 in [13]), a hyperplane can be placed separating the two so that each $\mathcal{Y}_i(v)$ is on a different side of the plane.

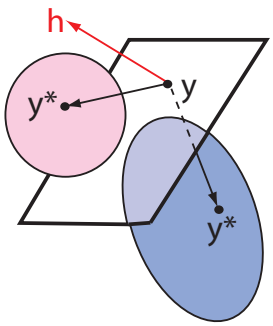

Figure 1.4: Hyperplane Test

If we let h be a normal vector to the separating hyperplane, y be a point on the hyperplane, then the test can be described by

$$\int_0^T h(t)^T (y(t) - y^*(t))\, dt \lessgtr 0. \tag{1.41}$$

In other words, once we know what h and y are, the hyperplane test amounts to a single integration as y^* is received. We immediately see its advantage in speed. Like in the signal design problem, the monograph [13] describes two different ways of computing the hyperplane. Here we focus on one of them, an analytical approach. We will examine the other approach later in Chapter 3.4, but with a different type of detection signal application.

When we use minimum proper continuously-varying detection signal v^*, the two output sets $\mathcal{Y}_i(v^*)$ are disjoint, but the closures of the two sets $\bar{\mathcal{Y}}_i(v^*)$ intersect at least at one point of the boundaries. We can find an intersecting point y and the hyperplane that passes y and is tangent to the sets by solving the constrained optimization

problem:

$$\min x(0)^T P_{\beta^*}^{-1} x(0) + \int_0^s \nu^T J_{\beta^*} \nu \, dt, \forall s \in [0,T] \tag{1.42a}$$

subject to

$$\dot{x} = Ax + Bv^* + M\nu \tag{1.42b}$$

$$\begin{pmatrix} E_0 & 0 \\ 0 & E_1 \end{pmatrix} \begin{pmatrix} y_0 \\ y_1 \end{pmatrix} = \begin{pmatrix} C_0 & 0 \\ 0 & C_1 \end{pmatrix} x + \begin{pmatrix} D_0 \\ D_1 \end{pmatrix} v^* + \begin{pmatrix} N_0 & 0 \\ 0 & N_1 \end{pmatrix} \nu \tag{1.42c}$$

$$0 = y_0 - y_1. \tag{1.42d}$$

Then the h is the Lagrange multiplier associated with the constraint (1.42d) and $y = y_0 = y_1$. The formulas of h and y for v^* that minimizes $q(v) = \int_0^T \|v\|^2 \, dt$ are given in Theorem 3.3.5 in [13].

1.5 Outline of Dissertation

In the following chapter, we look at the practical side of the optimal analog signal design algorithm. Two separate factors that have major effects on the performance of the optimal signal design software are investigated. In Chapter 3, in parallel with the growing connection with microprocessors, use of digital signals are considered for identifying faulty behavior of physical systems. We introduce an optimal digital design algorithm and discuss fault detection tests using such signals. Some of the extension to this research are discussed in Chapter 4. Finally, in Chapter 5 final conclusions are given while contributions of this work are summarized. Software codes we have developed for the algorithms are given in Appendix A.

Chapter 2

Practical Issues of The Optimal Analog Signal Design Algorithm

In this section, we study closely the optimal analog signal design algorithm that was summarized in Chapter 1.4. We point out two issues related to use of this algorithm and its software. The algorithm is based on a mathematical model that can account for almost every possible dynamic behavior of a physical system. In the following section, it will be shown that such broad accountability of a model may be limited in the purpose of obtaining algorithm solutions with better quality. Following that, we discuss a way to reduce computational cost of the algorithm and propose a suboptimal signal design algorithm. Through a computational experiment and analysis, the advantages and disadvantages are examined.

2.1 Managing Solution Quality Effected by Model Uncertainty

When describing a process in a quantitative mathematical expression, one often faces the discrepancy between the physical system and the mathematical model. In general,

the gap arises from two sources of uncertainty: unknown noises and disturbances that act on the physical system and unknown coefficients in the mathematical model. As an effort to capture the physical system more accurately in a model, the uncertainties are often included as a random variable or stochastic processes having certain probability distributions [26], [34]. Also, a different approach considers the uncertainties as vector functions belonging to a bounded set in a vector space. As many other studies like [6], [25], [46] have done, the optimal analog signal design algorithm assumes for a simple input-output relation equations with the latter type of uncertainty variables to mirror the real system behaviors.

2.1.1 Incorporating uncertainty into a model

Suppose the system under consideration has continuous dynamics and can be described by a linear input-output relationship:

$$\dot{x} = Ax + Bv \tag{2.1a}$$

$$y = Cx + Dv \tag{2.1b}$$

where x, v, y are state, detection signal, and output, respectively. In order to introduce robustness and to design auxiliary signals for (2.1), we consider the following system equations for modeling both *normal* ($i = 0$) and *faulty* behaviors ($i = 1$):

$$\dot{x}_i = (A_i + \delta A_i)x_i + (B_i + \delta B_i)v + \overline{M}_i w_i \tag{2.2a}$$

$$y = (C_i + \delta C_i)x_i + (D_i + \delta D_i)v + \overline{N}_i w_i \tag{2.2b}$$

The terms with w_i in (2.2a) and (2.2b) are linearly added to the original equations modeling any kind of *additive uncertainty* such as measurement errors and input

disturbances. The *model uncertainty* is represented by $(\delta A_i, \delta B_i, \delta C_i, \delta D_i)$. The uncertainty in the four coefficient matrices often vary depending on which equation it consists of and which variable it corresponds to. Thus, it is reasonable to consider a model uncertainty as follows:

$$\begin{pmatrix} \delta A_i & \delta B_i \\ \delta C_i & \delta D_i \end{pmatrix} = \begin{pmatrix} \widetilde{M}_i \\ \widetilde{N}_i \end{pmatrix} \Delta \begin{pmatrix} H_i & G_i \end{pmatrix} \tag{2.2c}$$

where $\widetilde{M}_i, \widetilde{N}_i, H_i, G_i$ are the matrices providing structure and scaling the uncertainty and Δ is a matrix of uncertainty parameters. Such a highly structured uncertain model has also studied in [43], [46].

The amount of the uncertainty in a model can be limited by inequality constraints. We consider an energy-type constraint on the additive uncertain quantities and uncertain initial condition

$$x_i(0)^T P_{0,i} x_i(0) + \int_0^T \|w_i\|^2 dt \;\; < \;\; 1, \tag{2.2d}$$

Also, it is assumed that the largest singular value of Δ satisfies

$$\bar{\sigma}(\Delta(t)) \;\; \leq \;\; 1. \tag{2.2e}$$

Note that scaling the matrices $\overline{M}_i, \overline{N}_i, H_i, G_i$ brings the same result as changing the bound in (2.2d),(2.2e). In general, when the allowable uncertainty bound is higher, we can design a more robust detection signal. However, it turns out that as the level of uncertainty increases in a model, the quality of the detection signal quickly declines.

We first show how the system equations and uncertainty constraints defined in (2.2) can be transformed into the form of (1.18) and (1.20) so that the optimal analog signal design algorithm can be applied.

Letting

$$\nu_{a,i} = \Delta\nu_{c,i},\ \nu_{b,i} = w, \tag{2.3}$$

we rewrite (2.2a)-(2.2b) as

$$\begin{aligned} \dot{x}_i &= A_i x_i + B_i v + \begin{pmatrix} \widetilde{M}_i & \overline{M}_i \end{pmatrix} \begin{pmatrix} \nu_{a,i} \\ \\ \nu_{b,i} \end{pmatrix} && \text{(2.4a)} \\ 0 &= G_i x_i + H_i v - \nu_{c,i} && \text{(2.4b)} \\ y &= C_i x_i + D_i v + \begin{pmatrix} \widetilde{N}_i & \overline{N}_i \end{pmatrix} \begin{pmatrix} \nu_{a,i} \\ \\ \nu_{b,i} \end{pmatrix} && \text{(2.4c)} \end{aligned}$$

We relax the first condition of (2.3) to be

$$\int_0^s \|\nu_{a,i}\|^2 - \|\nu_{c,i}\|^2 dt < 0, \quad 0 \le s \le T. \tag{2.5}$$

The uncertainty bounds are slightly relaxed by adding (2.5) and (2.2d): for $0 \le s \le T$,

$$x_i(0)^T P_{0,i} x(0) + \int_0^s \|\nu_{a,i}\|^2 + \|\nu_{b,i}\|^2 - \|\nu_{c,i}\|^2 dt < 1 \tag{2.6}$$

Note that when we let $x = \begin{pmatrix} x_0 \\ \\ x_1 \end{pmatrix}$ and $\nu = \begin{pmatrix} \nu_0 \\ \\ \nu_1 \end{pmatrix}$ where $\nu_i = \begin{pmatrix} \nu_{a,i}^T & \nu_{b,i}^T & \nu_{c,i}^T \end{pmatrix}^T$ and combine (2.4b) and (2.4c), we get a model in the form of (1.18) and (1.20) with $E^T = \begin{pmatrix} 0 & I \end{pmatrix}^T$. Thus, it can be solved by the algorithm described earlier. This rewriting introduces some conservatism. But it is important to note that this conservatism

does not introduce false positives or missed faults as with other approaches. Rather the conservatism results in larger than minimal test signals.

2.1.2 Computational experiments with various uncertainty levels

We carefully study an example problem while adjusting the level of uncertainty in its model.

The test problem

The equalized and linearized model of a single engine F-16 supersonic test vehicle is given by

$$\dot{x}_0 = \begin{pmatrix} -0.1689 & 0.0759 & -0.9952 \\ -26.859 & -2.5472 & 0.0689 \\ 9.3603 & -0.1773 & -2.4792 \end{pmatrix} x_0 + \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} v \tag{2.7a}$$

$$y = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0.9971 & 0.0755 \end{pmatrix} x_0 \tag{2.7b}$$

where the state vector $x = \begin{pmatrix} \xi_1 & \xi_2 & \xi_3 \end{pmatrix}$ represents side-slip, roll rate and yaw rate; a detection vector v is composed of side-slip acceleration command ($= \ddot{\xi}_1$) and stability axis roll acceleration command ($= \ddot{\zeta}$); and the vector y outputs side-slip and stability axis roll rate ($= \dot{\zeta}$) [42].

We wish to take into consideration uncertainty for the given model. Suppose that all possible normal behaviors of the test vehicle can be modeled by (2.4) and (2.6)

with $i = 0$ where A_0, B_0, C_0 are the coefficient matrices in (2.7) and

$$D_0 = 0,\ \begin{pmatrix} \widetilde{M}_0 & \overline{M}_0 \end{pmatrix} = \begin{pmatrix} I_{3\times 3} & I_{3\times 3} & | & 0_{3\times 2} \end{pmatrix},\ \begin{pmatrix} \widetilde{N}_0 & \overline{N}_0 \end{pmatrix} = \begin{pmatrix} 0_{2\times 6} & I_{2\times 2} \end{pmatrix},$$

$$G_0 = \begin{pmatrix} \widetilde{G}_0 \\ 0_{3\times 3} \end{pmatrix},\ \widetilde{G}_0 = \begin{pmatrix} .01 & .001 & .01 \\ 4.13 & .1 & .001 \\ 1 & .001 & .1 \end{pmatrix},\ H_0 = \begin{pmatrix} 0_{4\times 2} & 0.1 I_{2\times 2} \end{pmatrix},\ P_0 = I.$$

Faulty conditions simulating an electrical interruption to a flight control computer's input channels may be represented with the same model coefficients except

$$A_1 = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix},\ G_1 = \begin{pmatrix} \widetilde{G}_1 \\ 0_{3\times 3} \end{pmatrix},\ \widetilde{G}_1 = \begin{pmatrix} .01 & .1 & .01 \\ 0 & .1 & .1 \\ 0 & 0 & .1 \end{pmatrix},\ H_1 = \begin{pmatrix} 0_{4\times 2} & 0.11\, I_{2\times 2} \end{pmatrix}.$$

Finally, we take $U = W = I, F = A_0, G = B_0$ in (1.24) and $T = 1$, so the control is designed to both be small and to not overly affect the non-faulty system.

Computational Studies

In this study we adjust only the model uncertainty. The effect of different amounts of additive uncertainty on the optimal detection signal is straightforward and will be discussed later. The adjustment is carried out by two scaling factors of model uncertainty, w_G and w_H.

Initial testing showed that the algorithm failed if $w_G \geq 0.8$ and $w_H \geq 1$. In these cases there is too much model uncertainty for guaranteed detection. For $w_G = w_H = 0$ we have only additive uncertainty and the algorithm easily solves the problem. Accordingly we investigate the solution when both $0 \leq w_G < 0.8$ and $0 \leq w_H \leq 1$.

Because of the way the algorithm is executed, the key part of the algorithm is in determining the optimal values of β, λ. The software returns several functions and their graphs. Here we focus on the graph of λ_β, the test signal v, and the a-priori estimate assuming no fault has occurred. The variable "grid" is the number of mesh points used in the β interval to compute the max β^*, λ^* . We used a uniform mesh but future versions of the software could benefit from a variable mesh. Figures 2.1–2.3 show the results for $w_G = 0$ and several values of w_H. In Figure 2.2 it should be noted that v is a proper signal if and only if $-v$ is proper and sometimes the software switches sign. Also v has two components. For a given w_H one component of v is in the outer envelope and the other is in the inner envelope. Comparing Figure 2.8 to Figure 2.3 we see that in the presence of more model uncertainty that not only is the signal itself larger but its effect on the normal model is larger.

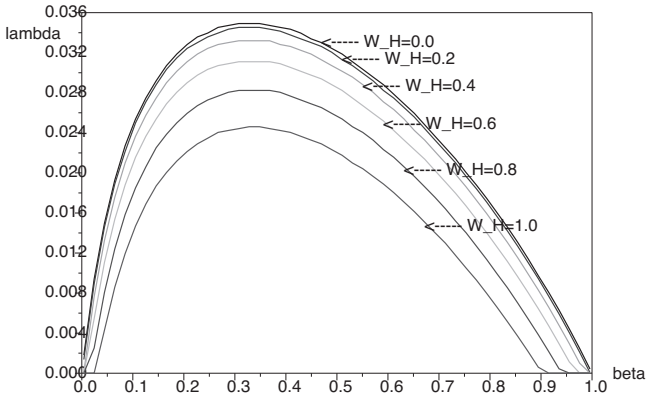

Figure 2.1: Graphs of λ_β for $w_G = 0$, β-grid=50.

The pictures changes dramatically as we raise w_G as shown in Figures 2.4–2.6.

Finally if we take the largest value of w_G for which we got convergence we get Figures 2.6–2.8.

Notice in Figures 2.6–2.8 that a grid of 150 was used. A finer mesh was needed in order for the algorithm to produce a solution. Comparing Figure 2.7 to Figure 2.2

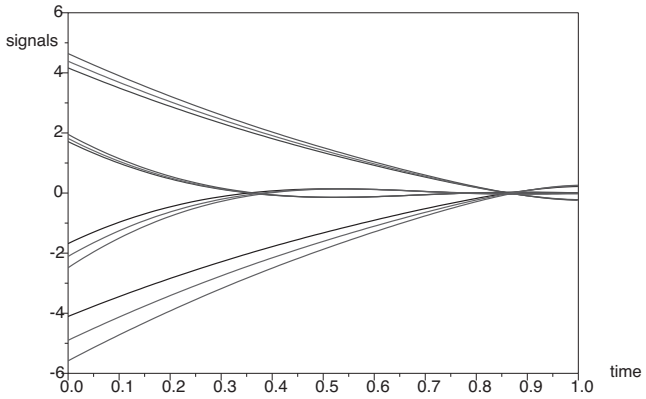

Figure 2.2: Auxiliary signals for $w_G = 0$, β-grid=50.

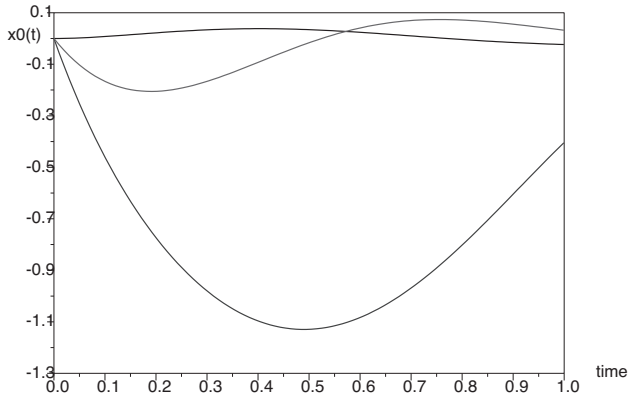

Figure 2.3: A priori estimate for $w_G = 0$, $w_H = 0$, β-grid=50.

we see that there is some change in the shape of the auxiliary signal as w_G, w_H vary, but the most dramatic difference is in the magnitude of the test signal so that getting the magnitude right is the more crucial part of the calculation. This is the same as computing λ^* correctly. If we look at Figures 2.1, 2.4, 2.5, we see that the curve is smooth on top. It would appear that the value of λ^* should be relatively insensitive to the grid. The situation changes as the amount of model uncertainty increases until λ^* no longer occurs at a horizontal tangent but instead occurs at a peak as in Figure 2.6. Consider Figure 2.9 which shows the same result as Figure 2.6 but on a coarser

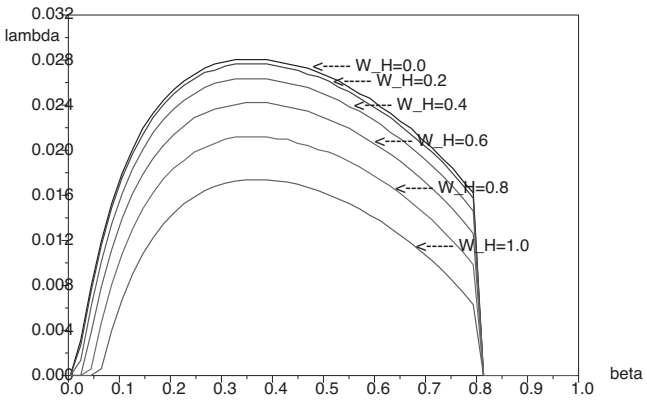

Figure 2.4: Graphs of λ_β for $w_G = 0.6$, β-grid=50.

grid. Figure 2.9 shows there can be shifts in λ^*, β^* for coarser grids especially at the larger w_G values.

Table 2.1 illustrates both the larger test signal needed to overcome model uncertainty when w_G, w_H are made nonzero and the numerical robustness of the solution on various grids. Table 2.2 shows the need for finer grids in the presence of more uncertainty and how the coarse grid solution can give a misleading value for $\|v\|$.

Table 2.1: Computation statistics on coarse grids

w_G	w_H	grid	β^*	λ^*	$\|v_1\|$	$\|v_2\|$
0	0	50	0.3077	0.0349	3.9739	0.2522
0.2	0.2	25	0.3360	0.0340	7.3277	0.4294
0.2	0.2	50	0.3279	0.0340	6.6824	0.3923
0.2	0.2	100	0.3340	0.0340	7.1742	0.4206
0.2	0.2	150	0.3293	0.0340	6.8098	0.3996

In using these algorithms it is important to recognize failure of the algorithm.

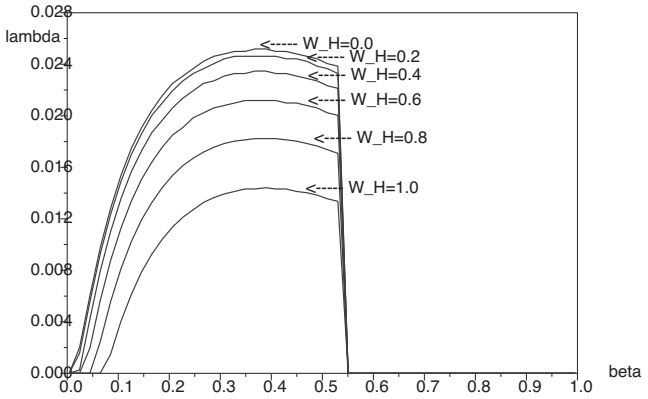

Figure 2.5: Graphs of λ_β for $w_G = 0.7$, β-grid=50.

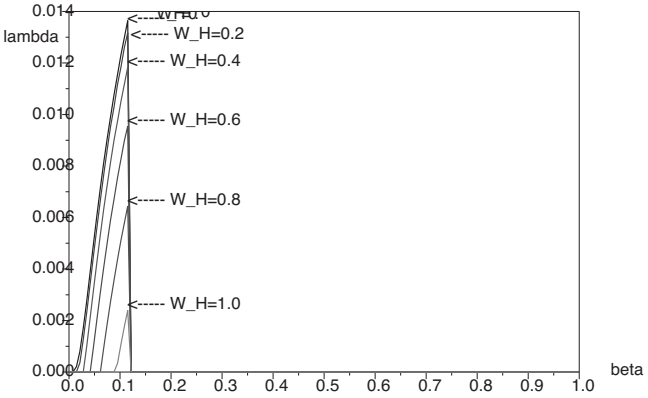

Figure 2.6: Graphs of λ_β for $w_G = 0.795$, β-grid=150.

Failure can occur either (1) because the problem has no solution, (2) because of insufficient fineness of the β mesh, or (3) because the problem is not computable due to illconditioning. These cases are indicated by any of the following: Scilab returns an error message and does not perform the requested calculations, the value of λ^* can be close to zero as in line 1 of Table 2.3, the auxiliary signal v can be close to zero as in line 3 of Table 2.3, or the domain of the auxiliary signal is not $[0, T]$. The later is usually caused by failure of the λ iteration. In case of failure of the algorithm, even on finer grids, it is recommended that the user scale down the size of the model

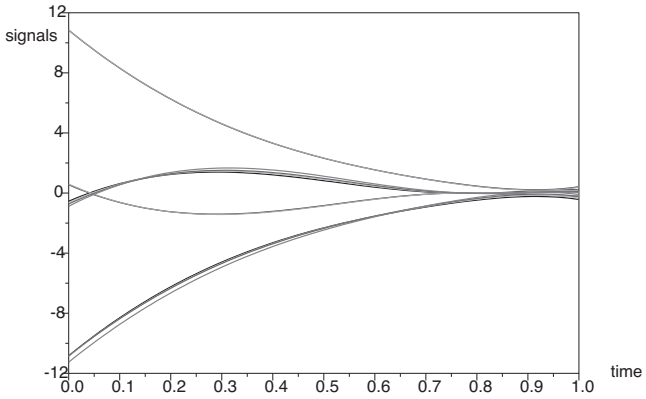

Figure 2.7: Auxiliary signals for $w_G = 0.795$, β-grid=150

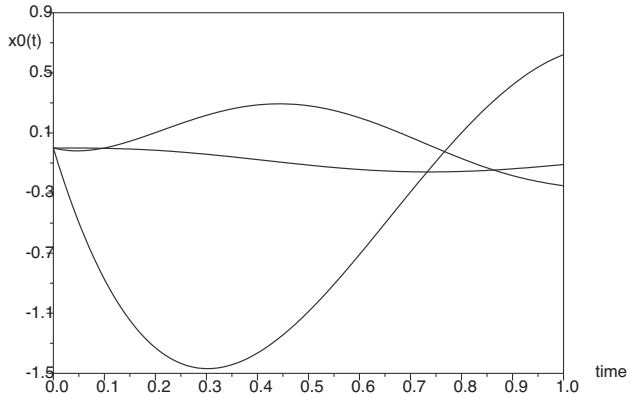

Figure 2.8: A priori estimate for $w_G = 0.795$, β-grid=150.

uncertainty as done by our w_G, w_H factors. This will often result in a test signal that can still be useful even though it will be somewhat less robust. Particularly, if the noise has a random nature, the auxiliary signals often work for noises much larger than the noise bound. That is because the worse case noise is itself a solution of an optimization problem and is often smooth and of a special form. As mentioned, the shape of the auxiliary signal is often robust. However, it is best to check it with a finer grid. Only when the λ^*, β^* values have settled down should the norm of the solution be accepted.

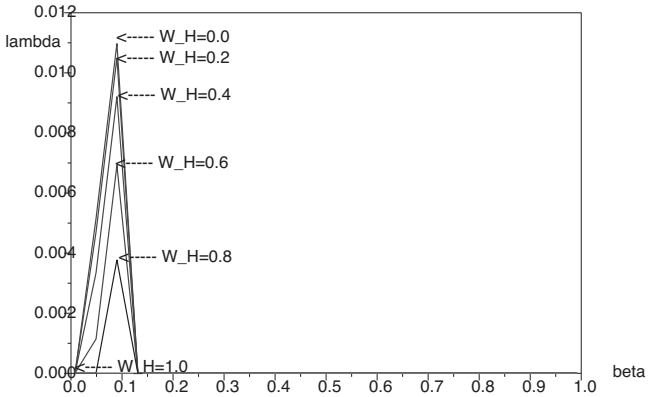

Figure 2.9: Graph of λ_β for $w_G = 0.795$, β-grid=25.

Table 2.2: Computation statistics on coarse grids for fixed $w_H = 0.8$

w_G	grid	β^*	λ^*	$\|v_1\|$	$\|v_2\|$
0.795	25	0.0900	0.0038	62.3248	2.6191
0.795	50	0.1052	0.0054	33.3328	1.2370
0.795	100	0.1127	0.0062	22.5194	0.8016
0.795	150	0.1151	0.0064	20.9995	0.7392

2.1.3 Why it happens: a geometrical explanation

We observed that different quantities of uncertainty, in particular model uncertainty, effect the quality of the computed optimal detection signal. A medium level of uncertainty required some minor adjustments in the software to improve the accuracy of solutions whereas a large quantity of uncertainty led to no solution. In this section we give a geometrical explanation for such results.

Table 2.3: Algorithmic failure indicated cases ($w_G = 0.795$)

w_H	grid	β^*	λ^*	$\|v_1\|$	$\|v_2\|$
1.0	25	0.0080	0.0000	2.6035	0.2106
1.0	50	0.1052	0.0014	34.7255	1.4605
0.2	50	0.1052	99.2188	0.0000	0.0000

Consider an uncertain model

$$\begin{aligned} \dot{x} &= Ax + Bv + M\nu \\ y &= (C + \delta C)x + N\nu \\ 1 &> \mathcal{S}(\delta C, \nu, x(0)) \end{aligned}$$

where $\delta C, \nu$ are model and additive uncertainties, respectively. Similarly to the way we obtained (1.19), we let linear operator $\mathcal{L}(h)$ be the solution of $\dot{z} = Az + h,\ z(0) = 0$. Then, the set of model-outputs y for given v is

$$\mathcal{Y}(v) = \mathcal{Y}(0) + \mathcal{W}(v)$$

where

$$\mathcal{W}(v) = \left\{ (C + \delta C)\mathcal{L}Bv \,\middle|\, \mathcal{S}(\delta C, \nu, x(0)) < 1 \right\} \tag{2.8a}$$

$$\mathcal{Y}(0) = \left\{ \left[(C + \delta C)\mathcal{L}M + N \right]\nu + (C + \delta C)e^{At}x(0) \,\middle|\, \mathcal{S}(\delta C, \nu, x(0)) < 1 \right\} \tag{2.8b}$$

$\mathcal{Y}(0)$ can be thought as the set of all possible system behaviors if no detection signal were used. $\mathcal{W}(v)$ contains vectors that move and/or expand $\mathcal{Y}(0)$.

Suppose there is only additive uncertainty not model uncertainty, that is, $\delta C = 0$.

Then, $\mathcal{W}(v)$ equals a single vector $C\mathcal{L}Bv$ instead of a set of vectors as in (2.8a). As shown on the left side of Figure 2.10, application of the single vector $\mathcal{W}(v)$ causes the output set $\mathcal{Y}(0)$ to move. Therefore, no matter how many outputs of normal and faulty models overlap, as long as the two output sets are not completely identical, there is a v that separates the two sets.

On the other hand, when model uncertainty is considered i.e. $\delta C \neq 0$, a set of vectors in (2.8a) not only transports but also expands the output set $\mathcal{Y}(0)$. Note that the bigger the magnitude of v is, the more expansion occurs. Thus, it is possible that designed normal and faulty system equations hold an excessive amount of model uncertainty for which no v can separate their output sets. The picture on the right of Figure 2.10 illustrates this.

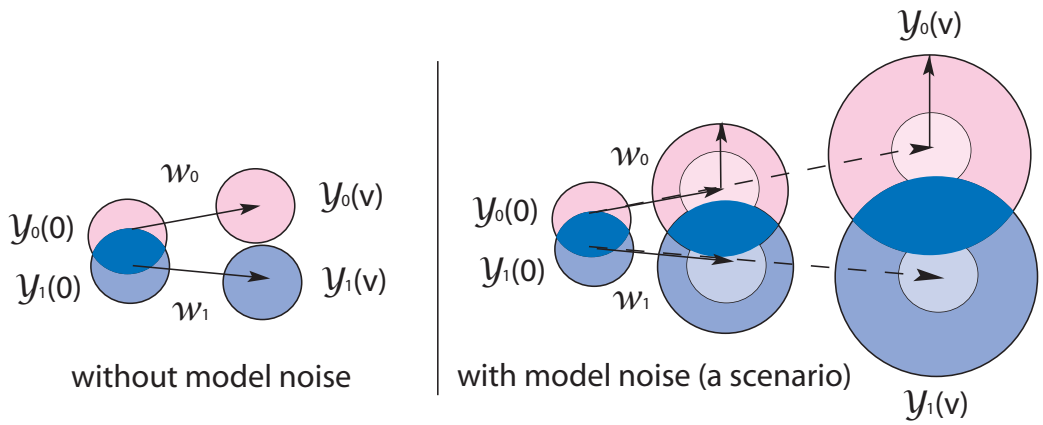

Figure 2.10: The set of outputs with and without model uncertainty

2.1.4 Conclusion

Through a careful examination of a test problem we have provided the first examination of the effect of model uncertainty in the algorithms developed in [13]. The cause of the effect has been explained geometrically. Ways to recognize algorithmic

difficulty have been also discussed and illustrated along with suggestions on how to deal with these circumstances.

2.2 Reducing Computational Cost

In the optimal analog signal design algorithm, part of the compuational effort comes from the β parameter which was introduced to replace the discontinuous maximization problem

$$\inf_{[...]} \max\{\mathcal{S}_v^0, \mathcal{S}_v^1\} \tag{2.9}$$

with a continuous one

$$\inf_{[...]} \max_{0\leq\beta\leq1} \beta\mathcal{S}_v^0 + (1-\beta)\mathcal{S}_v^1. \tag{2.10}$$

Note that [...] are the constraints defined in (1.23b)-(1.23f). To solve (2.10), the original software implements a simple search over a β grid as the function of beta is in general not in a special form such that an advanced search can be performed. Although the signal design can be done offline, and hence, is not required to be realtime, a versatile system that requires frequent fault detection tests can benefit from faster design of detection signals.

In this section, we propose a suboptimal approach that reduces computational time of the original algorithm. We also compare the optimal and suboptimal approaches and discuss the tradeoff between computational times and objective values. In the special case where there exists additive uncertainty only, an upper bound is found. A computational experiment follows providing an insight into the ratio between optimal and suboptimal cost.

2.2.1 A suboptimal approach

Suppose that uncertainty bound is given as

$$\mathcal{S}_v^0 + \mathcal{S}_v^1 < 2 \tag{2.11}$$

which is, of course, equivalent to

$$\frac{1}{2}\mathcal{S}_v^0 + \frac{1}{2}\mathcal{S}_v^1 < 1. \tag{2.12}$$

This uncertainty constraint is a relaxation of the original uncertainty bound $\mathcal{S}_v^0 < 1, \mathcal{S}_v^1 < 1$ and has the advantage that the perturbations are now measured by a single inner product norm given by (2.12). Let us use a tilde to denote the terms found with this norm. Then a proper signal is the v such that $\widetilde{\sigma}(v,s) \geq 1$ for all $s \in [0,T]$ where

$$\widetilde{\sigma}(v,s) = \inf_{\substack{\nu_0,\nu_1,\\ x_0,x_1}} \frac{1}{2}\mathcal{S}_v^0(x_0(0),\nu_0,s) + \frac{1}{2}\mathcal{S}_v^1(x_1(0),\nu_1,s) \tag{2.13}$$

subject to (1.23b)-(1.23f). That is,

$$\widetilde{\sigma}(v,s) = \phi_{\frac{1}{2}}(v,s).$$

$\phi_\beta(v,s)$ is defined in (1.26b). Thus, the alternative approach amounts to setting $\beta = 0.5$ without a grid search and proceding to the computation of the test signal.

2.2.2 Comparison of optimal and suboptimal approaches

The suboptimal approach achieves a great reduction in computational cost. It can be easily seen that it reduces the computational cost to about $\frac{1}{\text{number of grid points used}}$ of that of the optimal approach. But then what is the cost of not using the optimal

approach in favor of faster signal design?

To facilitate the cost comparison of the standard approach and the suboptimal approach, we introduce some terminology. The terms *proper* and *minimal proper* will refer to the approach that uses β search. The terms *subproper* and *minimal subproper* will refer to the approach that uses $\beta = 0.5$. Since the subproper bound admits a larger set of uncertainties, we immediately have that if a v is subproper, then it is a proper signal. If v^* denotes a minimal proper test signal and $\underline{v}^*$ denotes a minimal subproper test signal, then $q(v^*) \leq q(\underline{v}^*)$ where $q(v)$ is the cost function defined in (1.24).

Additive Uncertainty Only

The case when there is just additive uncertainty differs from the model uncertainty case in several important ways. For one, if v is proper (or subproper), then taking αv for $\alpha > 1$ is still proper (or subproper). This is not the case with model uncertainty since it is then possible for the output set $\mathcal{Y}_i$ to grow in size as α increases. Note Figure 2.10. Accordingly, we focus first on the simpler but still important case of additive uncertainty. In this case we can establish an upper bound.

PROPOSITION 2.1. *Suppose that we have linear models in the form of (1.18), (1.20) and that we are in the additive uncertainty case. Let v^* and $\underline{v}^*$ be the minimal proper and minimal subproper test signals respectively. Then*

$$q(v^*) \leq q(\underline{v}^*) \leq 2q(v^*). \tag{2.14}$$

Proof The first inequality naturally follows as v^* is optimal and $\underline{v}^*$ is a suboptimal solution to the problem in Table 1.2.

Note that when there is only additive noise, the size of an output set stays the same regardless the size of applied test signal. Also notice that the constraints are

linear and both $\phi_\beta(v)$ and $q(v)$ are quadratic in v. Let $\mathcal{Y}(b)$ be normal and faulty output sets given by the uncertainty bound b. If v^* is minimal proper that separates $\mathcal{Y}(\mathcal{S}_v^0 < 1, \mathcal{S}_v^1 < 1)$, then $\sqrt{2}v^*$ is minimal proper for the noise bound $\mathcal{S}_v^0 < 2, \mathcal{S}_v^1 < 2$. Since

$$\mathcal{Y}(\mathcal{S}_v^0 + \mathcal{S}_v^1 < 2) \subset \mathcal{Y}(\mathcal{S}_v^0 < 2, \mathcal{S}_v^1 < 2), \tag{2.15}$$

$\sqrt{2}v^*$ is also a proper signal for the noise bound $\mathcal{S}_v^0 + \mathcal{S}_v^1 < 2$. As $\underline{v}^*$ is the minimal proper for $\mathcal{S}_v^0 + \mathcal{S}_v^1 < 2$, i.e. minimum subproper, $q(\underline{v}^*) \leq q(\sqrt{2}v^*) = 2q(v^*)$. ■

In chapter 2.1.2, an example is given which is based on an aircraft model. This example has the nice feature that it includes parameters to set the model uncertainty. Here we introduce additive noise scaling parameters θ_i in this model and observe the change in cost when we vary the amount of additive uncertainty.

Suppose that we set the model uncertainty scaling factors w_G, w_H equal to zero so that there is no model uncertainty and only additive uncertainty. Then we get Figure 2.11.

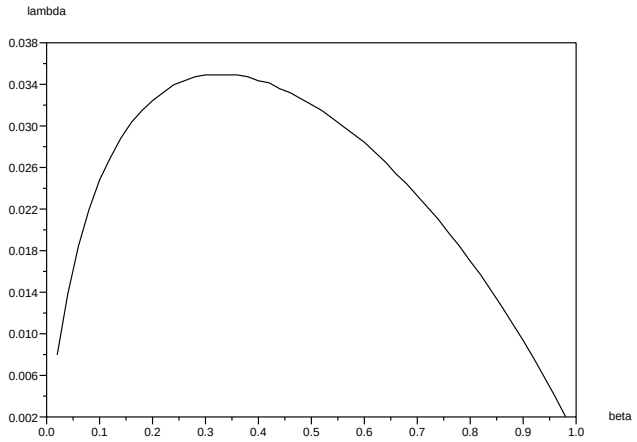

Figure 2.11: Graph of λ_β for additive uncertainty case $[w_G = w_H = 0,\ \theta_0 = \theta_1 = 1]$

Note that In the standard algorithm the function λ_β is the reciprocal of $q(v)$ for the minimal test signal satisfying $\phi_\beta(v) \geq 1$. A maximization is done over β to find β^*. Then

$$\lambda_{\beta^*} = \frac{1}{q(v^*)}. \tag{2.16}$$

Similarly,

$$\lambda_{0.5} = \frac{1}{q(\underline{v}^*)}. \tag{2.17}$$

λ_{β^*} is also denoted by λ^*.

From Figure 2.11 we see that $\beta^* = 0.3$, $\lambda^* = 0.0349$, $\lambda_{0.5} = 0.321$ and thus $\sqrt{\lambda^*/\lambda_{0.5}} = 1.043$. Thus the minimal suboptimal proper is only 4.3% larger in the $\sqrt{q}$ norm than the minimal proper signal. For this case we see that the minimal subproper is very similar to the minimal proper in norm. The test signals $v = \begin{pmatrix} v_1 \\ \\ v_2 \end{pmatrix}$ also have very similar shapes as shown in Figure 2.12 and 2.13.

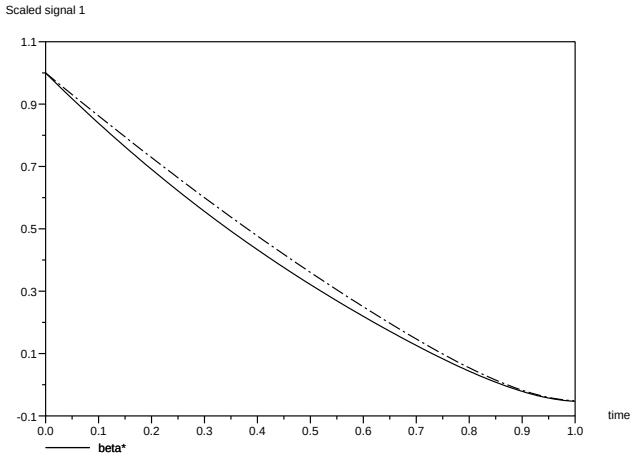

Figure 2.12: Scaled v_1 for β^* (solid) and $\beta = 0.5$ (dashed) $[w_G = w_H = 0,\ \theta_0 = \theta_1 = 1]$

Suppose now that we multiply the matrices $\overline{N}_i$ and $\overline{M}_i$ by a weighting factor of

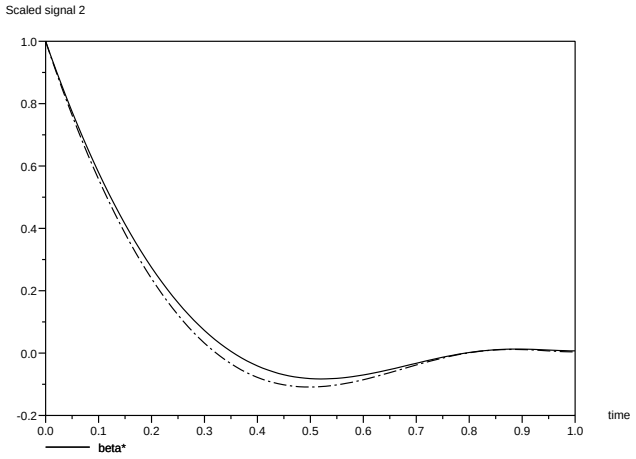

Figure 2.13: Scaled v_2 for β^* (solid) and $\beta = 0.5$ (dashed) $[w_G = w_H = 0,\ \theta_0 = \theta_1 = 1]$

10 by setting $\theta_i = 10$. This is the same as assuming that there is 10 times as much additive noise in model i. Figure 2.14 shows the graph of λ_β when we multiply just $\overline{M}_0, \overline{N}_0$ by 10 (solid line) or when we multiply just $\overline{M}_1, \overline{N}_1$ by 10 (dashed line).

We observe that the β^* is now away from 0.5. The weights that are higher than 10 result in β^*'s that are even further away from 0.5, that is, close to 0 or 1. Table 2.4 shows the results from these cases. We observe that the suboptimal cost is no more than twice of the optimal cost as it was argued in Proposition 2.1 .

Model Uncertainty

When model uncertainty is included the situation is more complex in that there need not be a proper test signal if the uncertainty becomes too large. Returning to the previous example, we first observe a significant shape and size difference between optimal and suboptimal signals when model uncertainty exists.

We took the minimal proper auxiliary signal and the minimum subproper auxiliary signal for the values of $\omega = w_G = w_H$ from 0.0 to 0.4 while $(\theta_0, \theta_1) = (1, 10)$. We then

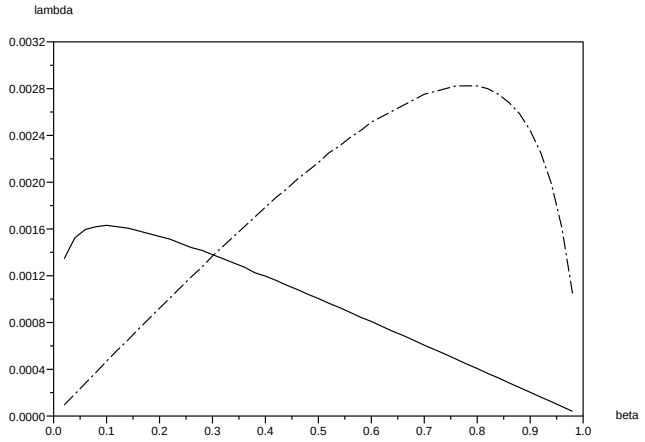

Figure 2.14: Graphs of λ_β for additive uncertainty case $[w_G = w_H = 0, \theta_0 = 1, \theta_1 = 10]$ (solid), $[w_G = w_H = 0, \theta_0 = 10, \theta_1 = 1]$ (dashed)

Table 2.4: Computational results with additive uncertainty weighting

θ_0	θ_1	β^*	λ^*	$\lambda_{0.5}$	$q(\underline{v}^*)/q(v^*)$
1	15	0.06	0.0009	0.0005	1.8
1	10	0.1	0.0016	0.0010	1.600
1	1	0.3	0.0349	0.0321	1.087
10	1	0.76	0.0028	0.0022	1.273
50	1	0.94	0.0002	0.0001	2

scaled them to have the same maximum value. Figure 2.15 shows first component v_1 of the minimal proper and minimal suproper signals when $\omega = 0.0$ and $\omega = 0.4$. We observe that the optimal and suboptimal signals were essentially the same curve regardless of the amount of model uncertainty. However, when we look at the v_2

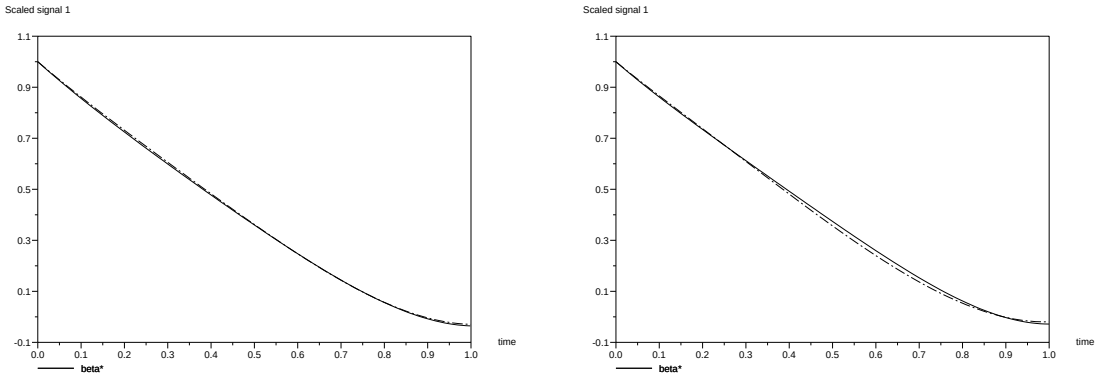

Figure 2.15: Scaled v_1 of minimal proper and minimal subproper signals for $\omega = 0.0$ (left), $\omega = 0.4$ (right).

component we see a very different situation as shown in Figure 2.16. The dashed lines substantially differs from the solid lines as we increase ω.

Tables 2.5–2.6 summarize the results for several values of $\omega = w_G = w_H$. Computations for optimal signal design were done using a 49 point β grid and, hence, the suboptimal algorithm takes only $\frac{1}{49}$ of the optimal solution computational time. We notice the size ratio between the suboptimal and optimal signals grows as we increase ω i.e. the amount of model uncertainty. When $\omega = 0.5$, for $\beta = 0.5$ there does not exist a signal v. The nonexistence of signal $\underline{v}^*$ is also confirmed in the Figure 2.17 by observing $\lambda_{0.5} = 0$ when $w_G = w_H = 0.5$.

The ratio of the norm of the minimal subproper and minimal proper is summarized in Table 2.7. Notice that the ratio ρ (or ϱ) is no longer bounded by $\sqrt{2}$ but, in fact,

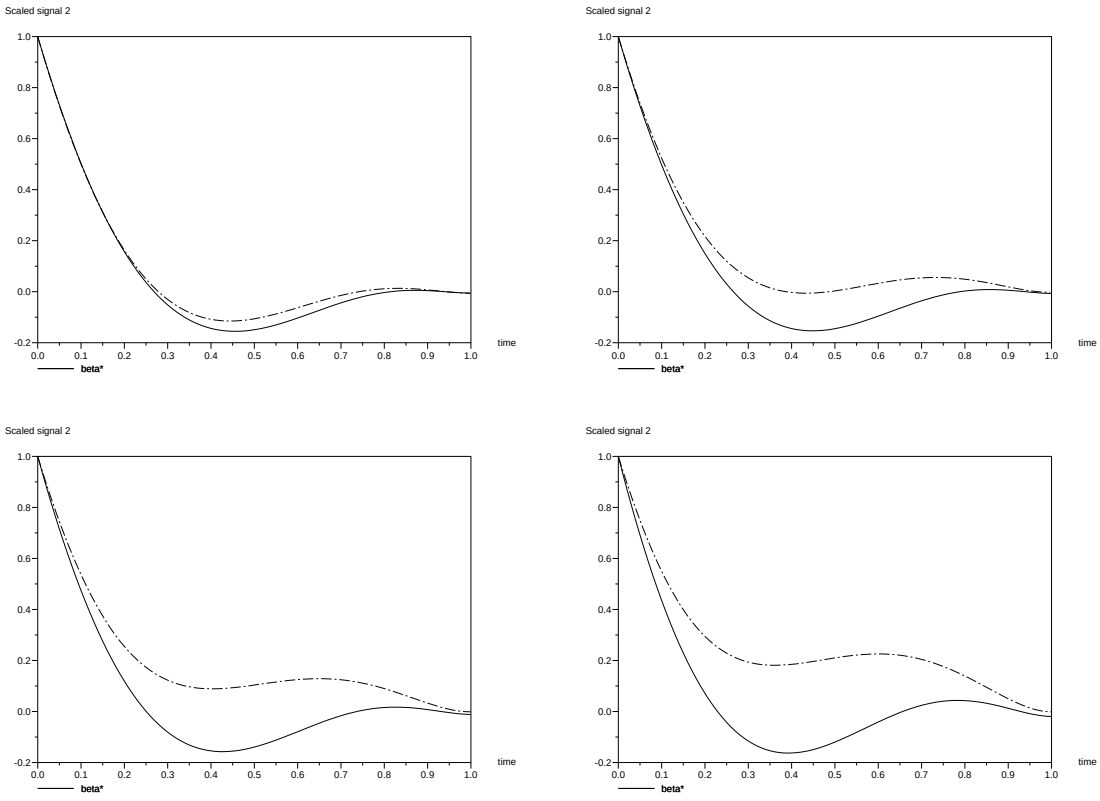

Figure 2.16: Scaled v_2 of minimal proper and minimal subproper signals for $\omega = 0.1$ (top left), 0.2, 0.3, 0.4 (bottom right)

goes to infinity. In theory $\lambda^* q(v*) = 1$ and hence $\varrho = \rho$. However, λ^* is found by the λ-iteration described earlier, and q is found by a numerical integration so this relationship is not exact.

2.2.3 Conclusion

In this section, we proposed a suboptimal signal design algorithm in order to speed up the auxiliary signal design process. Such an algorithm is useful when working with systems which require frequent fault detection because of their volatile dynamics.

Table 2.5: Optimal solution using β search

ω	β^*	λ^*	$\|\|v_1\|\|$	$\|\|v_2\|\|$	$q(v^*)$
0.0	0.1000	0.0016	542.3097	26.2765	621.1659
0.1	0.1000	0.0016	547.1717	25.8320	626.9430
0.2	0.0800	0.0016	561.2047	25.1881	639.6328
0.3	0.0800	0.0015	588.8456	22.3322	663.8517
0.4	0.0800	0.0014	631.2981	17.8443	701.7074
0.5	0.0600	0.0013	699.6586	13.1679	759.8189

Table 2.6: Suboptimal solution using $\beta = 0.5$

ω	λ_β	$\|\|v_1\|\|$	$\|\|v_2\|\|$	$q(\underline{v}^*)$
0.0	0.0010	889.9785	37.0422	1001.1970
0.1	0.0010	934.5567	39.2521	1061.8475
0.2	0.0008	1103.5285	47.2455	1282.0567
0.3	0.0006	1549.5774	67.8315	1837.5764
0.4	0.0002	3702.2719	154.1841	4283.9606
0.5	0			

We have compared the computation of a minimal proper auxilary signal v^* with the computation of a minimal subproper auxiliary signal $\underline{v}^*$ which is more like a standard robust control problem solution. It is seen that on some problems the suboptimal test signals are comparable to the optimal signals incurring only a small amount of cost increase, that is, being more "plant-hostile" in a limited amount. However, there were a number of situations where the subproper either failed to exist or was much larger in norm than the minimal proper test signal. Therefore, it is important to take account of the tradeoff between signal design speed and signal "plant-friendliness"

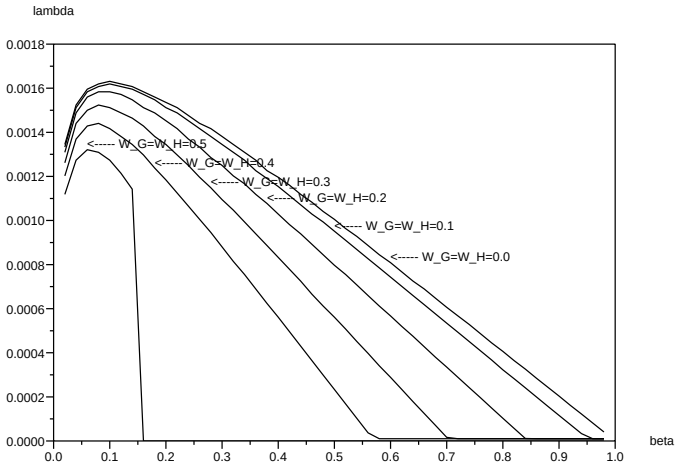

Figure 2.17: λ_β with $w_G = w_H$, $\theta_1 = 10$.

Table 2.7: Ratio of norms of $\underline{v}^*$ and v^*.

ω	$\varrho = \sqrt{\lambda^*/\lambda_{0.5}}$	$\rho = \sqrt{q(\underline{v}^*)/q(v^*)}$
0.0	1.2649111	1.269568
0.1	1.2649111	1.3014187
0.2	1.4142136	1.4157555
0.3	1.5811388	1.6637465
0.4	2.6457513	2.4708405
0.5	∞	*

when we consider to use the suboptimal algorithm.

Chapter 3

Optimal Digital Signal Design

With the rapid development and wide availability of microprocessors, many physical systems are designed to interact with digital computers for various purposes including fault detection. Also, often in practice, simple tests, such as turning on and off a gas valve to check leakage from a compressor, are all that is needed to check the condition of the system of interest. Like these cases, digital or piecewise constant signals are often highly desirable.

One way to generate piecewise constant (PWC) detection signals is to use sampled data system (SDS) models [41]. In the SDS models, the underlying system are continuous while the digital signal is computed based only on discrete-time sampled behavior information. Thus, intersample behaviors can be missed. If the sampling rate is relatively high, the system states will be revealed throughly enough in the SDS setting. The digital signal design algorithm in [41] is developed by equating the number of samples with the number of constant pieces of a detection signal. That is, when a simple digital signal, say a PWC function with two constant pieces, is asked for a fault detection test, [41] designs one based only on two sampled behavior observations possibly missing critical state changes occurred between the sampling times.

In this section, we develop an algorithm that designs digital fault detection signals based on continuously observed system behavior information. As in the previous chapter, we assume we have a continuous-time dynamic system whose behavior information is continuously available. In a sense, this work can also be interpreted as finding a PWC function, with a fixed number of constant pieces, that approximates the best to the optimal analog signal computed in Chapter 1.4.

In the rest of this section, the times where the PWC test signal v changes value will be often referred as the v-grid. We first develop an algorithm for computing minimal digital detection signals for a fixed v-grid. Then we discuss optimization over the v-grid of the signal. The v-grid optimization is a nonlinear inequality constrained optimization problem. We utilize an optimization solver from a software package to solve this nonlinear problem numerically. We present two computational experiments. Our primary interest here is in the case when there are a small number of constant pieces, perhaps only one or two. This is also the case where the digital signals computed by the algorithm to be discussed here outperform those computed by the sampled data approach and where the best way to approximate the continuous time signal is not obvious. Accordingly we include the sampled data test signal from [41] in our computational examples to show the differences. The sampled-data algorithm (SDS) is due to [41], however we provided the first implementation.

3.1 Digital Signal Design for Fixed v-grid

We follow the basic assumptions, system models and the detection test schemes used in the optimal analog signal design approach. The only difference is that the candidates of optimal proper signals are limited to piecewise constant functions with a specified number of constant pieces. In this section, it is further assumed that the length of each constant piece is given.

3.1.1 The digital detection signal design algorithm

The algorithm developed here is a variation of that of [41]. Here we must deal with a different type of noise on the output functions. Suppose the detection test is implemented during $[t_0, t_f]$. The linear model

$$\dot{x}_i = A_i x_i + B_i v + M_i \nu_i, \tag{3.1a}$$

$$E_i y = C_i x_i + D_i v + N_i \nu_i, \tag{3.1b}$$

represents normal ($i = 0$) and faulty systems ($i = 1$). Here x_i and y are physical state and measured information of its behavior (or simply output), respectively. A ν_i represents both additive and model uncertainty. The initial state $x_i(0)$ is also uncertain.

The input v is piecewise constant with $n+1$ pieces. Thus for $t_j, j = 0, 1, \cdots, n$,

$$v(t) = v_j \quad \text{if } t_j \leq t < t_{j+1} \tag{3.2}$$

where the v_j's are constant and $t_{n+1} = t_f$. The n-times t_j, $j = 0, 1, \cdots, n$ where v changes values will be called the v-grid and are denoted by $\mathbf{t}$. In the algorithms it is a column vector, but we will sometimes write it as a row vector to simplify the discussion and notation. System matrices have arbitrary but consistent dimensions; the only condition is that the N_i's have full row rank and E_i has full column rank.

The constraint (or uncertainty bound) on the initial condition and uncertainty variable is

$$\mathcal{S}_v^i(x_i(0), \nu_i, s) = x_i(t_0)^T P_i^{-1} x_i(t_0) + \int_{t_0}^{s} \nu_i^T J_i \nu_i \, dt < 1, \quad \forall t_0 \leq s \leq t_f. \tag{3.3}$$

The J_i's are diagonal matrices with 1 or -1 on their diagonal which are determined

by (2.6). When no model uncertainty is considered, $J_i = I$ and, hence, we need only consider $s = t_f$.

The assumption is that for fault detection, we have access to y, given a v, consistent with one of the models. The problem of designing a proper signal is equivalent to find v for which observation of y provides enough information to decide from which model y has been generated. That is, there exists no solution to (3.1), (3.1b), (3.2) and (3.3) for $i = 0$ and 1 simultaneously.

We consider a positive quadratic cost function on v of the form:

$$q(v) = \rho(t_f)^T W \rho(t_f) + \int_{t_0}^{t_f} \|v\|^2 + \rho^T U \rho \, dt, \tag{3.4a}$$

$$\dot{\rho} = F\rho + Gv, \quad \rho(t_0) = 0. \tag{3.4b}$$

where W, U are positive semi-definite matrices and F, G are chosen by design considerations.

Since the N_i's are full row rank, we have that for any piecewise constant functions v and L^2 functions y, there exist L^2 functions ν_i satisfying (3.1). Thus the non-existence of a solution to (3.1),(3.2) which satisfies (3.3) is equivalent to $\sigma(v, s) \geq 1$ for some s where

$$\sigma(v, s) = \inf_{\substack{\nu_0, \nu_1 \\ x_0, x_1}} \max\{\mathcal{S}_v^0(x_0(0), \nu_0, s), \mathcal{S}_v^1(x_1(0), \nu_1, s)\}$$

subject to (3.1),(3.2), for $i = 0, 1$. In [41] it is shown that while losing a certain type of allowed uncertainty variable, the infinite number of conditions in (3.3) can be replaced by the finite number of conditions

$$\mathcal{S}_v^i(x_i(0), \nu_i, t_k) < 1, \ k = 0, 1, \cdots, n+1.$$

Then

$$\begin{aligned} \sigma(v,t_k) &= \max_{\beta\in[0,1]} \phi_\beta(v,t_k), \text{ where} && \text{(3.5a)} \\ \phi_\beta(v,t_k) &= \inf_{\substack{\nu_0,\nu_1,y \\ x_0,x_1}} \beta\mathcal{S}_v^0(x_0(0),\nu_0,t_k) + (1-\beta)\mathcal{S}_v^1(x_1(0),\nu_1,t_k) && \text{(3.5b)} \\ & \quad \text{subject to (3.1),(3.2),} \end{aligned}$$

Thus for a given k, the optimal v is obtained by solving the following optimization problem:

$$\min_v q(v), \text{ subject to } \max_{\beta\in[0,1]} \phi_\beta(v,t_k) \geq 1. \tag{3.6}$$

This problem is in the same form as the problem solved to find optimal analog detection signals. The only difference is that the constraint needs to be satisfied for a finite number of times instead of for all the times between t_0 and t_f. This allows us to modify the key part of the optimal analog signal design algorithm. Both ϕ_β and q are quadratic functions of v since they are the solutions of a quadratic cost optimization problem subject to linear constraints. Let

$$\lambda^* = \max_{\substack{\mathbf{v}_k,\, k\leq n+1 \\ \beta\in[0,1]}} \frac{\phi_\beta(\mathbf{v}_k,k)}{q(\mathbf{v}_k)}. \tag{3.7}$$

Here

$$\mathbf{v}_k = \begin{pmatrix} v_0 \\ \vdots \\ v_{k-1} \end{pmatrix} \tag{3.8}$$

and the corresponding auxiliary signal $v(t)$ is defined in (3.2). When v is replaced by $\mathbf{v_k}$ in a formula it is understood that (3.2) holds. Suppose $\mathbf{v}_{k^*}^*$ realizes the max.

Then

$$\mathbf{v}^* = \frac{1}{\sqrt{\lambda^* q(\mathbf{v}^*_{k^*})}} \mathbf{v}^*_{k^*} \tag{3.9}$$

defines a PWC proper auxiliary signal which is minimal in the norm (3.4a) over the test period and with the given v-grid.

First, we reduce the problem (3.5b) to a standard optimal control problem. We can simplify (3.1b), $i = 0, 1$, to

$$(F_0E_0 - F_1E_1)y = F_0C_0x_0 - F_1C_1x_1 + (F_0D_0 - F_1D_1)v + F_0N_0\nu_0 - F_1N_1\nu_1 \tag{3.10}$$

by combining the two system equations and applying $\begin{pmatrix} F_0 & -F_1 \end{pmatrix}$ on both sides of the combined model. In general, F_i is a largest full row rank matrix such that $F_iE_i = 0$. When there is not model uncertainty, we have $E_0 = E_1 = I$ and set $F_0 = F_1 = I$. In any case, the left hand side of the equation in (3.10) is equal to 0. For simplicity, we reuse the following notations

$$C_i \triangleq F_iC_i, \quad D_i \triangleq F_iD_i, \quad N_i \triangleq F_iN_i. \tag{3.11}$$

Note that a multiplication of two full row rank matrices produces another full row rank matrix. Since the new N_i is a full row rank matrix, a QR decomposition produces an upper triangular matrix R and an orthogonal matrix Q so that $N_i^T = QR$ [16]. Let $R^T = \begin{pmatrix} \bar{N}_i & 0 \end{pmatrix}$ where $\bar{N}_i$ is non-singular. Then, we obtain the following change of coordinates in N_i and ν_i:

$$N_i\nu_i = R^TQ^T\nu_i = \begin{pmatrix} \bar{N}_i & 0 \end{pmatrix} \begin{pmatrix} \bar{\nu}_i \\ \widetilde{\nu}_i \end{pmatrix}. \tag{3.12}$$

Similarly, $M_i\nu_i = \begin{pmatrix} \bar{M}_i & \widetilde{M}_i \end{pmatrix} = \begin{pmatrix} \bar{\nu}_i \\ \widetilde{\nu}_i \end{pmatrix}$. Then we can use the following notation

$$A = \begin{pmatrix} A_0 - \bar{M}_0\bar{N}_0^{-1}C_0 & \bar{M}_0\bar{N}_0^{-1}C_1 \\ 0 & A_1 \end{pmatrix},\ B = \begin{pmatrix} B_0 - \bar{M}_0\bar{N}_0^{-1}(D_0 - D_1) \\ B_1 \end{pmatrix},$$

$$M = \begin{pmatrix} \widetilde{M}_0 & \bar{M}_0\bar{N}_0^{-1}\bar{N}_1 & 0 \\ 0 & \bar{M}_1 & \widetilde{M}_1 \end{pmatrix}, x = \begin{pmatrix} x_0 \\ x_1 \end{pmatrix},\ \nu = \begin{pmatrix} \widetilde{\nu}_0 \\ \bar{\nu}_1 \\ \widetilde{\nu}_1 \end{pmatrix},$$

$$P_\beta^{-1} = \begin{pmatrix} \beta P_0^{-1} & 0 \\ 0 & (1-\beta)P_1^{-1} \end{pmatrix},\ \Omega = \bar{N}_0^{-T} J_0 \bar{N}_0^{-1},$$

$$Q = 2\beta \begin{pmatrix} C_0^T\Omega C_0 & -C_0^T\Omega C_1 \\ -C_1^T\Omega C_0 & C_1^T\Omega C_1 \end{pmatrix},\ R = 2 \begin{pmatrix} \beta J_0 & 0 & 0 \\ 0 & \beta\bar{N}_1^T\Omega\bar{N}_1 + (1-\beta)J_1 & 0 \\ 0 & 0 & (1-\beta)J_1 \end{pmatrix},$$

$$S = 2\beta(D_0 - D_1)^T\Omega(D_0 - D_1),\ U = 2\beta \begin{pmatrix} 0 & -C_0^T\Omega\bar{N}_1 & 0 \\ 0 & C_1^T\Omega\bar{N}_1 & 0 \end{pmatrix},$$

$$V = 2\beta \begin{pmatrix} C_0^T\Omega(D_0 - D_1) \\ -C_1^T\Omega(D_0 - D_1) \end{pmatrix}, W = 2\beta \begin{pmatrix} 0 \\ \bar{N}_1^T\Omega(D_1 - D_0) \\ 0 \end{pmatrix}, \quad (3.13)$$

to express the optimization problem $\phi_\beta(\mathbf{v}_k, t_k)$ as follows

$$\phi_\beta(\mathbf{v}_k, t_k) = \inf_{\nu,x} x_0^T P_\beta^{-1} x_0 + \int_{t_0}^{t_k} \frac{1}{2} x^T Q x + \frac{1}{2}\nu^T R \nu + \frac{1}{2}\mathbf{v}_k^T S \mathbf{v}_k + x^T U \nu + x^T V \mathbf{v}_k + \nu^T W \mathbf{v}_k \; ds, \quad (3.14a)$$

$$\text{subject to } \dot{x} = Ax + B\mathbf{v}_k + M\nu \quad (3.14b)$$

We assume that Q and S are positive semi-definite and R is positive definite. Unlike the case where we seek continuously varying signals, this problem can be solved in two steps which are discussed in [41].

The optimization problem (3.14) can be expressed as follows:

$$\phi_\beta(\mathbf{v}_k, k) = \min_{x_0,\cdots,x_k} V_k(x_0, v_0, \ldots, v_{k-1}, x_k) \quad (3.15)$$

where $V_k(x_0, v_0, \ldots, v_{k-1}, x_k)$ is

$$\min_\nu x(t_0)^T P_\beta^{-1} x(t_0) + \int_{t_0}^{t_k} \frac{1}{2}(x^T Q x + \nu^T R \nu + \mathbf{v}_k^T S \mathbf{v}_k) + x^T U \nu + x^T V \mathbf{v}_k + \nu^T W \mathbf{v}_k \; ds, \quad (3.16)$$

subject to (3.14b) and $x_j = x(t_j), \;\; 0 \le j \le k$.

The first step starts with conditioning on $x(t_j)$ which allows us to solve the problem (3.16) separately over each subinterval $[t_j, t_{j+1}]$. Each of these optimization problems are of the form:

$$\psi_j(x_{j+1}, x_j, v_j) = \min_\nu \int_{t_j}^{t_{j+1}} \frac{1}{2} x^T Q x + \frac{1}{2}\nu^T R \nu + \frac{1}{2} v_j^T S v_j + x^T U \nu + x^T V v_j + \nu^T W v_j \; ds \quad (3.17)$$

subject to (3.14b), $x(t_j) = x_j$, and $x(t_{j+1}) = x_{j+1}$. The solution to this optimal control problem can be computed by solving its necessary conditions. Suppose the optimization problem (3.16) has a solution and consider the two-point boundary value linear system

$$\begin{pmatrix} \dot{x} \\ \dot{\lambda} \end{pmatrix} = \mathfrak{M} \begin{pmatrix} x \\ \lambda \end{pmatrix} - \begin{pmatrix} B - MR^{-1}W \\ UR^{-1}W \end{pmatrix} v_j, \tag{3.18a}$$

with boundary conditions $x(t_j) = x_j$ and $x(t_{j+1}) = x_{j+1}$,

$$\text{where } \mathfrak{M} = \begin{pmatrix} A - MR^{-1}U^T & -MR^{-1}M^T \\ UR^{-1}U^T - Q & UR^{-1}M^T - A^T \end{pmatrix}. \tag{3.18b}$$

Then, the optimal ν which gives the min in (3.17) satisfies

$$\nu = -R^{-1}(U^T x + W v_j + M^T \lambda). \tag{3.19}$$

Noting that $v(t) = v_j$ is constant over $[t_j, t_{j+1}]$, we can explicitly solve the boundary value problem (3.18). We show that there exists matrices $\mathcal{J}_\beta(j)$ such that

$$V_k(x_0, v_0, \cdots, v_{k-1}, x_k) = x_0^T P_\beta^{-1} x_0 + \sum_{j=0}^{k-1} \begin{pmatrix} x_{j+1}^T & x_j^T & v_j^T \end{pmatrix} \mathcal{J}_\beta(j) \begin{pmatrix} x_{j+1} \\ x_j \\ v_j \end{pmatrix}. \tag{3.20}$$

If the system matrices are time-invariant and the t_j's form a uniform grid (i.e., $t_{j+1} - t_j$ is constant), then the $\mathcal{J}_\beta(j)$'s do not depend on j and are written $\mathcal{J}_\beta$. Clearly all we

need to show is that

$$\psi_j(x_{j+1}, x_j, v_j) = \begin{pmatrix} x_{j+1} \\ x_j \\ v_j \end{pmatrix}^T \mathcal{J}_\beta(j) \begin{pmatrix} x_{j+1} \\ x_j \\ v_j \end{pmatrix}. \tag{3.21}$$

This is done constructively. Let

$$\xi(s) = \begin{pmatrix} x(t_j + s) \\ \lambda(t_j + s) \end{pmatrix}. \tag{3.22}$$

Then it is straightforward to show that $\xi(s) = \Psi(s)\xi(0) + \Phi(s)v_j$, where Ψ and Φ satisfy

$$\dot{\Psi} = \mathfrak{M}\,\Psi, \quad \Psi(0) = I, \tag{3.23}$$

$$\dot{\Phi} = \mathfrak{M}\,\Phi + \begin{pmatrix} B - MR^{-1}W \\ UR^{-1}W \end{pmatrix}, \quad \Phi(0) = 0. \tag{3.24}$$

The coefficient matrices in (3.23) and (3.24) are evaluated at $s+t_j$. Let $\tau_j = t_{j+1} - t_j$. Then we have

$$\left(\begin{array}{cc|cc} 0 & 0 & I & 0 \\ I & 0 & 0 & 0 \\ \hline -\Psi(\tau_j) & & I & \end{array}\right) \begin{pmatrix} \xi(0) \\ \xi(\tau_j) \end{pmatrix} = \left(\begin{array}{c} x_{j+1} \\ x_j \\ \hline \Phi(\tau_j)v_j \end{array}\right). \tag{3.25}$$

We thus obtain

$$\xi(s) = \Xi(\tau_j, s) \begin{pmatrix} x_{j+1} \\ x_j \\ v_j \end{pmatrix} \tag{3.26}$$

where

$$\Xi(\tau_j, s) = \Psi(s)\Pi(\tau_j) + \begin{pmatrix} 0 & 0 & \Phi(s) \end{pmatrix} \tag{3.27}$$

and

$$\Pi(\tau_j) = \begin{pmatrix} I & 0 \end{pmatrix} \left(\begin{array}{cc|cc} 0 & 0 & I & 0 \\ I & 0 & 0 & 0 \\ \hline -\Psi(\tau_j) & & I & \end{array}\right)^{-1} \begin{pmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & \Phi(\tau_j) \end{pmatrix}. \tag{3.28}$$

The inverse in (3.28) exists for τ_j thanks to the well-posedness assumption which in fact guarantees the invertibility for all $\tau \leq \tau_j$. Using (3.19) and the definition of ξ (3.22) we can rewrite $\psi_j(x_{j+1}, x_j, v_j)$ in (3.17) as follows

$$\frac{1}{2}\int_0^{\tau_j} \begin{pmatrix} \xi(s) \\ v_j \end{pmatrix}^T \begin{pmatrix} \mathfrak{J}_1 & \mathfrak{J}_2 \\ \mathfrak{J}_2^T & \mathfrak{J}_3 \end{pmatrix} \begin{pmatrix} \xi(s) \\ v_j \end{pmatrix} ds, \tag{3.29a}$$

where

$$\mathfrak{J}_1 = \begin{pmatrix} Q - UR^{-1}U^T & 0 \\ 0 & MR^{-1}M^T \end{pmatrix}, \mathfrak{J}_2 = \begin{pmatrix} V - UR^{-1}W \\ 0 \end{pmatrix}, \mathfrak{J}_3 = \begin{pmatrix} S - W^T R^{-1} W \end{pmatrix}. \tag{3.29b}$$

Using (3.26), it is easy to show that ψ_j can be expressed as in (3.21) with $\mathcal{J}_\beta(j)$ equal

to

$$\frac{1}{2}\int_0^{\tau_j} \left(\begin{array}{cc|c} \Xi(\tau_j,s) & & \\ & & \\ 0 & 0 & I \end{array}\right)^T \begin{pmatrix} \mathfrak{J}_1 & \mathfrak{J}_2 \\ \mathfrak{J}_2^T & \mathfrak{J}_3 \end{pmatrix} \left(\begin{array}{cc|c} \Xi(\tau_j,s) & & \\ & & \\ 0 & 0 & I \end{array}\right) ds. \tag{3.30}$$

All matrices in (3.30) except for Π are evaluated at $s + t_j$.

Before continuing, it should be noted that $\int_0^{\tau_j} \Theta(s)ds$ can be evaluated by numerically integrating $\dot{Z} = \Theta, Z(0) = 0$. Thus (3.30) can be evaluated by a numerical integration of a differential equation using the integrand of (3.30) coupled with the differential equations (3.23), (3.24). We follow this approach in our codes and thus avoid direct computation of the matrix exponential [35].

Now that we have the solution to the inner optimization problem, the second step is to solve

$$\phi_\beta(\mathbf{v}_k, t_k) = \min_{x_0,\cdots,x_k} \left\{ x_0^T P_\beta^{-1} x_0 + \sum_{j=0}^{k-1} \begin{pmatrix} x_{j+1}^T & x_j^T & v_j^T \end{pmatrix} \mathcal{J}_\beta(j) \begin{pmatrix} x_{j+1} \\ x_j \\ v_j \end{pmatrix} \right\}. \tag{3.31}$$

We can express this optimization problem as a large static optimization problem

$$\phi_\beta(\mathbf{v}_k, t_k) = \min_{\mathbf{x}_k} \begin{pmatrix} \mathbf{x}_k^T & \mathbf{v}_k^T \end{pmatrix} \begin{pmatrix} \mathcal{X}_\beta(k) & \mathcal{Y}_\beta(k) \\ \mathcal{Y}_\beta(k)^T & \mathcal{Z}_\beta(k) \end{pmatrix} \begin{pmatrix} \mathbf{x}_k \\ \mathbf{v}_k \end{pmatrix}, \tag{3.32}$$

where

$$\mathcal{X}_\beta(k) = \begin{pmatrix} X_\beta(0) & J_4^T(0) & & & \\ J_4(0) & X_\beta(1) & & & \\ & & \ddots & & \\ & & & X_\beta(k-1) & J_4^T(k-1) \\ & & & J_4(k-1) & X_\beta(k) \end{pmatrix} \tag{3.33}$$

with

$$X_\beta(0) = P_\beta^{-1} + J_2(0) \tag{3.34}$$

$$X_\beta(j) = J_1(j-1) + J_2(j) \text{ for } j = 1, \ldots, k-1 \tag{3.35}$$

$$X_\beta(k) = J_1(k-1) \tag{3.36}$$

and

$$\begin{pmatrix} J_1(j) & J_4(j) & J_5(j) \\ J_4^T(j) & J_2(j) & J_6(j) \\ J_5^T(j) & J_6^T(j) & J_3(j) \end{pmatrix} = \mathcal{J}_\beta(j) \text{ for } j = 0, \ldots, k-1 \ . \tag{3.37}$$

Also

$$\mathcal{Y}_\beta(k) = \begin{pmatrix} J_6(0) & & & \\ J_5(0) & J_6(1) & & \\ & J_5(1) & \ldots & \\ & & \ldots & J_6(k-1) \\ & & & J_5(k-1) \end{pmatrix} \tag{3.38}$$

and

$$\mathcal{Z}_\beta(k) = \mathrm{Diag}(J_3(0), J_3(1), \cdots, J_3(k-1)). \tag{3.39}$$

Assuming $\mathcal{X}_\beta(k) > 0$, we have the minimum in (3.32) exists and the solution is given in terms of the Schur's complement of the matrix defining the quadratic form in (3.32):

$$\phi_\beta(\mathbf{v}_k, t_k) = \mathbf{v}_k^T(\mathcal{Z}_\beta(k) - \mathcal{Y}_\beta(k)^T\mathcal{X}_\beta(k)^{-1}\mathcal{Y}_\beta(k))\mathbf{v}_k. \tag{3.40}$$

One computational problem is that $\mathcal{X}_\beta(k)$ can be very large making the direct construction of the Schur's complement impractical. Fortunately, $\mathcal{X}_\beta(k)$ has a band structure which can be used to recursively test its positivity and construct the Schur's complement of the large matrix. It can be shown that the matrix $\mathcal{X}_\beta(k)$ is positive definite if and only if $\Lambda_\beta(j)$, for $j = 0, \ldots, k-1$, is positive definite where

$$\Lambda_\beta(j+1) = X_\beta(j+1) - J_4(j)\Lambda_\beta(j)^{-1}J_4^T(j) \tag{3.41}$$

with $\Lambda_\beta(0) = X_\beta(0)$, and the $X_\beta(j)$'s defined in (3.34)–(3.36).

If $\mathcal{X}_\beta(k)$ is not positive definite for any β, then there exists no proper auxiliary signal of length k because ϕ_β is $-\infty$ and cannot satisfy the constraint $\sigma(v, t_k) \geq 1$.

Once we know that the minimization problem has a finite solution, we can compute it recursively. Let

$$\Delta_\beta(k) = \mathcal{Z}_\beta(k) - \mathcal{Y}_\beta(k)^T\mathcal{X}_\beta(k)^{-1}\mathcal{Y}_\beta(k) \tag{3.42}$$

where $\mathcal{X}_\beta(k)$, $\mathcal{Y}_\beta(k)$ and $\mathcal{Z}_\beta(k)$ are defined in (3.33), (3.38) and (3.39). Then the symmetric matrix $\Delta_\beta(k)$ is obtained from the recursive formulae

$$\Delta_\beta(j+1) = \begin{pmatrix} \Delta_\beta(j) & 0 \\ 0 & J_3(j) \end{pmatrix} - \Gamma(j+1)^T\Lambda_\beta(j+1)^{-1}\Gamma(j+1) - \begin{pmatrix} 0 & \Gamma(j)^T\Lambda_\beta^{-1}(j)J_6(j) \\ J_6^T(j)\Lambda_\beta^{-1}(j)\Gamma(j) & J_6^T(j)\Lambda_\beta^{-1}(j)J_6(j) \end{pmatrix},$$

$$\Gamma(j+1) \;=\; \begin{pmatrix} 0 & J_5 \end{pmatrix} - J_4 \Lambda_\beta(j)^{-1} \begin{pmatrix} \Gamma(j) & J_6 \end{pmatrix}$$

with $\Delta(0) = [\,]$ and $\Gamma(0) = [\,]$.

Since q is a positive quadratic function, for some positive-definite matrix $\mathcal{Q}(k)$, we have $q(\mathbf{v}_k) = \mathbf{v}_k^T \mathcal{Q}(k) \mathbf{v}_k$. For example, if q is the L^2 norm of $\mathbf{v_k}$, then $\mathcal{Q}(k) = \text{Diag}((t_1 - t_0)I, \ldots, (t_k - t_{k-1})I)$. Let

$$\lambda_\beta(k) = \max_{\mathbf{v}_k} \frac{\phi_\beta(\mathbf{v}_k, k)}{q(\mathbf{v}_k)}. \tag{3.43}$$

Then $\lambda_\beta(k)$ is the largest eigenvalue of $\mathcal{Q}(k)^{-\frac{1}{2}} \Delta_\beta(k) \mathcal{Q}(k)^{-\frac{1}{2}}$. It is also the largest value of λ for which

$$\Delta_\beta(k) - \lambda \mathcal{Q}(k) \tag{3.44}$$

is singular. In general, it is preferable not to perform the inversion, we use the latter definition of $\lambda_\beta(k)$. Thus the computation of $\lambda_\beta(k)$ amounts to solving a generalized eigenvalue problem for which reliable computer program exists. Note also that $\Delta_\beta(k)$ is constructed recursively, so $\lambda_\beta(k)$ is obtained by a recursive formula.

The value of λ^* defined in (3.7) can now be computed as follows

$$\lambda^* = \max_{\beta, k \le n+1} \lambda_\beta(k). \tag{3.45}$$

The max is obtained by a simple search over a β-grid defined on the interval $(0, 1)$. Let $\beta = \beta^*$ and $k = k^*$ yield the maximum in (3.45). Then a minimal proper PWC auxiliary signal for a given v-grid is computed by (3.9) where $\mathbf{v}_{k^*}^*$ is any vector in the null-space of $\Delta_{\beta^*}(k^*) - \lambda^* \mathcal{Q}(k^*)$.

3.1.2 Numerical issues

The digital signal design algorithm is implemented in Scilab environment. The major part of the algorithm is composed of manipulating matrices and solving differential equations and an eigenvalue problem. Scilab provides functions each of which condenses many of these computations into a line of code. In particular, `ode` and `spec` solve, respectively, the ordinary differential equations (ODEs) and the eigenvalue or the generalized eigenvalue problem, .

Currently, ODEs in the algorithm are chosen to be solved by *lsoda* of package ODEPACK [19]. It uses nonstiff predictor-corrector Adams method initially and dynamically monitors its computation in order to decide if stiff Backward Differentiation Formula (BDF) method is needed. See [45] for more information on these methods. Depending on the specific problem, various types of ODE solver can be utilized. Also, `%ODEOPTIONS` may be used to have required accuracy and stability in the selected ODE solver [11] .

The dimension of the digital signal and the required number of digital pieces are directly related to the size of the matrices which define the eigenvalue problem in the algorithm. Designing a high dimensional digital signal with many digital pieces can demand to increase the size of variable storage. This is done via `stacksize(`n`)` where n is the required storage size.

3.2 Optimization of v-grid of a Digital Signal

The previous section showed how to construct a minimal digital test signal for guaranteed fault detection for a given v-grid. In this section, we discuss finding an even smaller signal by minimizing over the v-grid when there does not exist model uncertainty. In one sense this is straightforward since we call an existing optimizer to do so.

But as is often the case with practical optimization problems a number of technical questions arise including the possibility of local minimums, convergence of iterations, choice of methods, etc. We will address some of those issues here.

3.2.1 Problem setup

For each fixed n, let

$$\Gamma_n = \left\{ \mathbf{t} = \begin{pmatrix} t_1 & \dots & t_n \end{pmatrix}^T \;\middle|\; 0 < t_1 < \ldots < t_n < t_f \right\}. \tag{3.46}$$

That is Γ_n is the set of all possible v-grids for a PWC signal with $n+1$ constant pieces. Then the problem of finding optimal v-grid for a minimal proper PWC signal can be expressed as

$$\min_{\mathbf{t}\in\Gamma_n} \Big\{ \;\; \min_{v} q(v) \tag{3.47a}$$

$$\text{subject to} \quad v_j = v(t) \text{ if } t_j \le t < t_{j+1},\; j = \{0,\ldots,n\},\; t_0 = 0,\; t_{n+1} = t_f, \tag{3.47b}$$

$$\text{such that} \quad \max\Big(x_0(0)^T P_0 x_0(0) + ||\nu_0||^2_{L^2}\,,\; x_1(0)^T P_1 x_1(0) + ||\nu_1||^2_{L^2} \Big) \ge 1 \tag{3.47c}$$

$$\text{for all } x_i, \nu_i \text{ that satisfy}$$

$$\dot{x}_0 = A_0 x_0 + B_0 v + M_0 \nu_0 \tag{3.47d}$$

$$\dot{x}_1 = A_1 x_1 + B_1 v + M_1 \nu_1 \tag{3.47e}$$

$$y = C_0 x_0 + D_0 v + N_0 \nu_0 \tag{3.47f}$$

$$y = C_1 x_1 + D_1 v + N_1 \nu_1 \;\Big\}. \tag{3.47g}$$

Here $\|g\|^2_{L_2} = \int_{t_0}^{t_f} |g(t)|^2 dt$ and $|\cdot|$ denotes the usual Euclidian vector norm. We assume that for a given v-grid $\mathbf{t} \in \Gamma_n$ a nonzero q-minimal proper PWC test signal

$v^*(t)$ exists. The domain Γ_n is a bounded set. The set

$$\overline{\Gamma}_n = \Big\{ \mathbf{t} = \begin{pmatrix} t_1 & \dots & t_n \end{pmatrix}^T \Big| \; 0 \le t_1 \le \ldots \le t_n \le t_f \Big\}, \tag{3.48}$$

is the closure of Γ_n and is compact. In what follows if we wish to emphasize that a quantity such as v^* or λ depends on $\mathbf{t}$ we write $v^*(\mathbf{t})$ or $\lambda(\mathbf{t})$.

LEMMA 3.1. *There always exists a minimizer* $\mathbf{t}^*$ *for the optimization problem (3.47).*

Proof Consider an optimization problem

$$\min_{\mathbf{t} \in \overline{\Gamma}_n} \Big\{ \min_v q(v) \text{ subject to } (3.47b) - (3.47g) \Big\}. \tag{3.49}$$

The problem (3.49) always has a minimizer because of the compactness of $\overline{\Gamma}_n$ and the continuity of q in the variables $\mathbf{t}, v$. Suppose its minimizer(s) $\bar{\mathbf{t}}^*$ can be found only in $\overline{\Gamma}_n \setminus \Gamma_n$. This says the q-minimizing v is, in fact, a PWC function with less than $n+1$ constant pieces. But given any PWC v which is q-minimal and proper, we can always reduce the value of $q(v)$ by subdividing an interval. Therefore, there always exists a minimizer in Γ_n. ■

Having existence of a minimizer, it is further assumed that there is a (local) minimizer $\mathbf{t}^*$ of (3.47) for which v^* on that $\mathbf{t}^*$ is unique, up to the sign of v^*. This is not a restrictive assumption. Non-unique optimal solutions v^* are present when two output sets $\mathcal{Y}_i(v^*)$, $i = 0, 1$ are not strictly convex so that the closures of $\mathcal{Y}_i(v^*)$ intersect at a parallel, flat side of their borders. This geometry arises when both A_i matrices share a common eigenvalue, eigenvector pair. But this occurrence can be prevented since a little perturbation in an element of a matrix changes all of the eigenvalues of that matrix. This implies that it takes a very special system to get

both output sets which are not strictly convex and thus allow for a local minimizer v^* which is not isolated. The computational examples to follow will show that there can be multiple local minima.

Let $f(\mathbf{t})$ be the q-norm of a minimal proper PWC signal for the given $\mathbf{t} \in \Gamma_n$. Then, the problem (3.47) is rewritten as follows:

$$\min_{\mathbf{t}} f(\mathbf{t}) \quad \text{subject to} \quad A\mathbf{t} < b \tag{3.50}$$

where

$$A = \begin{pmatrix} 1 & & \\ -1 & \ddots & \\ & \ddots & 1 \\ & & -1 \end{pmatrix}, \; b = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ -t_f \end{pmatrix}.$$

This is a nonlinear optimization problem with inequality constraints.

3.2.2 Objective function $f(\mathbf{t})$

$f(\mathbf{t})$ is a nonlinear function of v-grid $\mathbf{t}$. In this subsection, we gather the function's natural characteristics as well as the properties that are gained by the two assumptions made in Section 3.2.1.

The objective function $f(\mathbf{t})$ is obtained after solving a minimization problem with a set of constraints (3.47b)-(3.47g). In Section 3.1 it was known that a minimal proper PWC signal $v^*(\mathbf{t})$ for a given $\mathbf{t}$ is given by an eigenvector of the matrix $\mathcal{Q}^{-1/2}\Delta\mathcal{Q}^{-1/2}$ corresponding to λ^*. Also, note that (3.9) indicates that

$$\frac{1}{\lambda^*(\mathbf{t})} = q(v^*) = f(\mathbf{t}). \tag{3.51}$$

Note that the assumption of the existence of a non-zero proper PWC signal implies that $\frac{1}{q(v^*(\mathbf{t}))}$ is well-defined for every $\mathbf{t} \in \Gamma_n$. Thus, the original problem of minimizing $f(\mathbf{t})$ is the same as that of maximizing $\lambda^*(\mathbf{t})$, which maps $\mathbf{t} \in \Gamma_n$ to a real value in the interval $[0, \infty)$, i.e.

$$\min_{\mathbf{t}\in\Gamma_n} f(\mathbf{t}) = \max_{\mathbf{t}\in\Gamma_n} \frac{1}{q(v^*(\mathbf{t}))} = \max_{\mathbf{t}\in\Gamma_n} \lambda^*(\mathbf{t}). \tag{3.52}$$

This equivalence between the two optimization problems provides us the chance to understand $f(\mathbf{t})$ through $\lambda^*(\mathbf{t})$. As seen in (3.44) and (3.45), we have that

$$\lambda^*(\mathbf{t}) = \max_{\beta} \sigma_{\max}(\mathcal{M}(\beta, \mathbf{t})) \quad \text{where } \mathcal{M} = \mathcal{Q}^{-1/2}\Delta\mathcal{Q}^{-1/2}. \tag{3.53}$$

Here, $\sigma_{\max}(\mathcal{M})$ implies the largest eigenvalue of the symmetric matrix $\mathcal{M}$. Note that $\lambda^*(\mathbf{t})$ can be written as $\sigma_{max}(\mathcal{M}(\beta^*(\mathbf{t}), \mathbf{t})$. The existence of β^* for every $\mathbf{t} \in \Gamma_n$ is guaranteed by the existence of nonzero $v^*(\mathbf{t})$.

It is known that the eigenvalues are continuous in several parameters if the matrix operator is continuous in a domain [27]. Since the maximum of a finite number of continuous functions is continuous, the largest eigenvalue of continuous eigenvalues is also a continuous function. Note also that if a function that is continuous in multiple variables, say x_1 and x_2, is optimized over a variable x_1 on a compact set, the resulting function is continuous in the rest of variable x_2. Thus, if $\mathcal{M}(\beta, \mathbf{t})$ is continuous in β and $\mathbf{t}$, we can obtain continuity for $\lambda^*(\mathbf{t})$.

LEMMA 3.2. *$\lambda^*(\mathbf{t})$ is continuous in $\mathbf{t} \in \Gamma_n$.*

Proof We prove that $\mathcal{M}(\beta, \mathbf{t})$ is continuous in the domain. Note that $\mathcal{M} = \mathcal{Q}^{-1/2}\Delta\mathcal{Q}^{-1/2}$. $\mathcal{Q}$ is the coefficient matrix of v-quadratic term of $q(v)$ in (3.4). Since the product of continuous functions is continuous and $\mathcal{Q}$ is continuous in $\mathbf{t} \in \Gamma_n$ and β, it suffices to prove continuity of Δ in β and $\mathbf{t}$. (3.42) and (3.33)-(3.39) show that

continuity of $\mathcal{J}_\beta(j), j = 1, \ldots, n$ is the necessary condition for Δ to be a continuous function. In (3.30), $\mathcal{J}_\beta(j)$ is defined as follows:

$$\mathcal{J}_\beta(j) = \frac{1}{2}\int_0^{t_{j+1}-t_j} \left(\begin{array}{cc|c} & & \\ \multicolumn{2}{c|}{\Xi(\tau_j, s)} & \\ & & \\ \hline 0 & 0 & I \end{array}\right)^T \begin{pmatrix} \mathfrak{J}_1 & \mathfrak{J}_2 \\ \mathfrak{J}_2^T & \mathfrak{J}_3 \end{pmatrix} \left(\begin{array}{cc|c} & & \\ \multicolumn{2}{c|}{\Xi(\tau_j, s)} & \\ & & \\ \hline 0 & 0 & I \end{array}\right) ds.$$

The $\mathfrak{J}_i$, $i = 1, 2, 3$ are defined in (3.29b) and (3.13). Each of them is continuous in $\beta \in [0, 1]$ and with respect to $\mathbf{t}$. Note that when R has a zero square matrix on its diagonal, say $R = \begin{pmatrix} \bar{R} & 0 \\ 0 & 0 \end{pmatrix}$, then we use $\begin{pmatrix} \bar{R}^{-1} & 0 \\ 0 & 0 \end{pmatrix}$ for R^{-1}. In addition, $\Xi(\tau_j, s)$, which is defined in (3.27) with (3.23),(3.24) and (3.28), is also continuous in t_j and t_{j+1}. Integrand and endpoints are continuous in β and t_j, t_{j+1}, hence, $\mathcal{J}_\beta(j), j = 1, \ldots, n$ is continuous in $\beta \in [0, 1]$ and $\mathbf{t} \in \Gamma_n$. This proves that $\mathcal{M}(\beta, \mathbf{t})$ is a continuous function in its domain. Continuity of $\lambda^*(t)$ follows. ■

In the above proof, it is not hard to see that $\mathcal{M}$ is smooth in $\beta \in (0, 1), \mathbf{t} \in \Gamma_n$. Notice that $\mathcal{Q}$ is smooth in β and $\mathbf{t}$ and given a level of smoothness, that will automatically also be true for products. Also note that given a level of smoothness of a matrix valued function $\mathcal{X}$, if $det(\mathcal{X}) \neq 0$, then the same level of smoothness will hold for $\mathcal{X}^{-1}$.

The partial differentiability of $\mathcal{M}(\beta, \mathbf{t})$ implies the same property for its own repeated eigenvalues. However, when the eigenvalues are ordered, say

$$\mu_1(\beta, \mathbf{t}) \leq \cdots \leq \mu_N(\beta, \mathbf{t}), \tag{3.54}$$

the $\mu_i(\beta, \mathbf{t})$ are not necessarily differentiable since the eigenvalues may not be differentiable quantities at points where they coalesce. However, in any case they are,

at least, piecewise smooth since there are a finite number of crossing points in any bounded domain. It is expected that the largest eigenvalue, $\lambda^*(\mathbf{t})$, is also at least a piecewise smooth function. In the best scenario where all eigenvalues are distinct on the entire domain Γ_n, $\lambda^*(\mathbf{t})$ will be infinitely differentiable [27]. In our study, by the second assumption we have

LEMMA 3.3. *There is a maximizer* $\mathbf{t}^*$ *such that* $\lambda^*(\mathbf{t})$*, and hence* $f(\mathbf{t})$*, is locally smooth near* $\mathbf{t}^*$*.*

Proof In the second assumption, we suppose the existence of a maximizer $\mathbf{t}^*$ such that the minimal proper signal $v^*(\mathbf{t}^*)$ is unique, up to the sign of $v^*(\mathbf{t}^*)$. Note that $v^*(\mathbf{t})$ is an eigenvector of a matrix corresponding to the eigenvalue $\lambda^*(\mathbf{t})$. Suppose that the eigenvalue $\lambda^*(\mathbf{t})$ has multiplicity greater than one in a neighborhood $\mathcal{N}(\mathbf{t}^*)$ of $\mathbf{t}^*$. Then, there are more than one linearily independent eigenvector. That is multiple proper signals $v^*(\mathbf{t})$ for $\mathbf{t} \in \mathcal{N}(\mathbf{t}^*)$. Thus, $v^*(\mathbf{t}^*)$ is not unique which is a contradiction. Therefore, the uniqueness of $v^*(\mathbf{t}^*)$ implies that the eigenvalue $\lambda^*(\mathbf{t})$ is simple in the neighborhood of $\mathbf{t}^*$ and hence $\lambda^*(\mathbf{t})$ is smooth in the neighborhood of $\mathbf{t}^*$. ■

Note that if the minimization problem inside of the curly brackets of (3.47) is defined without (3.47b), the minimal proper signal is not limited to PWC functions but can be any L^2 integrable function. It has been proved that if the matrices D_0 and D_1 in (3.47f) and (3.47g), respectively, are the same, the minimal proper signals are continuous functions [13]. In our codes, it is easier to work on compact sets rather than open sets. The next lemma justifies doing that.

LEMMA 3.4. *If the minimization problem, which is defined inside of the curly brackets of (3.47) but excluding (3.47b), has a continuous global minimizer of the analog optimal signal design problem, then the local minimizers of the problem (3.47)*

are equal to those of (3.49), i.e.

$$\left\{\mathbf{t}^* \middle| f(\mathbf{t}^*) \leq f(\mathbf{t}),\, \mathbf{t} \in \mathcal{N}(\mathbf{t}^*) \subset \Gamma_n \right\} = \left\{\mathbf{t}^* \middle| f(\mathbf{t}^*) \leq f(\mathbf{t}),\, \mathbf{t} \in \mathcal{N}(\mathbf{t}^*) \subset \overline{\Gamma}_n \right\}. \quad (3.55)$$

Proof The continuous global minimizer v_c^* is an optimizer that is found in the set of L^2 integrable functions, and this set includes the set of the PWC functions associated with the feasible v-grid vectors $\overline{\Gamma}_n$ of the problem (3.49). Thus, v_c^* has smaller q- norm than any PWC signals $\overline{v}^*$ that correspond to minimizers $\mathbf{t}^* \in \overline{\Gamma}_n$ for every n. Note that convergence in the q-norm is equivalent to L^2 convergence. In addition, as n increases, $\overline{v}^*$ converges to v_c^*. Note that a PWC signal with $n+1$ constant pieces can approximates a continuous function more closely than one with n or fewer constant pieces. Also, note that the PWC signals made by $\mathbf{t} \in \Gamma_n$ have more constant pieces that those by $\mathbf{t} \in \overline{\Gamma}_n \backslash \Gamma$. This implies that no minimizer of the problem (3.49) is in $\overline{\Gamma}_n \backslash \Gamma_n$. Therefore,

$$\begin{aligned} & \left\{\mathbf{t}^* \middle| f(\mathbf{t}^*) \leq f(\mathbf{t}),\, \mathbf{t} \in \mathcal{N}(\mathbf{t}^*) \subset \overline{\Gamma}_n \right\} \\ = \; & \left\{\mathbf{t}^* \middle| f(\mathbf{t}^*) \leq f(\mathbf{t}),\, \mathbf{t} \in \mathcal{N}(\mathbf{t}^*) \subset \Gamma_n \right\} \cup \left\{\mathbf{t}^* \middle| f(\mathbf{t}^*) \leq f(\mathbf{t}),\, \mathbf{t} \in \mathcal{N}(\mathbf{t}^*) \subset \overline{\Gamma}_n \backslash \Gamma_n \right\} \\ = \; & \left\{\mathbf{t}^* \middle| f(\mathbf{t}^*) \leq f(\mathbf{t}),\, \mathbf{t} \in \mathcal{N}(\mathbf{t}^*) \subset \Gamma_n \right\} \cup \emptyset. \end{aligned}$$

■

3.3 Computational Experiments

There are a number of available software packages for nonlinear constrained optimization. For our problem, we take advantage of a nonlinear optimization package in Scilab [11] called *optim*. We wrote the objective function $f(\mathbf{t})$ in a Scilab program given in [15]. It is easily fed into the optimization routine. However, as is often the

case with optimization applications, there are some technical issues which we shall now discuss. For comparison purposes, we let v_c^* , be the minimal continuously varying signal, $v_{u,n}^*$ the minimal piecewise signal with $n+1$ pieces and uniform grid, $v_{p,n}^*$ the minimal piecewise signal with $n+1$ pieces, and $v_{s,n}^*$ the minimal sampled data signal with $n+1$ pieces using uniform sampling. Where no confusion results we omit the n subscript. The examples will be used both to discuss the grid optimization and to illustrate the difference between the optimal sampled data and optimal PWC signals on coarse v-grids.

3.3.1 Implementation

optim contains nonlinear unconstrained and bound-constraint optimization routines. Our problem has inequality constraints $A\mathbf{t} < b$ (3.50) which are more complex than simple bound constraints. In our implementation we replace the inequality constraints with bound constraints during each iteration and then follow that with a nonlinear projection of $\mathbf{t}$ into $\bar{\Gamma}_n$ after each iteration. More precisely, each time when an iterate $\mathbf{t}$ is selected in the bound constraints

$$\left(0,\cdots,0\right)^T \leq \mathbf{t} \leq t_f \left(1,\cdots,1\right)^T, \tag{3.56}$$

the elements of $\mathbf{t}$ are sorted in increasing order. This preconditioning ensures that our iterates stay in the set $\overline{\Gamma}_n$ while allowing us to use *optim* as a part of our v-grid optimizer. We select test problems that meet the condition of Lemma 3.4, that $D_0 = D_1$, so that optimization over $\overline{\Gamma}_n$ will provide the solution to our problem of optimization over Γ_n.

There are many methods for solving nonlinear simple-bound optimization problems [28]. We use the *QN* parameter in *optim* to utilize a limited memory projected quasi Newton method with BFGS updates (L-BFGS) [9]. This L-BFGS routine is

designed for twice continuously differentiable objective functions, and the differentiability is the key for convergence. One problem with using this method is that our objective function may not satisfy this differentiability condition. Nevertheless, it is at least piecewise-smooth as discussed earlier. Further, if initial iterates are close enough to a locally smooth minimizer, in exact arithmetic L-BFGS will march to the minimizer. Note that we assume the existence of such minimizer.

In the following computational tests, for each problem we repeat the optimization process several times using different initial solutions. The gradient vector is approximated by a first-order difference

$$\nabla f(\mathbf{t})_i \approx \frac{f(\mathbf{t} + h \cdot e_i) - f(\mathbf{t})}{h} \quad \text{where} \quad \nabla f(\mathbf{t}) = \begin{pmatrix} \nabla f(\mathbf{t})_1 \\ \vdots \\ \nabla f(\mathbf{t})_n \end{pmatrix} \tag{3.57}$$

and e_i is the ith unit vector. Stopping criteria can be specified in five ways. Considering possible non-differentiabilty at points, iteration stagnation, expensive function evaluations, and function evaluation accuracy, we utilize all the five by limiting the numbers of iterations and function evaluations to 20 each and setting the minimum variations in iterates, gradients and objective values to 10^{-4}.

3.3.2 Example 1 : Unique global optimizer

We revisit the example of a single engine F-16 supersonic test vehicle [48]. The equalized and linearized model of the test vehicle and the modes representing its

normal operation are given by

$$\begin{aligned}
\dot{x}_0 &= \begin{pmatrix} -0.1689 & 0.0759 & -0.9952 \\ -26.859 & -2.5472 & 0.0689 \\ 9.3603 & -0.1773 & -2.4792 \end{pmatrix} x_0 + \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} v + \begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix} \nu_0 \\
y &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0.9971 & 0.0755 \end{pmatrix} x_0 + \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix} \nu_0.
\end{aligned}$$

A fault simulating an electrical interruption to a flight control computer's input channels may be represented the same as (2.7) except

$$A_1 = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}.$$

Let $P_0 = P_1 = I$. The three states are side slip, roll rate, and yaw rate. The control is $v = \begin{pmatrix} v^a & v^b \end{pmatrix}^T$ where v^a is side slip acceleration command (rudder input) and v^b is the stability axis roll acceleration command (stick input). The detection signal is applied on the same channels as the control vector and it is assumed that during the test period the control is nulled. The test period is [0,5]. We take $q(v)$ to be the L^2 norm of v.

Before applying a nonlinear optimization algorithm, we study the cost function $f(\mathbf{t})$ of the example. Figure 3.1 shows the graphs of $f(\mathbf{t})$ when $\mathbf{t} \in \Gamma_1$ and Γ_2. The values of $f(\mathbf{t})$ are computed on 30 and 30×30 meshes. In the 3-D figure on the right side of Figure 3.1, the white triangle domain is the Γ_2, and only the plot above it is meaningful. It is clearly shown that the two cost functions are smooth in Γ_1 and

Γ_2. There is a unique global minimizer $\mathbf{t}_g$ near 1.73 and $[1.04, 2.41]$ in each case. The minimum values $f(\mathbf{t}_g)$ are about 2.759 and 2.565, respectively. Note that when v-grid vectors $\mathbf{t} = [t_1, t_2] \in \Gamma_2$ satisfy $t_1 = 0$, $t_2 = t_f$ or $t_1 = t_2$, the $\mathbf{t}$'s are the same as having one v-grid point, i.e. $\mathbf{t} \in \Gamma_1$. It is observed that the shape of the curve above the t_1 and t_2 axes and the line $t_1 = t_2$ in the 3-D graph coincides with the graph on the left.

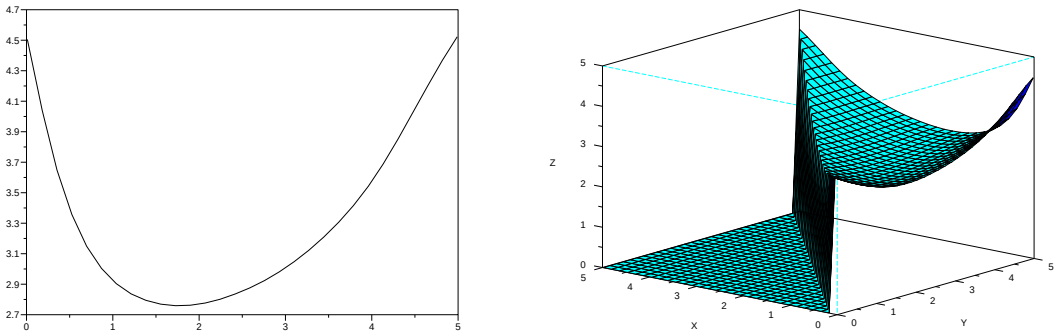

Figure 3.1: Example 1: Objective function $f(\mathbf{t})$ when $\mathbf{t} = [t_1]$ and $\mathbf{t} = [t_1, t_2]$

The v-grid optimization is tested on Γ_n where $n = 1, 2, 3, 4$. For each n, our goal is to obtain a v-grid vector $\mathbf{t}^*$ generating a minimal proper PWC signal with $n + 1$ constant pieces. The results are summarized in Table 3.1.

IT is the number of iterations, Fevl is the number of function evaluations, and STOP is what lead to iteration termination. The usual stopping criteria was the variation in f (f-var).

For A2 and A7 we do not get convergence. A2 is terminated at a v-grid vector that is not quite at the optimizer yet. That is, through a line search failure which resulted in a change in $\mathbf{t}$ ($\mathbf{t}$-var) that was below tolerances. A7 shows that L-BFGS completely fails to move from the given initial solution to a better solution resulting in a a linear

Table 3.1: Example 1: v-grid optimization results

	initial iterate				final iterate $\mathbf{t}^*$				$f(\mathbf{t}^*)$	IT	Fevl	STOP
A1	1				1.769				2.7586	3	7	f-var
A2	3.9				1.233				2.8302	2	7	$\mathbf{t}$-var
A3	1	4			1.065	2.489			2.5646	5	10	f-var
A4	0.1	0.4			1.065	2.490			2.5646	6	11	f-var
A5	2	2.5	3		0.741	1.687	2.933		2.5004	7	10	f-var
A6	0.5	2	4.5		0.743	1.695	2.932		2.5004	8	13	f-var
A7	0.1	0.4	0.6		0.1	0.4	0.6		3.2363	1	17	LSF
A8	0.5	2	2.5	3	0.566	1.279	2.121	3.220	2.4712	9	15	f-var
A9	3.5	4	4.2	4.6	0.567	1.278	2.119	3.223	2.4712	11	19	f-var

search failure (LSF). Further investigation revealed that these two failures are caused by numerical error rather than algorithmic failure. In A2, the objective function curve is relatively flat near the iterate 1.233, but its gradient is clearly negative. It appears that the inaccuracy of function evaluation is large relative to step size $h = 10^{-5}$. As a result the approximated gradient is computed to be a positive value, the opposite sign of the true gradient. Similar numerical problems occur in A7 at the initial point $(0.1, 0.4, 0.6)$. With a larger h, say 10^{-2} for A2 and 10^{-4} for A7, both iterations converge to their minimizers.

As we scan down the 4th column of Table 3.1, we notice the decrease of $f(\mathbf{t}^*)$. As mentioned earlier, there exists trade-off between easy-implementability and minimality (plant-friendliness). The five-step signal generated by $\mathbf{t}^*$ of A8 (or A9) is relatively harder to implement than the one-step signal of A1 but it has smaller $f(\mathbf{t}^*)$.

When there is no priori information, one of the most natural v-grid points is the uniform one. Uniform grids often return relatively "good" detection signals [15]. To

compare the solution obtained using grid optimization to that of uniform grids, we define a function $r(\mathbf{t}^*)$ as follows

$$r(\mathbf{t}^*) = \frac{|f(\mathbf{t}^*) - f(\mathbf{t}_u)|}{f(\mathbf{t}_u)} \times 100 \tag{3.59}$$

where $\mathbf{t}_u$ is a uniform v-grid vector i.e. $\mathbf{t}_u = \begin{pmatrix} \frac{t_f}{n} & \frac{2t_f}{n} & \dots & \frac{(n-1)t_f}{n} \end{pmatrix}^T$ and $\mathbf{t}^*$ is the optimal or the best v-grid vector known. $r(\mathbf{t}^*)$ indicates how much improvement the grid optimization make in comparison to uniform grid. Table 3.2 shows $r(\mathbf{t}^*)$'s for Example 1. The amount of improvement ranges from 3.35% to 1.04%. Since our fault detection scheme does not require online signal design, optimization can be done in advance securing a smaller ("plant-friendlier") PWC detection signal. Some applications, like chemical processes, that are highly sensitive to auxiliary inputs, would welcome such signal quality improvement. Notice that the smaller n is, the more reduction in v using grid optimization there is. It is also apparent in Figure 3.2 that when n is smaller, the difference between $v^*(\mathbf{t}_u)$ and $v^*(\mathbf{t}^*)$ are bigger. However, this strict reciprocal relationship between n and $r(\mathbf{t}^*)$, $\mathbf{t}^* \in \Gamma_n$ is not always true. We will see a typical exception in the next example.

Table 3.2: Example 1: Uniform grid $\mathbf{t}_u$ vs. optimized grid $\mathbf{t}^*$

	uniform grid $\mathbf{t}_u$	$f(\mathbf{t}_u)$	best grid known $\mathbf{t}^*$	$f(\mathbf{t}^*)$	$r(\mathbf{t}^*)$
Γ_1	2.5	2.8541	1.769	2.7586	3.35
Γ_2	1.67 3.33	2.6183	1.065 2.489	2.5646	2.05
Γ_3	1.25 2.5 3.75	2.5361	0.741 1.687 2.933	2.5004	1.41
Γ_4	1 2 3 4	2.4972	0.566 1.279 2.121 3.220	2.4712	1.04

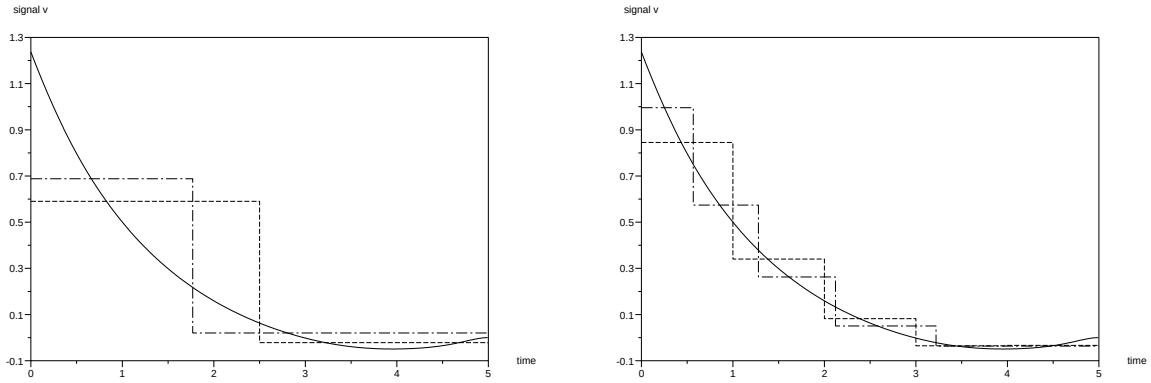

Figure 3.2: Example 1: Detection signals v_c^a (solid), $v^a(\mathbf{t}^*)$ (dash-dot), $v^a(\mathbf{t}_u)$ (dash) for $n = 1$ (left), $n = 4$ (right)

3.3.3 Example 2 : Multiple optimizers

Example 2 is an academic test problem to illustrate several other points of v-grid optimization. A normal model is assumed to have the form

$$\dot{x_0} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} x_0 + \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} v + 10^{-4} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \nu_0 \tag{3.60a}$$

$$y = \begin{pmatrix} 1 & 2 \end{pmatrix} x_0 + \begin{pmatrix} 0 & 0 & 1 \end{pmatrix} \nu_0. \tag{3.60b}$$

The failed model is the same except

$$A_1 = \begin{pmatrix} 0 & 3 \\ -3 & 0 \end{pmatrix}. \tag{3.60c}$$

As before, the detection period is [0,5] and $P_0 = P_1 = I$.

For the problem ex Figures 3.3, 3.4, and 3.5 show the first component of the three optimal signals for each of $n+1 = 3, 10, 100$. We observe that when $n+1 = 3$, v_u^* and v_s^* are not only quite different from v_c^* but also are very different from each other. Note that when all four places are considered, v_u^* is smaller than v_c^* for relatively large n. Since the PWC algorithm involves computations of a matrix whose dimension grows as n increases, the q values for $n + 1 = 25, 50, 100$ are considered accurate to two places. However, as n increases, v_u^* and v_s^* become more alike and also converge to v_c^* as illustrated in Figure 3.4 and Figure 3.5.

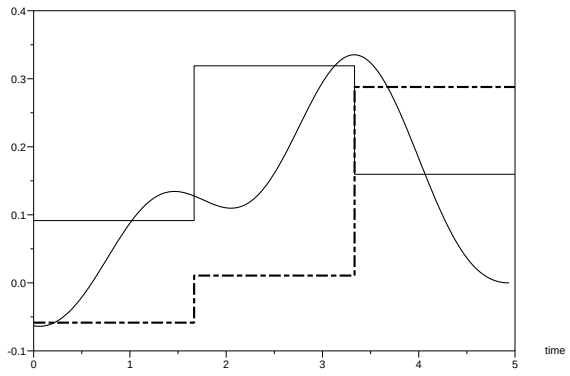

Figure 3.3: Example 2: First component of v_c^* (thick solid), v_u^* (thin solid), and v_s^* (dashed) for $n + 1 = 3$

The L_2 norms of the difference vectors $v_c^* - v_u^*$ and $v_c^* - v_s^*$ are shown in Table 3.3 in the last two columns. Ignoring the first three entries of each column, we observe the PWC column is approximately half the SDS column. Further computations show that the L_2 norm of the difference vectors $v_c^* - v_u^*$ is like the function $\frac{1}{n}$ while that of $v_c^* - v_s^*$ is like $\frac{2}{n+1}$. Thus in this example v_u^* converges to v_c^* about twice as fast as v_s^* does.

We also compare computational efficiency using CPU times which is only an approximate indicator. In Table 3.3, the CPU time is how long it took for the algorithm

Table 3.3: Example 2: Convergence statistics for v_u^* and v_s^*

	q		CPU Time		Diff. from v_c^*	
$n+1$	PWC	SDS	PWC	SDS	PWC	SDS
1	0.4906	6.7537	11.351	6.091	0.5418	2.3884
2	0.3514	1.5554	6.880	4.340	0.3302	1.1564
3	0.3507	0.3254	5.418	3.265	0.4030	0.6280
4	0.2972	0.4794	4.726	2.754	0.1978	0.4128
5	0.2906	0.2864	4.138	2.357	0.1898	0.3128
10	0.2708	0.2685	3.312	1.913	0.0996	0.1901
15	0.2665	0.2652	3.000	2.055	0.0682	0.1370
20	0.2649	0.2640	3.031	2.233	0.0538	0.1100
25	0.2642	0.2635	3.210	2.468	0.0451	0.0923
50	0.2632	0.2629	6.657	6.364	0.0308	0.0575
100	0.2629	0.2628	46.349	47.210	0.0261	0.0403
CS	0.2643		84.390		0	

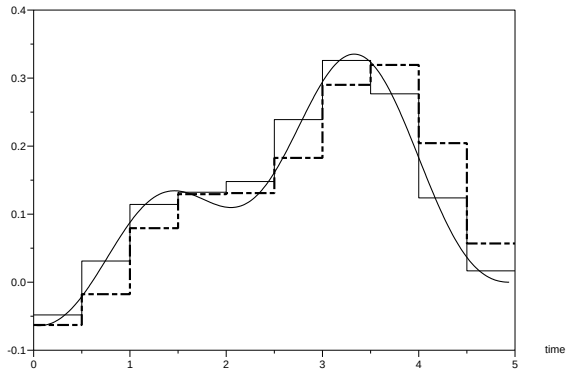

Figure 3.4: Example 2: First component of v_c^* (thick solid), v_u^* (thin solid), and v_s^* (dashed) for $n+1=10$

to finish a β-grid iteration where most of the core computations for finding the optimal signals are implemented. During a β-iteration each algorithm executes a set of computations 51 times with one of 51 β values.

Note that when $n+1=100$, while the v_u^* and v_s^* signals are very similar to v_c^* in norm, the CPU time for v_u^* and v_s^* are much less than that of v_c^*. Thus the algorithms of this paper could be computationally useful even in the computation of estimates of an optimal analog signal. However, it should be noted that CPU time reduction might not come easily in the case where a model has a larger state dimension.

When n is small, such as in Figure 3.3, we see a major difference between v_u^* and v_s^* with the difference becoming more dramatic as n takes on lower values. In some applications the most desirable values for implementation purposes might be $n+1=1$ or $n+1=2$. Looking at the size of the test signals in the second and third columns of Table 3.3 we see that the SDS signal is approximately 14 times as large as the uniform PWC signal. By the time n has increased to three, the sizes of the test signals is approximately the same. However, there is an important difference. v_u^* is a proper test signal for the original continuous time system with continuous output

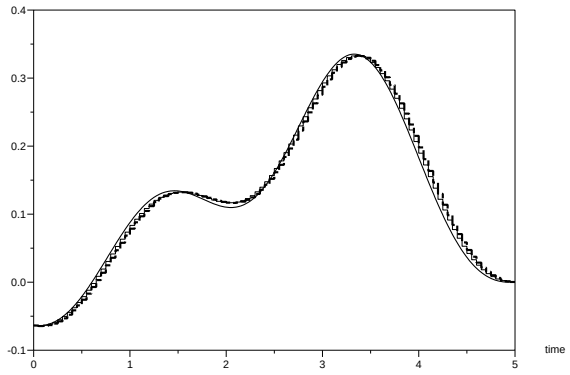

Figure 3.5: Example 2: First component of v_c^* (thick solid), v_u^* (thin solid), and v_s^* (dashed) for $n+1=100$

observation. It is also the minimal piecewise constant one with $n+1$ uniform steps. Therefore it guarantees fault detection in the original continuous time problem for the given error bounds. The SDS test signal v_s^* is not guaranteed to be proper for CS since the permisable uncertainty for the SDS are a subset of those for CS. The SDS test signal can be used in the continuous time system with continuous output observation but we need to either use a larger multiple of v_s^* to guarantee we have a proper test signal or we need to be willing to accept guaranteed detection for reduced error bounds. It is possible to compute both what the multiple would need to be and also what the reduced error bound would be which would make v_s^* proper. However, that is not done with the techniques of this paper and requires sophisticated optimization software [7] as in [12].

As shown in Figure 3.6 the cost function $f(\mathbf{t})$ of this particular problem is smooth in the two feasible regions Γ_1, Γ_2. The computational results show that the global minimizer in each case is achieved at about 2.39 and $\begin{pmatrix} 2.41 & 4.13 \end{pmatrix}^T$ with objective value 0.3507 and 0.3117, respectively. Notice the inflection point in the Figure 3.6 on the left and the second minimizer in the graph on the right. In Example 1, both

objective functions were strictly convex and had a unique minimizer. Example 2 shows that the objective function that we try to minimize could be quite complicated with many local minima.

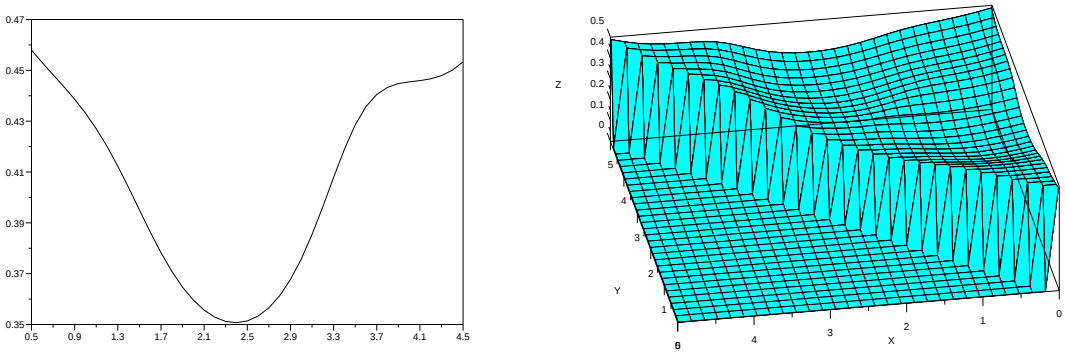

Figure 3.6: Example 2: Objective function $f(\mathbf{t})$ when $\mathbf{t} = [t_1]$ (left) and $\mathbf{t} = [t_1, t_2]$ (right)

For this example, we performed grid optimization for $n = 1, 2, 3, 4$, as before. The overall results are summarized in Table 3.4. While B1-B4 successfully find the global minimizers, the initial solutions of B5 and B6 converge to the second best minimizer. The optimization B7-B13 in the domain Γ_3 and Γ_4 are also successful finding various minimizers.

Among many possible initial solutions, B3, B6, B8, and B11 choose the vectors of uniform grid points, which are likely used for PWC detection signal design if one does not utilize a systematic method for searching grid points with better quality. All four vectors of uniform grid converged to other v-grid vectors after a number of iterations. Again, we ask how much we gain through the optimization process. Table 3.5 shows the values of $r(\mathbf{t}^*)$ defined in (3.59).

As seen in Figure 3.6 on the left, the optimal v-grid point $\mathbf{t}^*$ is close to the uniform grid point 2.5 when $n = 1$. As a result, the detection signal of $\mathbf{t}^*$ is similar to that

Table 3.4: Example 2: Grid optimization results

	initial iterate				final iterate $\mathbf{t}^*$				$f(\mathbf{t}^*)$	IT	Fevl
B1	1.3				2.393				0.3507	3	9
B2	3.3				2.392				0.3507	4	11
B3	2.5				2.393				0.3507	2	5
B4	1	4			2.379	4.043			0.3114	4	9
B5	0.1	0.4			0.954	2.502			0.3166	5	9
B6	1.667	3.333			0.945	2.503			0.3166	5	9
B7	0.1	2.51	4.31		0.893	2.482	4.073		0.2838	3	6
B8	1.25	2.5	3.75		0.929	2.496	4.079		0.2839	2	9
B9	3	3.5	4.5		2.383	3.821	4.236		0.3079	4	8
B10	0.1	0.4	0.6		0.717	1.146	2.501		0.3130	7	13
B11	1	2	3	4	0.923	2.296	2.694	4.075	0.2775	2	5
B12	1	2.5	3.5	4.1	0.888	2.491	3.931	4.306	0.2810	6	13
B13	0.1	0.4	0.6	0.9	0.585	0.880	1.185	2.497	0.3121	9	17

Table 3.5: Example 2: Uniform v-grid $\mathbf{t}_u$ vs. optimized $\mathbf{t}^*$

	$f(\mathbf{t}_u)$	best grid known $\mathbf{t}^*$				$f(\mathbf{t}^*)$	$r(\mathbf{t}^*)$
Γ_1	0.3514	2.393				0.3507	0.19
Γ_2	0.3507	2.379	4.043			0.3114	12.62
Γ_3	0.2972	0.893	2.482	4.073		0.2838	4.70
Γ_4	0.2906	0.923	2.296	2.694	4.075	0.2775	4.71

of the uniform grid gaining little advantage of optimization. The first component of two signals are shown in Figure 3.7 on the left. On the other hand, notice a big gap between the two signals in Figure 3.7 on the right. When $\mathbf{t}_u$, $\mathbf{t}^* \in \Gamma_2$, the optimized grid $\mathbf{t}^*$ generates a PWC signal that is 12% less compared to the uniform v-grid generated signal. For some systems that are highly sensitive to inputs, the reduction in v or more precisely the "plant-hostility" feature of v can be significant.

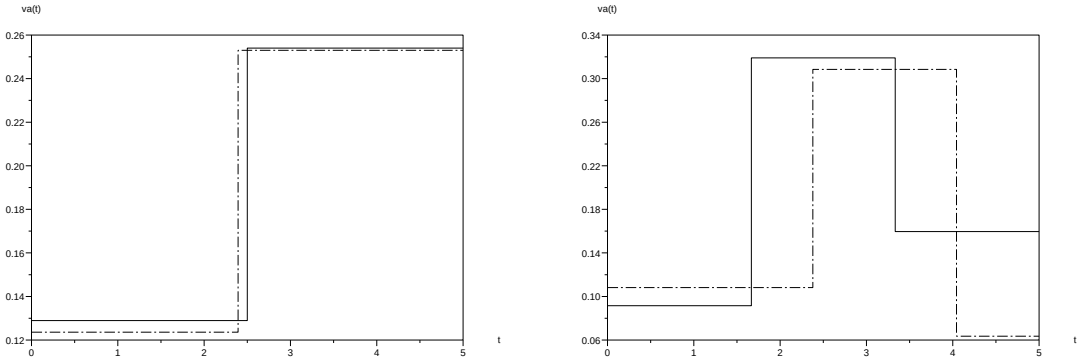

Figure 3.7: Example 2: detection signal $v^a(\mathbf{t}^*)$ (dash-dot) and $v^a(\mathbf{t}_u)$ (solid) when $n = 1$ (left), $n = 2$ (right)

3.4 Fault Detection Tests

Given a proper test signal has been computed there remains the task of using it. Since the PWC signal is proper for the optimal analog signal design problem it may be used for fault detection as in [13]. Two methods are presented there and Section 1.4.2 briefly summarized them. In this section, we discuss the implementation of the two detection tests when using optimal digital signals.

Standard test with optimal digital signal

The standard test is to set up a filter for each model. Each filter essentially computes the amount of uncertainty that would be required for the model to produce the

observed output up to that point. A model is rejected when this value exceeds the noise bound for that model.

The actual formula to implement these tests is obtained by solving the problem (1.40) and is given as theorem 3.3.4 in [13]. For completeness, we include the formula here.

The standard test filter: Suppose that

1. *Either N_i is invertible for all t or*

$$N_{i\perp}^T J_i N_{i\perp} > 0, \ \forall\, t \in [0, t_f] \tag{3.61}$$

2. *The Riccati equation*

$$\begin{aligned}\dot{P}_i = (A_i - S_i R_i^{-1} C_i) P_i + P_i (A_i - S_i R_i^{-1} C_i)^T \\ - P_i C_i^T R_i^{-1} C_i P_i + Q_i - S_i R_i^{-1} S_i^T, \qquad P_i(0) = P_{i,0},\end{aligned} \tag{3.62}$$

where

$$\begin{pmatrix} Q_i & S_i \\ S_i^T & R_i \end{pmatrix} = \begin{pmatrix} M_i \\ N_i \end{pmatrix} J_i^{-1} \begin{pmatrix} M_i \\ N_i \end{pmatrix}^T \tag{3.63}$$

has a solution on $[0, t_f]$.

Then, a realizability test for model i is

$$\gamma_{i,s}(v, y) < 1, \quad \text{for all } s \in [0, t_f] \tag{3.64a}$$

where

$$\gamma_{i,s} = \int_0^s \mu_i^T R_i^{-1} \mu_i \, dt, \tag{3.64b}$$

and μ_i is the output of the following system:

$$\dot{\hat{x}}_i = A_i\hat{x}_i - (S_i + P_iC_i^T)R_i^{-1}\mu_i + B_iv^* \tag{3.64c}$$

$$\mu_i = C_i\hat{x}_i - (E_iy - D_iv^*) \tag{3.64d}$$

with $\hat{x}_i(0) = 0$.

Hyperplane test with optimal digital signal

Hyperplane test is to find a hyperplane that separates the two output sets and to determine which side of the hyperplane an observed output corresponds to, deciding from which condition of the observing system the output is produced.

The separating hyperplane test was depicted in Figure 1.4 and described in (1.41) as

$$\int_0^{t_f} a(t)^T(\bar{y}(t) - y^*(t))\,dt \lesseqgtr 0 \tag{3.65}$$

where a is a normal vector to the hyperplane and $\bar{y}$ is a point on the hyperplane.

In Chapter 1.4.2 we discussed that solving the constrained optimization problem (1.42) provides the required information of the hyperplane, i.e. a and y, assuming that the applied detection signal separates the normal and faulty output sets such that the closures of the two sets have at least one point of intersection.

This geometry changes when we use digital detection signals. The assumption of existence of intersecting points is, in general, not met even when we use the minimal proper digital signal. Note that the set of PWC functions with a fixed number of constant pieces is not dense in L^2 functional space. Thus, while an optimal analog detection signal guarantees more than one intersecting points in the closures of the sets, an optimal digital signal may not. The Figure 3.8 depicts the geometry of the two output sets after applying an optimal proper analog signal and a proper digital

detection signal $\overline{v}^*$.

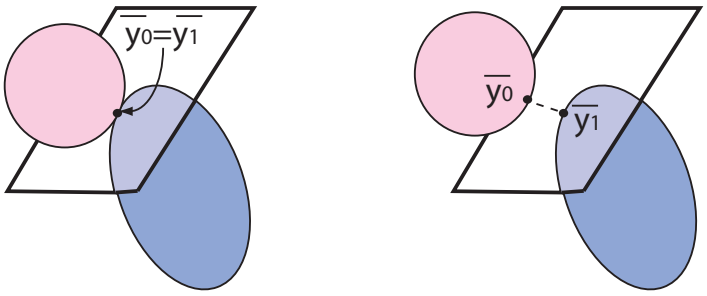

Figure 3.8: Output sets after applying an optimal proper analog signal (left) and a proper digital signal (right)

[13] and [17] discuss a numerical approach to find the separating hyperplane for a proper v when the two output sets are a distance apart from each other without any point intersecting. Utilizing that approach, when we have a digital proper detection signal, the fault detection test can be done as follows:

Offline Hyperplane Computation:

1. Let $\overline{v}^*$ be a proper detection signal from the optimal digital signal algorithm. (Scilab)

2. Perform constrained optimization (SOCS) to solve the problem

$$\min \|y_0 - y_1\|^2 \tag{3.66a}$$

subject to the constraints

$$\begin{aligned}
\dot{x}_0 &= A_0 x_0 + B_0 \overline{v}^* + M_0 \nu_0 && (3.66b)\\
E_0 y_0 &= C_0 x_0 + D_0 \overline{v}^* + N_0 \nu_0 && (3.66c)\\
\dot{\omega}_0 &= \nu_0^T J_0 \nu_0, \;\; \omega_0(0) = x_0(0)^T P_{0,0} x_0(0), && (3.66d)\\
\omega_0(s) &\leq 1, \;\; 0 \leq s \leq t_f && (3.66e)\\
\dot{x}_1 &= A_1 x_1 + B_1 \overline{v}^* + M_1 \nu_1 && (3.66f)\\
E_1 y_1 &= C_1 x_1 + D_1 \overline{v}^* + N_1 \nu_1 && (3.66g)\\
\dot{\omega}_1 &= \nu_1^T J_1 \nu_1, \;\; \omega_1(0) = x_1(0)^T P_{1,0} x_1(0), && (3.66h)\\
\omega_1(s) &\leq 1, \;\; 0 \leq s \leq t_f && (3.66i)
\end{aligned}$$

3. Let $\bar{y}_0$ and $\bar{y}_1$ be the closest points computed by the optimization.

4. Compute $a(t)$ be the normal vector of the separating hyperplane (Scilab)

$$a = \frac{\bar{y}_0 - \bar{y}_1}{\|\bar{y}_0 - \bar{y}_1\|}, \tag{3.67}$$

5. Compute $\bar{y}(t)$, the point on the separating hyperplane, as the midpoint, of the line segment connecting $\bar{y}_0$ and $\bar{y}_1$ (Scilab)

$$\bar{y} = \frac{\bar{y}_0 + \bar{y}_1}{2}. \tag{3.68}$$

6. Compute an output value y_0 (or y_1) of Model 0 (or Model 1) by solving

$$y_i = C_i x_i^* + D_i \overline{v}^* \tag{3.69}$$

where x_i^* is the unique solution of the ordinary differential equation $\dot{x}_i = A_i x_i +$

$B_i\overline{v}^*$ with $x_i(0) = 0$

7. Determine which side of the hyperplane Model 0 (or Model 1) is located. Let

$$\phi(z) = \langle a, z - \bar{y} \rangle. \tag{3.70}$$

Suppose that

$$\phi(y_0) = \underline{\epsilon} > 0. \tag{3.71}$$

Then it follows that

$$\phi(y_1) = \overline{\epsilon} < 0. \tag{3.72}$$

Online Fault Detection:

1. Apply $\overline{v}^*$ to the system under examination and measure output y^* at time t_f.

2. If $\phi(y^*) < 0$, conclude that the tested system is operating under faulty condition. Otherwise, the system is in normal condition.

It is assumed for an optimal PWC signal to push two output sets enough away from each other. Thus, the normal vector a in (3.67) is well defined.

3.5 Conclusion

Our goal to design digital signals for fault detection has been formulated into a model identification problem and we further sought more desirable digital signals through v-grid optimization. The integration of model identification and grid optimization problems were expressed as a nonlinear constrained optimization problem. Under local smoothness assumption, the problem was solved numerically using a scientific software package. Two computational studies show that the solution of the v-grid

optimization problem can sometimes improve the quality of detection signal by significant amounts. In practice, detection signal design can be done offline. Thus, when systems are sensitive relative to the extra input, or when the continuous time test signal is not well approximated by a coarse uniform grid, it is recommended to optimize the grid-points of piecewise constant detection signals.

We designed digital signals based on continuously observed system behavior and compared them with those that are computed in a sampled-data setting. We showed that in particular when a simple detection signal is required, our digital signals possess better quality than those that were designed on limited information.

Chapter 4

Future Work

We discuss some of the remaining open problems concerning the fault detection approach considered in this dissertation.

4.1 Fault Detection Using Optimal Digital Signal

Alternative algorithm for hyperplane test

When we considered a hyperplane test as a fault detection method when using designed digital signals, a numerical algorithm is examined to find the set-separating hyperplane. For a part of the algorithm an optimal control software SOCS was suggested to be used in determining the minimizers y_0^* and y_1^* for the problem (3.66) while the rest of algorithm was implemented by Scilab. While we have not done so, another way to implement the hyperplane test is to derive the closed-form solution to (3.66) and implement it in Scilab or in any other numerical computation software package. [13] provided a closed formula for the hyperplane when an optimal analog detection signal is used. In that case, the optimal control problem (1.42) was formulated.

To solve (3.66), it may be easier to consider

$$\begin{aligned} \dot{x}_i &= A_i x_i + B_i \bar{v}^* + \mathcal{M}_i \mu_i && \text{(4.1a)} \\ 0 &= G_i x_i + H_i \bar{v}^* - z_i && \text{(4.1b)} \\ y_i &= \mathcal{C}_i x_i + \mathcal{D}_i \bar{v}^* + \mathcal{N}_i \mu_i && \text{(4.1c)} \end{aligned}$$

instead of

$$\begin{aligned} \dot{x}_i &= A_i x_i + B_i \bar{v}^* + M_i \nu_i && \text{(4.2a)} \\ E_i y_i &= C_i x_i + D_i \bar{v}^* + N_i \nu_i. && \text{(4.2b)} \end{aligned}$$

Note that the equivalent relationship between (4.1) and (4.2) was discussed in Chapter 2.1.1. Here we set

$$\nu_i = \begin{pmatrix} \mu_i \\ z_i \end{pmatrix}, \; M_i = \begin{pmatrix} \mathcal{M}_i & 0 \end{pmatrix}, \; E_i = \begin{pmatrix} 0 \\ I \end{pmatrix},$$

$$C_i = \begin{pmatrix} G_i \\ \mathcal{C}_i \end{pmatrix}, \; D_i = \begin{pmatrix} H_i \\ \mathcal{D}_i \end{pmatrix}, \; N_i = \begin{pmatrix} 0 & -I \\ \mathcal{N}_i & 0 \end{pmatrix}.$$

Then, the problem (3.66) can be rewritten as

$$\min_{x_0(0),\mu_0,x_1(0),\mu_0} \int_0^{t_f} \left| \mathcal{C}_0 x_0 - \mathcal{C}_1 x_1 + \mathcal{N}_0 \mu_0 - \mathcal{N}_1 \mu_1 + (\mathcal{D}_0 - \mathcal{D}_1) \bar{v}^* \right|^2 dt \qquad \text{(4.3a)}$$

subject to the constraints

$$\begin{aligned}
\dot{x}_0 &= A_0 x_0 + B_0 \bar{v}^* + \mathcal{M}_0 \mu_0 && (4.3b)\\
\dot{x}_1 &= A_1 x_1 + B_1 \bar{v}^* + \mathcal{M}_1 \mu_1 && (4.3c)\\
\dot{\omega}_0 &= \mu_0^T \mu_0 - (G_0 x_0 + H_0 \bar{v}^*)^T (G_0 x_0 + H_0 \bar{v}^*), && (4.3d)\\
\dot{\omega}_1 &= \mu_1^T \mu_1 - (G_1 x_1 + H_1 \bar{v}^*)^T (G_1 x_1 + H_1 \bar{v}^*), && (4.3e)\\
\omega_0(s) &\leq 1, \ \forall\ s \in [0, t_f], && (4.3f)\\
\omega_1(s) &\leq 1, \ \forall\ s \in [0, t_f], && (4.3g)\\
\omega_0(0) &= x_0(0)^T P_{0,0} x_0(0), && (4.3h)\\
\omega_1(0) &= x_1(0)^T P_{1,0} x_1(0). && (4.3i)
\end{aligned}$$

This is an optimal control problem with inequality state constraints. Necessary conditions for the optimality of such type of problems have been investigated extensively in the past as well as numerical methods for the solutions [10], [24], [31], [44].

It would be interesting to derive theoretical answers to the optimal control problem and to compare them with the approximated solutions found by SOCS.

Digital detection signal design in sampled-data setting

For a physical system whose dynamic behavior varies continuously in time, it is often that only a part of the behavioral information is exhibited for technical or economical reasons. Nikoukhah and Campbell [41] developed an active fault detection approach for such systems where they designed a digital (or piecewise constant) detection signal v_s^* based on snapshot information taken at a set of discrete time points. In Chapter 3, we have compared the PWC signal v_s^* with another digital signal v_u^* which was designed by fully utilizing all the information of the same type of system. We noticed

that v_s^* is much less favorable than v_u^* as its disruption of regular operation measures relatively higher, in particular, when the detection signal is required to be simple.

It turns out that v_s^* is computed by equating the number of samples with the number of constant pieces of a detection signal. This equation results in ignoring significant portion of system information when a simple digital signal is requested for fault detection. As a simple extension to [41], one may free up the equality condition and allow to exhibit as many number of behavior information as specified. Assuming that uniform sampling of m information is suggested for each constant piece, the problem of designing minimal proper digital signal with n constant pieces can be formulated as follows:

$$\min_{v} q(v) \tag{4.4a}$$

$$\text{subject to} \quad v(t) = v_j \text{ if } t_j \leq t < t_{j+1},\ j = \{0, \ldots, n\},\ t_0 = 0,\ t_{n+1} = t_f, \tag{4.4b}$$

$$\text{such that} \quad \max\left\{ \quad \omega_v^0(x_0(0), \nu_0, k) \quad , \quad \omega_v^1(x_1(0), \nu_1, k) \quad \right\} \geq 1 \tag{4.4c}$$

$$\text{for all } k = 1, \ldots, n+1 \text{ and for all } x_i, \nu_i \text{ that satisfy}$$

$$\dot{x}_0 = A_0 x_0 + B_0 v + M_0 \nu_0, \tag{4.4d}$$

$$z_0 = G_0 x_0 + H_0 v \tag{4.4e}$$

$$y(j) = C_0 x_0(t_j) + N_0 \mu_0(j), \quad j = 1, \ldots, mn \tag{4.4f}$$

$$\dot{x}_1 = A_1 x_1 + B_1 v + M_1 \nu_1, \tag{4.4g}$$

$$z_1 = G_1 x_1 + H_1 v \tag{4.4h}$$

$$y(j) = C_1 x_1(t_j) + N_1 \mu_1(j), \quad j = 1, \ldots, mn \tag{4.4i}$$

where

$$\omega_v^i(x_i(0), \nu_i, k) = x_i(t_0)^T P_{i,0} x_i(t_0) + \int_{t_0}^{t_k} |\nu_i|^2 - |z_i|^2 \, ds + \sum_{l=1}^{k} \sum_{\substack{j=1+ \\ m(l-1)}}^{ml} \frac{(t_l - t_{l-1})}{m} |\mu_i(j)|^2. \tag{4.4j}$$

Although this extension increases complexity of the problem, the solution approach is expected to be similar to that of [41].

4.2 Incipient Fault Detection

While the work in this dissertation made digital-signal-based fault detection be available for linear continuous systems in addition to the existing analog-signal-based method in [13], it can be applied only for the faults that cause their system to be immediately at a different mode. The multimodel framework relied on such an assumption. Some faults, on the other hand, take the system slowly and gradually to an unhealthy state and it may be desirable to detect them when they start to appear. Coping with such faults is called incipient fault detection. Incipient faults are often modeled by a drift in a system parameter and are indicated by the change in this parameter. Parameter change detection is usually done by comparing an online parameter estimate against its pre-specified threshold. Recently, Nikoukhah and Campbell introduced a two-model approach for discerning parameter change in the context of incipient fault detection in [40]. It is an active detection approach and is designed specifically for an analog signal. As the work in the dissertation, digital signal based method can be developed for diagnosing incipient faults. The system

behavior subject to incipient faults may be modeled as follows:

$$\dot{x} = A(\theta)x + B(\theta)v + M\nu \tag{4.5a}$$

$$y = C(\theta)x + D(\theta)v + N\nu \tag{4.5b}$$

with the uncertainty bound

$$x(0)^T P_0^{-1} x(0) + \int_0^s \nu^T \nu \, dt < 1 \ \forall s \in [0, t_f]. \tag{4.5c}$$

As in [40], a faulty behavior can be described by a small parameter deviation $\delta\theta$ from a reference $\bar{\theta}$. Then, the problem may be set up similarly to the original digital-signal-based detection having this parameter fault considered:

$$\min_v q(v) \tag{4.6a}$$

$$\text{subject to} \quad v(t) = v_j \text{ if } t_j \leq t < t_{j+1},\ j = \{0, \ldots, n\},\ t_0 = 0,\ t_{n+1} = t_f, \tag{4.6b}$$

$$\text{such that} \quad \max\Big(\quad \omega_v^0(x_0(0), \nu_0, k) \quad , \quad \omega_v^1(x_1(0), \nu_1, k) \quad \Big) \geq 1 \tag{4.6c}$$

$$\text{for all } k = 1, \ldots, n+1 \text{ and for all } x_i, \nu_i \text{ that satisfy}$$

$$\dot{x}_0 = A(\bar{\theta})x_0 + B(\bar{\theta})v + M\nu_0 \tag{4.6d}$$

$$y = C(\bar{\theta})x_0 + D(\bar{\theta})v + N\nu_0 \tag{4.6e}$$

$$\dot{x}_1 = A(\bar{\theta} + \delta\theta)x_1 + B(\bar{\theta} + \delta\theta)v + M\nu_1 \tag{4.6f}$$

$$y = C(\bar{\theta} + \delta\theta)x_1 + D(\bar{\theta} + \delta\theta)v + N\nu_1 \tag{4.6g}$$

where

$$\omega_v^i(x_i(0), \nu_i, k) = x_i(0)^T P_0^{-1} x_i(0) + \int_0^{t_k} \nu_i^T \nu_i \, dt, \tag{4.6h}$$

One way to deal with matrices in the form of $X(\bar{\theta}+\delta\theta)$ is to assume that the matrix X is differentiable functions of θ near $\bar{\theta}$ and to use linear approximation. That is to set

$$X(\bar{\theta}+\delta\theta) = X(\bar{\theta}) + \delta\theta X_\theta(\bar{\theta}).$$

[40] also assumes that the initial state of differential variable $\delta\theta x_1$ is zero. These two assumptions are expected to provide a way to solve the problem defined above. The actual computations and solution software development are left to future research.

4.3 Other Extensions

One of the important extensions to this work would be testing the developed algorithms on appropriate physical systems. As a start, a group of people at NAVSEA and NCSU are working on the design and application of a digital signal to detect leakage of a compressor.

While our digital signal design algorithm can handle all uncertainty including model uncertainty, the corresponding software is valid for models with up to additive uncertainty. Current version of the software sets k^* to $n+1$ to find a max λ^* in (3.7). It was possible thanks to the simplification explained in Chapter 3.1.1. To deal with model uncertainty, however, as described in the algorithm it will be required to add another layer of iteration to find the k^*.

Our digital signal design algorithm may also be extended from linear continuous dynamical systems to other types of systems such as linear discrete dynamical systems or nonlinear dynamical systems. Although some initial computational studies [12] showed that the analog detection signal found from linearlized system models can be used for certain nonlinear systems with some conservatism, it is not clear how well digital signals will work for them. Among numerous questions for these systems, some

are as follows:

- For what type of nonlinear systems can a (analog or digital) signal found from linearlized models be used?
- Can we design a (analog or digital) detection signal that guarantees fault detection in nonlinear systems? If can, what modifications are needed in the linearlized-model based algorithm?

Chapter 5

Concluding Remarks and Summary of Contributions

With the assumption that each of normal and faulty system behaviors can be completely described by a bounded subset of a vector space, we sought digital signals that attain nearly perfect detection. As the signal is an auxiliary input, among many detection guaranteeing signals, we desired the piecewise-constant (PWC) function that causes the least disruption to regular operations.

We first assumed a fixed length of each constant piece for a PWC signal and obtained the desired signal by solving a nested optimization problem. The necessary conditions were derived analytically and were used for formulating the outer problem, a matrix eigenvalue problem. The solution of the matrix eigenvalue problem was then numerically computed in Scilab.

We applied our implementation of the digital signal design algorithm to an academic test problem. To compare the resulting optimal PWC signal to that of the sampled-data (SD) approach, an alternative digital signal design method, we also implemented the SD algorithm in Scilab. We observed that, in particular, when a simple detection signal is required, our digital signals possess more desirable characteristics as an auxiliary fault detection than those that were designed by the SD approach. We

also pointed out the weakness of the SD approach and suggested a way to improve its algorithm in Chapter 4. This work yield two software solutions, PWC and SD, and the publication described below:

- D. Choe, S. L. Campbell, and R. Nikoukhah, *Comparison between failure detection test signals for continuous systems and sampled-data systems*, Proceedings of 2006 IEEE Conference on Decision and Control.

We further ameliorated this signal design algorithm by having it self-compute lengths of constant pieces (LCP: previously noted as v-grid) that attain a better detection signal than the others that could have been produced by neighborhood values of LCP. The optimal LCP was computed by solving a nonlinear inequality constrained optimization problem. After analyzing the existence of a locally smooth optimizer under a not very restrictive assumption, we utilized a Scilab solver that optimizes functions subject to simple bounds and solved the problem by projecting each iterate into the feasible region. Through two computational studies, we confirmed that the LCP-optimizing algorithm produces digital signals with even more improved quality as a auxiliary input particularly when generating detection signals with a few constant pieces. This part of our work contributed to an upgraded version of the PWC software and the following preprint:

- D. Choe, S. L. Campbell, and R. Nikoukhah, *Optimal Piecewise-constant Signal Design for Active Fault Detection*, submitted to the International Journal of Control, January, 2007.

The digital based fault detection was a natural extension of a recently developed fault detection method which utilized analog signals. As their successful fault detection depends heavily on how well a system model mirrors its corresponding physical system, at the beginning of our research, the relationship between the modeling errors and the detection scheme was investigated. In Chapter 2.1, through a computational

case study of a test problem we illustrated the effect of model uncertainty to the analog detection signal solution which computation is dependent on the amount of allowed uncertainty in the model. We explained geometrically the cause of reported algorithmic difficulties in the algorithm when too much uncertainty is allowed and made suggestions on how to deal with them. This part of the work contributed to the following publication:

- D. Choe, S. L. Campbell, and R. Nikoukhah, *Auxiliary Signal Design for Active Failure Detection: A Case Study*, Proceedings of the 2005 International Conference on Control and Automation, 1008 - 1013

One of the main goals of this work was to make the analog signal based fault detection methods more attractive and practical to fault detection practitioner. In one perspective, the development of digital signal based detection was motivated in the same context considering the easier implementation of PWC function compared to a continuously varying one. In Chapter 2.2 we addressed reducing computational cost for designing analog detection signal for the systems that have volatile dynamics and require frequent detection tests. We suggested a suboptimal algorithm which produces continuously varying detection signals possessing somewhat higher energy than the signals of the original algorithm but reduces computational time by a significant amount. We programmed a Scilab driver that controls the two algorithm codes and compared optimal and suboptimal signals to each other. We derived an upper bound of the suboptimal cost value for a set of problems and provided an insight into the bound for the other set of problems. This part of work contributed to the following publication:

- D. Choe, S. L. Campbell, and R. Nikoukhah, *A Comparison of Optimal and Suboptimal Auxiliary Signal Design Approaches for Robust Failure*

Detection, Proceedings of the 2005 IEEE Conference on Control Applications, 1473 - 1478,

and the driver conduced to the following publication:

- S. L. Campbell, K. J. Drake, I. Andjelkovic, K. Sweetingham, and D. Choe, *Model Based Failure Detection Using Test Signals from Linearizations: A Case Study*, Proceedings of 2006 IEEE International Symposium on Intelligent Control, 2659 - 2664.

Finally, in the last chapter, we introduced several open questions, setting up the specific optimization problem to be solved, but leaving the detail of the work to future research. The two different types of signals (suboptimal analog, digital) designed by this work provide enhanced usability in fault detection for certain types of linear continuous dynamical systems. The presented numerical solution tools (PWC, SD) and drivers are promising to become a critical part of supervisory systems.

Appendix A

Software

A.1 Optimal Digital Signal Design Algorithm

A.1.1 Driver when a v-grid is given

```
clear;clc;lines(0)
function sigEnrg=friendly(A0,B0,M0,C0,D0,N0,A1,B1,M1,C1,D1,N1, ...
                                                P0,P1,scrQ,T,v_grid)

//Load functions
getf funcPWC.sce;
getf funcPWC_SDS.sce;

//Combine model system matrices
[nx,nv,ny,nnu0,nnu1,A,B,M,Qwob,Swob,Uwob,Vwob,Wwob,midRwob]=...
               combind2mdls(A0,B0,C0,D0,M0,N0,A1,B1,C1,D1,M1,N1);

//Predefine frequently used matrices
n2x=2*nx; n2xPnv=n2x+nv; n4xPnv=n2xPnv+n2x;
nJb1=n4xPnv^2; nJb2=n2xPnv^2;
nJb3=n4xPnv*n2xPnv; nPsi= n2x^2; nPhi= n2x*nv;
preIx=eye(nx,nx); preIv=eye(nv,nv); pre0x=zeros(nx,nx);
pre0v=zeros(nv,nv);
preI2x=eye(n2x,n2x); pre02x=zeros(n2x,n2x); pre0v2x=zeros(nv,n2x);
preI=eye(n2xPnv,n2xPnv); prePi=[preI2x,pre02x];
prePi0=[zeros(nx,n2x);preIx,pre0x];
prePi1=[preIx,pre0x;zeros(nx,n2x)];
eye_nPsi=matrix(preI2x,[nPsi,1]);
```

```
//Default beta grid
beta_grid=50;
bg=beta_grid; adj=1/(bg*5); bb=linspace(adj,1-adj,bg+1);

//Compute & plot optimal PWC signal for a given v-grid
sigEnrg=compV(v_grid)

endfunction

//Input data: model coefficients, test period, v-grid
//Example 2 in Chapter 3.3.3
A0=[0,-1;1,0]; A1=[0,3;-3,0];
B0=eye(2,2); B1=B0;
M0=0.0001*[eye(2,2),zeros(2,1)]; M1=M0;
C0=[1,2]; C1=C0;
D0=zeros(C0*B0); D1=D0;
N0=[0,0,1]; N1=N0;
P0=eye(A0); P1=P0;
scrQ=[];
T=5;
v_grid=[0.929;2.496;4.079]; //column vector

//Signal design algorithm execution
sigEnrg=friendly(A0,B0,M0,C0,D0,N0,A1,B1,M1,C1,D1,N1,P0,P1,...
        scrQ,T,v_grid);
```

A.1.2 Driver when v-grid is optimized

```
clear;clc; lines(0);
function [fval,xval,gval]=friendliest(A0,B0,M0,C0,D0,N0,A1,B1,M1,...
       C1,D1,N1,P0,P1,scrQ,T,v_grid,stop_crt,stop_x,imp_val,history)

//Load functions
exec funcPWC.sce;
exec funcPWC_SDS.sce;

//Combine model system matrices
[nx,nv,ny,nnu0,nnu1,A,B,M,Qwob,Swob,Uwob,Vwob,Wwob,midRwob]=...
        combind2mdls(A0,B0,C0,D0,M0,N0,A1,B1,C1,D1,M1,N1);

//Predefine frequently used matrices
n2x=2*nx; n2xPnv=n2x+nv; n4xPnv=n2xPnv+n2x;
nJb1=n4xPnv^2; nJb2=n2xPnv^2; nJb3=n4xPnv*n2xPnv;
nPsi= n2x^2; nPhi= n2x*nv;
preIx=eye(nx,nx); preIv=eye(nv,nv);
pre0x=zeros(nx,nx); pre0v=zeros(nv,nv);
preI2x=eye(n2x,n2x); pre02x=zeros(n2x,n2x); pre0v2x=zeros(nv,n2x);
preI=eye(n2xPnv,n2xPnv); prePi=[preI2x,pre02x];
prePi0=[zeros(nx,n2x);preIx,pre0x];
prePi1=[preIx,pre0x;zeros(nx,n2x)];
eye_nPsi=matrix(preI2x,[nPsi,1]);
```

```
//Default beta grid
beta_grid=50;
bg=beta_grid; adj=1/(bg*5); bb=linspace(adj,1-adj,bg+1);

//Compute & plot optimal PWC signal with v-grid optimization
if argn(2)<21 then HISTORY=%F; else HISTORY=%T; end
[fval,xval,gval]=optim(costf,'b',zeros(v_grid),...
     (T-.1^5)*ones(v_grid),v_grid,'qn','ar', stop_crt(1),...
     stop_crt(2),stop_crt(3),stop_crt(4),stop_x,imp=imp_val);

endfunction

//Input data: model coefficients, test period,
//             v-grid, stopping criteria,
//             iteration display option
//Example 2 in Chapter 3.3.3
A0=[0,-1;1,0]; A1=[0,3;-3,0];
B0=eye(2,2); B1=B0;
M0=0.0001*[eye(2,2),zeros(2,1)]; M1=M0;
C0=[1,2]; C1=C0;
D0=zeros(C0*B0); D1=D0;
N0=[0,0,1]; N1=N0;
P0=eye(A0); P1=P0;
scrQ=[];
T=5;
v_grid=[2.5]; //column vector
stop_crt=[20,20,.1^4,.1^4];
```

```
stop_x=.1^4*ones(v_grid);
imp_val=2; //iteration display option

//Signal design algorithm execution
global hist
hist=[];
[fval,xval,gval]=friendliest(A0,B0,M0,C0,D0,N0,A1,B1,M1,C1,D1,N1,...
             P0,P1,scrQ,T,v_grid,stop_crt,stop_x,imp_val,hist);
xval, fval, gval //optional display
if hist~=[] then
   fprintfMat('ite_hist',hist,'%4.4f');
   iter_history=fscanfMat('ite_hist')
end
```

A.1.3 Functions in *funcPWC.sce*

```
   function objVal=compV(t_grid);
//Organize time grid
t_grid=NoRepeat(t_grid,T);
tt=[0,gsort(t_grid,'g','i'),T];

//v-grid
//t0=tt(1); tn=tt($);
tau_vec=[tt(2:$)-tt(1:$-1)]; ns=size(tau_vec,2);

//Cost function
//Default: L2 norm of signal
```

```
if scrQ==[] then
   for i=1:ns,
scrQ=sysdiag(scrQ, tau_vec(i)*eye(nv,nv));
   end;
end

//Find lambda_beta at every beta (0,1)
lam_b=[];lam_s=-1; Del_s=[]; beta_s=-1;  t=tic()
for bt=bb,
  //default & pre-build beta related matrix
  Pb=sysdiag(bt*P0,(1-bt)*P1); //dropping the inverse notation
  Q=bt*Qwob; S=bt*Swob; U=bt*Uwob; V=bt*Vwob; W=bt*Wwob;
  R=2*sysdiag(bt*eye(nnu0-ny,nnu0-ny),..
     (1-bt)*eye(midRwob)+bt*midRwob,..
     (1-bt)*eye(nnu1-ny,nnu1-ny));
  invR=inv(R); mr=M*invR;mru=mr*U';mrm=mr*M';mrw=mr*W;
  ur=U*invR;uru=ur*U';urw=ur*W;
  Pshi=[A-mru,-mrm; uru-Q, mru'-A']; B4Phi=[B-mrw;urw];
  Omg11=sysdiag(Q-uru,mrm);
  Omg12=[V-urw;zeros(nx,nv)];
  Omg22=[S-W'*invR*W];

  //Form {mathcal J_beta}.
  //%ODEOPTIONS=[1,0,0,tau/10,0,2,2000,12,5,0,-1,-1];
  J_beta=compJ(tau_vec(1));
  [J1,J2,J3,J4,J5,J6]=partiJ(J_beta);
  Lam_o=Pb+J2;
```

```
  if min(real(spec(Lam_o)))<=0 then
     error("Not p.d. Cannot use this algorithm"); end;
  Del_o=[];Gam_o=[];
  if size(tau_vec,2)>1 then
     for tau=tau_vec(2:$)
         J_beta=compJ(tau);
         [J1_n,J2,J3_n,J4_n,J5_n,J6_n]=partiJ(J_beta);
         [Del_n,Gam_n,Lam_n]=BuildDel(J1,J2,J3,J4,J5,J6,...
                              Gam_o,Lam_o,Del_o);
         Lam_o=Lam_n; Gam_o=Gam_n; Del_o=Del_n;
         J1=J1_n;J3=J3_n;J4=J4_n;J5=J5_n;J6=J6_n; end
     J2=zeros(J1); J1=J1_n;J3=J3_n;J4=J4_n;J5=J5_n;J6=J6_n;
  else J2=zeros(J1);
  end
  Del_n=BuildDel(J1,J2,J3,J4,J5,J6,Gam_o,Lam_o,Del_o);
  lam=lbcalc(Del_n);
  if lam>lam_s then lam_s=lam; Del_s=Del_n; beta_s=bt; end
  lam_b=[lam_b,lam];
end //beta
betaite_time=toc();

// Plot lambda_beta function
if lam_s == 0 then error('no proper auxiliary signal exists'); end
xset('window',0); xtitle('','beta','lambda');
a=gca(); a.thickness=1.7; a.line_style=1;
plot2d(bb,lam_b)
```

```
// v_n^* & normalization
[al,be,eigvec]=spec(Del_s,scrQ);
v=eigvec(:,find(lam_s==real(al./be)));
cost=costcalc(v);
v_s=v/sqrt(lam_s*cost);

// Compute total energy of signals
v_enrg=[];
for i=1:nv
  xset('window',i); //xtitle('','time','signal v');
  v_sig=v_s(i:nv:$);
  v_enrg=[v_enrg, costcalc(v_sig)]; //L^2 norm of v
  a=gca(); a.thickness=2; a.line_style=1;
  plot2d2(tt',[v_sig;v_sig($)]);
end

//result_str=msprintf("&%1.2f&%1.4f&%1.4f&%1.4f&%1.4f&%1.4f&..
     %1.3f\\", tt(2),beta_s,lam_s,v_enrg(1),v_enrg(2),..
      v_enrg(1)+v_enrg(2),betaite_time)

objVal=real(v_enrg(1)+v_enrg(2));
   endfunction

   function [fval,gval,ind,hist]=costf(t,ind);
//Computing object function value and gradient
[fval,gval]=mynumdiff(compV,t,.1^5*ones(t));
if HISTORY==%T then global hist, hist=[hist; t',fval,gval]; end
```

```
   endfunction

   function vector=NoRepeat(vec,Tf)
//Eliminating repeated values in v-grid
num_vec=size(vec,'r'); vector=[];
for i=1:num_vec
    if vec(i)~=0 & vec(i)~=Tf then
     TF=%T;
     for j=i+1:num_vec
         if vec(i)==vec(j) then TF=%F; break; end;
     end
     if TF then vector=[vector, vec(i)]; end;
    end
end;
  endfunction

function[nx,nv,ny,nnu0,nnu1,A,B,M,Qwob,Swob,Uwob,Vwob,Wwob,midRwob]..
     =combind2mdls(A0,B0,C0,D0,M0,N0,A1,B1,C1,D1,M1,N1)
//sizes of matrices
[ny,nnu0]=size(N0); nnu1=size(N1,'c');
ntilnu0=nnu0-ny;    ntilnu1=nnu1-ny;
nx0=size(A0,'c');   nx1=size(A1,'c'); nx=nx0+nx1; nv=size(B0,'c');

//constant orthogonal change of coordinates on the noise
//use a QR decomposition on N_i^T(tall) to get N_i^T=QR
//where R=[bar{N}_i^T; 0] =>
// N_i Q=[bar{N}_i, 0](fat) & new nu_i=Q^T nu_i
```

```
[Q0,R0]=qr(N0');   [Q1,R1]=qr(N1');
barN0=R0(1:ny,:)'; barN1=R1(1:ny,:)';

//change in M_i after coordi change: M_iQ Q^{-1}\nu_i
newM0=M0*Q0;           newM1=M1*Q1;
barM0=newM0(:,1:ny);   barM1=newM1(:,1:ny);
tilM0=newM0(:,ny+1:$); tilM1=newM1(:,ny+1:$);

//combined model system matrices
p=barM0*inv(barN0);
A=[A0-p*C0, p*C1; zeros(nx1,nx0), A1];
B=[B0-p*(D0-D1); B1];
M=[tilM0,p*barN1, zeros(nx0,ntilnu1); zeros(nx1,ntilnu0), newM1];

//cost weighing matrices wo/ beta
omg=inv(barN0); omg=omg'*omg; p0=C0'*omg;
p1=C1'*omg; p3=barN1'*omg;
Qwob=2*[p0*C0, -p0*C1; -p1*C0, p1*C1];
Swob=2*(D0-D1)'*omg*(D0-D1);
Uwob=2*[zeros(nx0+nx1,ntilnu0),[-p0*barN1; p1*barN1],..
     zeros(nx0+nx1,ntilnu1)];
Vwob=2*[p0*(D0-D1);-p1*(D0-D1)];
Wwob=2*[zeros(ntilnu0,nv);-p3*(D0-D1);zeros(ntilnu1,nv)];
midRwob=p3*barN1;
  endfunction

  function Vdot=fV(t,V)
```

```
//'V' vector is composed of
Jb1=matrix(V(1:nJb1),[n4xPnv,n4xPnv]);
Jb2=matrix(V(nJb1+1:nJb1+nJb2),[n2xPnv,n2xPnv]);
Jb3=matrix(V(nJb1+nJb2+1:nJb1+nJb2+nJb3),[n4xPnv,n2xPnv]);
Psi=matrix(V(nJb1+nJb2+nJb3+1:$-nPhi),[n2x,n2x]);
Phi=matrix(V($-nPhi+1:$),[n2x,nv]);

//Build Vdot vector
p1=Psi'*Omg11; p2=Phi'*Omg11; p=p2*Psi;
Jb1dot=[[p1*Psi;zeros(n2x,n2x);p], zeros(n4xPnv,n2x), ...
        [p';zeros(n2x,nv);p2*Phi]];
Jb2dot=sysdiag(zeros(n2x,n2x),Omg22);
Jb3dot=[zeros(n4xPnv,n2x),[Psi'*Omg12;zeros(n2x,nv);Phi'*Omg12]];
Psidot=Pshi*Psi;
Phidot=Pshi*Phi+B4Phi;
Vdot=[matrix(Jb1dot,[nJb1,1]);matrix(Jb2dot,[nJb2,1]);...
      matrix(Jb3dot,[nJb3,1]); matrix(Psidot,[nPsi,1]);...
      matrix(Phidot,[nPhi,1])];
  endfunction

  function J_beta=compJ(tau)
//J_beta computation
V=ode([zeros(nJb1,1);zeros(nJb2,1);zeros(nJb3,1);eye_nPsi;...
    zeros(nPhi,1)],0,tau,fV);
Pi_I=[prePi*inv([prePi0,prePi1;-matrix(V($-nPhi-nPsi+1:$-nPhi,$),..
     [n2x,n2x]),preI2x])*...
     sysdiag(preI2x,matrix(V($-nPhi+1:$,$),[n2x,nv]));preI];
```

```
Pi_IJb3=Pi_I'*..
   matrix(V(nJb1+nJb2+1:nJb1+nJb2+nJb3,$),[n4xPnv,n2xPnv]);
J_beta=.5*(Pi_I'*matrix(V(1:nJb1,$),[n4xPnv,n4xPnv])*Pi_I +matrix(...
       V(nJb1+1:nJb1+nJb2,$),[n2xPnv,n2xPnv]) + Pi_IJb3 + Pi_IJb3' );
  endfunction

  function lb = lbcalc(Del_n)
//lambda calculator
lb=max(real(spec(Del_n,scrQ)));
  endfunction

  function cost = costcalc(v)
//total cost of a signal = v'*scrQ*v
//Here assume that scrQ=diag(tau_vec(i)'s) => cost = L^2 norm of v
//v can be each element of a signal or the combinations
m=max(size(v))/ns;
cost=0;
for i=1:ns, cost=cost+tau_vec(i)*v(m*(i-1)+1:m*i)'*v(m*(i-1)+1:m*i);
end
  endfunction

function [%y0,%g]=mynumdiff(%f,%x,%dx)
//Modification of Scilab function "numdiff"
//To eliminate redundant f(x) eval, have mynumdiff to output f(x)
// given a function %f from R^n to R^p
//computes the matrix g such as
//    [   d f    ]
```

```
//    [      i   ]
//g = [   ----   ]
// ij [   d x    ]
//    [      j   ]
// using finite difference methods
if type(%f)==15 then
  params=%f;params(1)=null();
  %f=%f(1)
else
  params=list()
end
if %f==10 then //hard coded function given by its name
  error('hard coded function not allowed, create a Scilab function..
        using call""')
end

%x=%x(:);
%n=size(%x,'*')
if argn(2)<3 then
  %dx=sqrt(%eps)*(1+1d-3*abs(%x))
end
if size(params)==0 then
  %y0=%f(%x)
  %g(size(%y0,1),%n)=0
  for %j=1:%n
    %v=0*%x;%v(%j)=%dx(%j);
    %g(:,%j)=(-%y0+%f(%x+%v))/%dx(%j);
```

```
  end
else
  %y0=%f(%x,params(:))
  %g(size(%y0,1),%n)=0
  for %j=1:%n
    %v=0*%x;%v(%j)=%dx(%j);
    %g(:,%j)=(-%y0+%f(%x+%v,params(:)))/%dx(%j);
  end
end
endfunction
```

A.1.4 Functions in *funcPWC–SDS.sce*

//These functions are used by both PWC and SD algorithm

```
  function [J1,J2,J3,J4,J5,J6]=partiJ(J_beta)
//Partition{\mathcal J}
J1=J_beta(1:nx,1:nx);
J4=J_beta(1:nx,nx+1:n2x);
J5=J_beta(1:nx,n2x+1:$);
J6=J_beta(nx+1:n2x,n2x+1:$);
J3=J_beta(n2x+1:$,n2x+1:$);
J2=J_beta(nx+1:n2x,nx+1:n2x);
  endfunction


function[Del_n,Gam_n,Lam_n]=BuildDel(J1,J2,J3,J4,J5,J6,Gam_o,..
          Lam_o,Del_o)
//Delta Calculator
```

```
invLam_o=inv(Lam_o); preJ4iLamo=J4*invLam_o;
Lam_n=J1+J2-preJ4iLamo*J4';
if min(real(spec(Lam_n)))<=0 then
   error("Not p.d. Cannot use this algorithm");end
Gam_n=[-preJ4iLamo*Gam_o,J5-preJ4iLamo*J6];
preDel=Gam_o'*invLam_o*J6;
Del_n=sysdiag(Del_o,J3)-Gam_n'*inv(Lam_n)*Gam_n-
      [zeros(Del_o),preDel; preDel',J6'*invLam_o*J6];
 endfunction
```

A.2 Sampled-Data Signal Design Algorithm

A.2.1 Driver and Functions

```
clear; clc;
//Main Purpose: To compute optimal auxiliary signal for SD systems
//Assumptions: linear constant coefficient, additive noise only
//         uniform input injection/ output sampling
//User Input: model coefficients, test period,
//            sampling rate, beta grid
//Output: v*(=optimal SD signal)

//Main Solver
//Input data
//Aircraft example
A0=[-.1689 .0759 -.9952;-26.859 -2.5472 .0689; ..
     9.3603 -.1773 -2.4792];
```

```
A1=[1 1 0; 0 1 1; 0 0 1];
B0=[zeros(1,2);eye(2,2)]; B1=B0;
M0=[eye(3,3) eye(3,3)]; M1=M0;
C0=[1 0 0; 0 .9971 .0755]; C1=C0;
N0=[eye(2,2)]; N1=N0;
P0=eye(A0);P1=P0;Q0=[];Q1=Q0;U0=Q0;U1=Q0;//Cost=[];
//3(.0001): example 3.3.1
A0=[0,-1;1,0];A1=[0,3;-3,0];B0=eye(2,2);B1=B0;
M0=.0001*[eye(2,2)];M1=M0;
C0=[1,2];C1=C0;N0=[1]; N1=N0;
P0=eye(A0);P1=P0;
sr=5; //sampling rate

//a. Load functions
exec funcPWC_SDS.sce;
exec funcSDS.sce;

//b. Uniform sampling rate and beta search grid
smpl_rate=sr; test_period=5; T=test_period;
num_sampling=T*smpl_rate; ns=num_sampling;
tau=T/ns; tt=0:tau:T; //time
beta_grid=50; bg=beta_grid;
adj=1/(bg*5); bb=linspace(adj,1-adj,bg+1);

//c. Size of state, signal, mu and nu uncertainty vectors
[nx0,nnu0]=size(M0); [nx1,nnu1]=size(M1);
[ny,nmu0]=size(N0); nmu1=size(N1,2); nv=size(B0,2);
```

```
nx=nx0+nx1; nnu=nnu0+nnu1; nmu=nmu0+nmu1;
n2x=2*nx; n2xPnv=n2x+nv; n4xPnv=n2xPnv+n2x;
nJb= n4xPnv^2; nPsi= n2x^2; nPhi= n2x*nv;//total # of Jb,Psi,Phi

//d. Default & pre-build matrices
preIx=eye(nx,nx); preIv=eye(nv,nv);
pre0x=zeros(nx,nx); pre0v=zeros(nv,nv);
preI2x=eye(n2x,n2x); pre02x=zeros(n2x,n2x); pre0v2x=zeros(nv,n2x);
eye_nPsi=matrix(preI2x,[nPsi,1]);
prePi=[preI2x,pre02x];
prePi0=[zeros(nx,n2x);preIx,pre0x];
prePi1=[preIx,pre0x;zeros(nx,n2x)];

//e. Build system matrices
[A,B,C,M,N]=build_mat(A0,B0,C0,M0,N0,A1,B1,C1,M1,N1);

//f. Find lambda_beta at every beta (0,1)
lam_b=[];lam_s=-1; Del_s=[]; beta_s=-1; t=tic()
for bt=bb,
  //default & pre-build beta related matrix
  Pb=sysdiag(bt*P0,(1-bt)*P1); //dropping the inverse notation
  preJb=0.5*M*sysdiag(1/bt*eye(nnu0,nnu0),..
        1/(1-bt)*eye(nnu1,nnu1))*M';
  preLam=tau*C'*inv(N*sysdiag(1/bt*eye(nmu0,nmu0),1/(1-bt)*..
     eye(nmu1,nmu1))*N')*C; //=tau*C'*inv(N*inv(Ub)*N')*C;
  Pshi=[A,-preJb; pre0x,-A'];
```

```
//Form {mathcal J_beta}. Given assumptions,
//J_beta is same for any [t_j, t_{j+1}].
%ODEOPTIONS=[1,0,0,tau/10,0,2,2000,12,5,0,-1,-1];
V=ode( [zeros(nJb,1);eye_nPsi;zeros(nPhi,1)],0,tau,fV);
Pi=prePi*..
   inv([prePi0,prePi1;-matrix(V(nJb+1:nJb+nPsi,$),[n2x,n2x]),..
   preI2x])*sysdiag(preI2x,matrix(V($-nPhi+1:$,$),[n2x,nv]));
J_beta=..
   [Pi' eye(n2xPnv,n2xPnv)]*matrix(V(1:nJb,$),[n4xPnv,n4xPnv])*..
   [Pi; eye(n2xPnv,n2xPnv)];
//Partition J_beta to J_#s
[J1,J2,J3,J4,J5,J6]=partiJ(J_beta);
calX=(J1+preLam);

Del_o=[];Gam_o=[];
for t=tt(1:$-1),  //Build Delta
  if t==0 then Lam_o=Pb+J2;
    if min(real(spec(Lam_o)))<=0 then
    error("Not p.d. Cannot use this algorithm"); end;
  else //Build Lamb, chk if {\mathcal X_beta}(n)>0 & Build Delta
    [Del_n,Gam_n,Lam_n]=..
         BuildDel(J1,J2,J3,J4,J5,J6,Gam_o,Lam_o,Del_o);
    Lam_o=Lam_n; Gam_o=Gam_n; Del_o=Del_n;
  end
end //time
J2=zeros(J1);
Del_n=BuildDel(J1,J2,J3,J4,J5,J6,Gam_o,Lam_o,Del_o);
```

```
  lam=lbcalc(Del_n);
  if lam>lam_s then
  lam_s=lam; Del_s=Del_n; beta_s=bt; end //lam*/beta*/Del*
  lam_b=[lam_b,lam];
end //beta
betaite_time=toc();

//g. Plot of the lambda_beta function
if lam_s == 0 then error('no proper auxiliary signal exists'); end
xset('window',0); xtitle('','beta','lambda');
a=gca();a.thickness=2; a.line_style=2;
plot2d(bb,lam_b)

//h. v_n^*
[al,be,eigvec]=spec(Del_s,tau*eye(ns*nv, ns*nv));
v=eigvec(:,find(lam_s==real(al./be)));

//i. Normalize v
v_s=v/sqrt(lam_s*tau*v'*v); //total cost=v'*Cost*v=tau*v'*v

//j. Compute total energy of signals used
v_enrg=[]; plt=[];
for i=1:nv
  //xbasc(i+ns);
  xset('window',i); //xtitle('','time','signal v');
  v_sig=v_s(i:nv:$);
  v_enrg=[v_enrg, tau*v_sig'*v_sig];
```

```
  a=gca(); a.thickness=4; a.line_style=8;
  plot2d2(tt(1:$)',-[v_sig;v_sig($)]);
end

//k. Output
//result_str=msprintf("%d&%d&%1.4f&%1.4f&%1.4f&%1.4f&%1.4f&%1.3f&..
      %1.4f&%1.4f&%1.4f\\",sr,T,beta_s,lam_s,v_enrg(1),v_enrg(2),..
      v_enrg(1)+v_enrg(2),betaite_time,nrm_sup, nrm_L1, nrm_L2 );
```

A.3 Optimal Analog Signal Design Algorithm

A.3.1 Driver

```
  lines(0)
  result_cs=[]; //store results
  // Input data & weights.
  W_G=0; W_H=0; //weights on G & H
  // weight on uncertainty(J)
  //J_W=1 (model uncertainty) J_W=-1 (additive uncertainty)
  J_W=-1;

  //Aircraft example
  A0=[-.1689 .0759 -.9952;-26.859 -2.5472 .0689;..
        9.3603 -.1773 -2.4792];
  A1=[1 1 0; 0 1 1; 0 0 1];
  B0=[zeros(1,2);eye(2,2)]; B1=B0;
  M0=[eye(3,3) eye(3,3) zeros(3,2)]; M1=M0;
```

```
G0=W_G*[.01 .001 .01; 4.13 .1 .001; 1 .001 .1; zeros(3,3)]; G_0=[];
G1=W_G*[.01 .1 .01; 0 .1 .1; 0 0 .1; zeros(3,3)];G_1=[];
H0=W_H*[zeros(4,2);.1*eye(2,2)];H_0=[];
H1=W_H*[zeros(4,2);.11*eye(2,2)];H_1=[];
C0=[1 0 0; 0 .9971 .0755]; C1=C0;
D0=zeros(C0*B0); D1=D0; N0=[zeros(2,6) eye(2,2)]; N1=N0;
P0=eye(A0);P1=P0;

// Period of test(T) and beta search grid
T=10; //assumes t0=0
beta_grid=50; bg=beta_grid;
adj=1/(bg*5); bb=linspace(adj,1-adj,bg+1);
t_from= adj; t_to=1- adj;

// Size of state, signal, mu and nu uncertainty vectors
nx0=size(A0,2); nx1=size(A1,2); nv=size(B0,2);
nnu0=size(M0,2); nnu1=size(M1,2);
ny=size(C0,1); nz0=size(G0,1); nz1=size(G1,1);
nx=nx0+nx1; nzy0=ny+nz0; nzy1=ny+nz1;

// Defintion of J, xi
J0=sysdiag(eye(nnu0,nnu0),-J_W*eye(nz0,nz0));
J1=sysdiag(eye(nnu1,nnu1),-J_W*eye(nz1,nz1));
nf=nx0; b=0; Om=b*eye(nf,nf);Ga=b*eye(nf,nf);
F=zeros(A0);G=zeros(B0); //F=rand(nf,nf);G=rand(nf,nv);

// Build system matrices
```

```
[A,B,C,D,M,N]=datas(A0,B0,C0,D0,G0,H0,M0,N0,...
                                    A1,B1,C1,D1,G1,H1,M1,N1);

// Find lambda_beta on a beta-grid in (0,1)
lb=[];t=tic()
for bt=bb,
lb=[lb lbcalc(bt,P0,P1,J0,J1,A,B,C,D,M,N,F,G,Ga,Om)]; end
betaite_time=toc(); [lam,i]=max(lb); bt=bb(i);

// Plot of the lambda_beta function
xbasc(0);
xset('window',0); xtitle('','beta','lambda');
a=gca(); a.thickness=2; a.line_style=0;
plot2d(bb,lb,1); //xstring(bb(i),lb(i),);

// Find optimal signals
grid=100; //TT=linspace(0,T,grid*T+1);
[AA,V0,VT,B,C,D,Gx,S,R]=rio(lam,bt);
[TT,xx]=tpbvs(AA,V0,VT,T,grid*T+1);
nuv=-inv(Gx)*[D'*inv(R)*C,D'*inv(R)*S'-B']*xx;

// Calculate scaling factor(alpha)
tau=1/grid; //assumes uniform time interval
cost=delta(); alpha=1/(sqrt(cost*lam));

// Compute cost=delta(v)
nuv($-nv+1:$,:)=alpha*nuv($-nv+1:$,:);
```

```
  cost=delta();
  signal_1=tau*nuv($-nv+1,1:$-1)*nuv($-nv+1,1:$-1)';
  signal_2=tau*nuv($,1:$-1)*nuv($,1:$-1)';

  // Plot v_j(t)
  a=gca();a.thickness=2; a.line_style=0;
  for i=1:nv
    //xbasc(i);
    xset('window',i); xtitle('','time','signal v');
    a=gca();a.thickness=2; a.line_style=0;
    v_sig=nuv($-nv+i,:)';
    plot2d(TT',v_sig);  end

// Output
result_str=msprintf("%1.4f&%1.4f&[%1.2f, %1.2f]&%d&%1.4f&%1.4f&...
     %1.4f&%1.4f&%1.4f&%1.3f",W_G,W_H,t_from,t_to,bg,bt,lam,..
     signal_1,signal_2,cost,betaite_time)
result_cs=[result_cs;result_str];
```

A.3.2 Functions

Programmed by R. Nikoukhah except *v, fV, delta* functions

```
clear; clc;
  %ODEOPTIONS=[1,0,0,%inf,0,2,1000,12,5,0,-1,-1];

function [A,B,C,D,M,N]=..
    datas(A0,B0,C0,D0,G0,H0,M0,N0,A1,B1,C1,D1,G1,H1,M1,N1);
```

```
  [ny,nnu0]=size(N0); [ny,nnu1]=size(N1);
  [nz1,nx1]=size(G1); [nz0,nx0]=size(G0);
  E0=[zeros(nz0,ny);eye(ny,ny)]; E1=[zeros(nz1,ny);eye(ny,ny)];
  M0=[M0,zeros(nx0,nz0)]; M1=[M1,zeros(nx1,nz1)];
  N0=[zeros(nz0,nnu0),-eye(nz0,nz0);N0,zeros(ny,nz0)];
  N1=[zeros(nz1,nnu1),-eye(nz1,nz1);N1,zeros(ny,nz1)];
  C0=[G0;C0];C1=[G1;C1];  D0=[H0;D0];D1=[H1;D1];
  F=kernel([E0;E1]')';F0=F(:,1:nz0+ny);F1=F(:,1+nz0+ny:$);
  F0=eye(F0); F1=-eye(F1);//<-to have the same matrices as SDS
  B=[B0;B1];A=sysdiag(A0,A1);M=sysdiag(M0,M1);
  C=[F0*C0,F1*C1];D=F0*D0+F1*D1;N=[F0*N0,F1*N1];
endfunction

function [T,xx]=tpbvs(A,V0,VT,T,nb)
  if argn(1)>1 then T=T+2;end;
  [A ,X ,bs]=bdiag(A); j=1; sp=[];
  for i=bs'
    sp=[sp,min(real(spec(A(j:j+i-1,j:j+i-1))))*ones(1,i)];
    j=j+i;
  end
  n=size(A,2);
  [sp,I]=gsort(sp);
  A=A(I,I); X=X(1:n,I); V0=V0*X;VT=VT*X; h=max(find(~(sp<0)));
  Ab=A(1:h,1:h);Af=A(h+1:$,h+1:$);II=eye(Ab);JJ=eye(Af);
  [PPh,rd]=ode('root',[II(:);JJ(:)],0,T,sys,1,sysr);
  if rd<>[] then T=rd(1); end
  if argn(1)>1 then
```

```
    if argn(2)<5 then nb=100,end
    phb=matrix(PPh(1:h^2),h,h);
    phf=matrix(PPh(h^2+1:$),n-h,n-h);
    x0T=kernel([sysdiag(eye(Ab),-phf),sysdiag(-phb,eye(Af));V0,VT]);
    TT=linspace(0,T,nb); xbT=x0T(n+1:n+h); xf0=x0T(h+1:n);
    Xb=ode(xbT,T,TT($:-1:1),sysb);
    Xf=ode(xf0,0,TT,sysf);
    xx=X*[Xb(:,$:-1:1);Xf];T=TT;pause
  end
endfunction

function xd=sysb(t,x)
  xd=Ab*x
endfunction

function xd=sysf(t,x)
  xd=Af*x
endfunction

function phd=sys(t,Ph)
  phb=matrix(Ph(1:h^2),h,h);
  phf=matrix(Ph(h^2+1:$),n-h,n-h);
  phbd=-Ab*phb;  phfd=Af*phf;
  phd=[phbd(:);phfd(:)];
endfunction

function r=sysr(t,Ph)
```

```
  phb=matrix(Ph(1:h^2),h,h);
  phf=matrix(Ph(h^2+1:$),n-h,n-h);
  r=det([sysdiag(eye(Ab),-phf),sysdiag(-phb,eye(Af));V0,VT]);
endfunction

function [AA,V0,VT,B,C,D,Gx,S,R]=rio(lam,bt)
  Pb=sysdiag(P0/bt,P1/(1-bt));
  Jb=sysdiag(bt*J0,(1-bt)*J1);
  T=[M,B;N,D]*inv(sysdiag(Jb,-lam*eye(nv,nv)))*[M,B;N,D]';
  Q=T(1:nx,1:nx);
  S=T(1:nx,nx+1:$);
  R=T(nx+1:$,nx+1:$);
  No=kernel(N);
  if min(real(spec(No'*Jb*No))) <= 0 then
      error('test 1 failed');end
  D=[N,D];C=[C,zeros(ny+nz1+nz0,nf)];
  A=sysdiag(A,F);B=[M,B;zeros(nf,nnu0+nnu1+nz0+nz1),G];
  Gx=sysdiag(Jb,-lam*eye(nv,nv));
  T=[B;D]*inv(Gx)*[B;D]';
  Q=T(1:nx+nf,1:nx+nf);
  S=T(1:nx+nf,nx+nf+1:$);
  R=T(nx+nf+1:$,nx+nf+1:$);
  AA=[A-S*inv(R)*C,Q-S*inv(R)*S';
      C'*inv(R)*C+sysdiag(zeros(nx,nx),-lam*Ga),-(A-S*inv(R)*C)'];
  V0=[inv(Pb),zeros(nx,nf),-eye(nx,nx),zeros(nx,nf);
    zeros(nf,nx),eye(nf,nf),zeros(nf,nx+nf);
    zeros(nx+nf,2*nx+2*nf)];
```

```
    VT=[zeros(nx+nf,2*nx+2*nf);
    zeros(nx,nx+nf),eye(nx,nx),zeros(nx,nf);
    zeros(nf,nx),-lam*Om,zeros(nf,nx),eye(nf,nf)];
endfunction

function lam=lbcalc(bt,P0,P1,J0,J1,A,B,C,D,M,N,F,G,Ga,Om,prec)
   printf('Computing lambda for beta=%f\n',bt);
  [lhs,rhs ]=argn()
  if rhs==15 then prec=.01;end
  lmin=0.00001;lmax=100;
  [AA,V0,VT]=rio(lmax,bt);
    t=tpbvs(AA,V0,VT,T)
  if t<T then lam=0;printf('does not work '),  return;end
  while lmax-lmin>prec*lmin
    lam=(lmax+lmin)/2; //printf('Trying lambda=%f',lam)
    [AA,V0,VT]=rio(lam,bt);
    t=tpbvs(AA,V0,VT,T);
    if t<T  then
      lmin=lam,//printf(' --too small, I only got to %f\n",t)
    else
      lmax=lam,//printf(' --I can do better, I got to  %f\n",t)
    end
  end
  lam=lmin,//printf('I take lambda=%f\n',lam);
endfunction

function vt=v(t)
```

```
i_cur=int(t/tau)+1;
if t<T-tau then //v(ti)+(t-ti)*(v(t{i+1})-v(ti))=
vt=(1-t+TT(i_cur))*nuv($-nv+1:$,i_cur)+(t-TT(i_cur))*..
    nuv($-nv+1:$,i_cur+1);
else vt=nuv($-nv+1:$,$); end
endfunction

function Vdot=fV(t,V,v) //Ga=W, Om=U
z=V(1,1); xi=V(2:1+nf,1);
zdot=v(t)'*v(t)+xi'*Om*xi; xidot=F*xi+G*v(t);
Vdot=[zdot;xidot];
endfunction

function cost=delta()
cst=ode(zeros(1+nf,1),0,TT,list(fV,v));
cost=cst(1,$)+xx(nx+1:nx+nf,$)'*Ga*xx(nx+1:nx+nf,$);
endfunction
```

List of References

[1] M.F. Abbod and D.A. Linkens. Fault diagnosis for closed-loop drug infusion. In *IEE Colloquium on Condition Monitoring Machinery, External Structures and Health*, 1999.

[2] K. Akouz and E-K Boukas. H_∞ based fault detection and isolation for markovian jump systems. In *16th IFAC World Congress, Prague, Czech Republic*, July 2005.

[3] S. Armeni, A. Casavola, and E. Mosca. Robust fault detection and isolation filters design with sensitivity constraint for lpv systems. In *16th IFAC World Congress, Prague, Czech Republic*, July 2005.

[4] D. J. Atkinson, M. L. James, D. Lawson, R. G. Martin, and H. Porta. Automated spacecraft monitoring. In *Proceedings of IEEE Int'l Conf. Systems, Man and Cybernetics*, 1990.

[5] M. Basseville and I. V. Nikiforov. *Detection of abrupt changes : theory and application*. Prentice Hall, 1993.

[6] D. P. Bertsekas and I. B. Rhodes. Recursive state estimation for a set-membership description of uncertainty. *IEEE Transactions on Automatic Control*, 16(2):117–128, 1971.

[7] J. T. Betts and W. P. Huffman. Sparse optimal control software socs. *Mathematics and Engineering Analysis Technical Document MEA-LR-085, Boeing Information and Support Services*, 1997.

[8] L. Blackmore and B. Williams. Finite horizon control design for optimal model discrimination. In *Proc. of the Conf. on Decision and Control and the European Control Conf.*, 2005.

[9] J. F. Bonnans. A variant of a projected variable metric method for bound constrained optimization problems. *Institut National de Recherche en Informatique, Report*, 242, 1983.

[10] A. E. Bryson, W. F. Denha, and S. E. Dreyf. Optimal programming problems with inequality constraints i: Necessary conditions for extremal solutions. *AIAA Journal*, 1(1):2544–2550, 1963.

[11] S. L. Campbell, J. P. Chancelier, and R. Nikoukhah. *Modeling and Simulation in Scilab/Scicos*. Springer, 2005.

[12] S. L. Campbell, K. J. Drake, I. Andjelkovic, K. Sweetingham, and D. Choe. Model based failure detection using test signals from linearizations: A case study. In *Proceedings of IEEE Int. Conf Control Applications*, 2006.

[13] S. L. Campbell and R. Nikoukhah. *Auxiliary signal design for failure detection*. Princeton University Press, Princeton, NJ, 2004.

[14] J. P. Chancelier, F. Delebecque, C. Gomez, M. Goursat, R. Nikoukhah, and S. Steer. *An introduction to Scilab*. Springer-Verlag, 2002.

[15] D. Choe, S.L. Campbell, and R. Nikoukhah. Comparison between failure detection test signals for continuous systems and sampled-data systems. In *Proc. of the Conf. on Decision and Control*, 2006.

[16] J. W. Demmel. *Applied Numerical Linear Algebra*. SIAM, Philadelphia, 1997.

[17] K. J. Drake. *Analysis of Numerical Methods for Fault Detection and Model Identification in Linear Systems with Delays. PhD thesis*. North Carolina State University, 2003.

[18] J. Gertler. *Fault Detection and Diagnosis in Engineering Systems*. Marcel Dekker, Inc., New York, 1998.

[19] A. C. Hindmarsh. Odepack, a systematized collection of ode solvers. *IMACS Transactions on Scientific Computation*, 1:55–64, 1983.

[20] K. Horton. *Fault Detection And Model Identification In Linear Dynamical Systems. PhD thesis*. North Carolina State University, 2001.

[21] C. P. i Sabadi. Case based reasoning as an extension of fault dictionary methods for linear electronic analog circuits diagnosis. In *Ph.D Dissertation*, 2004.

[22] A. Ingimundarson, J. M. Bravo, V. Puig, and T. Alamo. Robust fault diagnosis using parallelotope-based set-membership consistency tests. In *Proceedings of IEEE Conf. Decision and Control and European Control Conference*, 2005.

[23] R. Iserman. *Fault-Diagnosis Systems*. Springer, Heidelberg, 2006.

[24] D. H. Jacobson, M. M. Lele, and J. L. Speyer. New necessary conditions of optimality for control problems with state-variable inequality constraints. *Journal of Mathematical Analysis and Applications*, 35(2):255–284, 1971.

[25] B. N. Jain. Guaranteed error estimation in uncertain systems. *IEEE Transactions on Automatic Control*, 20(2):230 – 232, 1975.

[26] R. E. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME - Journal of Basic Engineering*, 82(D):35–45, 1960.

[27] T. Kato. *Perturbation Theory for Linear Operators*. Springer, 1995.

[28] C. T. Kelley. *Iterative Methods for Optimization*. SIAM, 1999.

[29] F. Keresteciogl u. *Change detection and input design in dynamical systems*. Research Studies Press, Baldock, Hertfordshire, 1993.

[30] D. E. Kirk. *Optimal Control Theory: An Introduction*. Dover publications, 1970.

[31] E. Kreindler. Additional necessary conditions for optimal control with state-variable inequality constraints. *Journal of Optimization Theory and Applications*, 38(2):241–250, 1982.

[32] F. L. Lewis and V. L. Syrmos. *Optimal Control*. John Wiley & Sons, Inc, 1995.

[33] C. D. Meyer. *Matrix Analysis and Applied Linear Algebra*. SIAM, New York, 2000.

[34] G. Miller and A. Pankov. Linear-quadratic stochastic optimal control problem with incomplete information and uncertain noise statistics. In *Proceedings of the 45th IEEE Conference on Decision and Control*, 2006.

[35] C. Moler and C. Van Loan. Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later. *SIAM Review*, 45(1):3–49, 2003.

[36] C. B. Moler and G. W. Stewart. An algorithm for generalized matrix eigenvalue problems. *SIAM Journal on Numerical Analysis*, 10(2):241–256, 1973.

[37] O. Moseler and R. Isermann. Application of model-based fault detection to a brushless dc motor. *IEEE Transactions on Industrial Electronics*, 47(5):1015–1020, 2000.

[38] H. H. Niemann and J. Stoustrup. Filter design for failure detection and isolation in the presence of modeling errors and disturbances. In *Proceedings of IEEE Conf. Decision and Control*, 1996.

[39] R. Nikoukhah. Guaranteed active failure detection and isolation for linear dynamical systems. *Automatica*, 34(11):1345–1358, 1998.

[40] R. Nikoukhah and S. L. Campbell. Robust detection of incipient faults: an active approach. In *Proceedings of IEEE Med. Conf. Control and Automation*, 2006.

[41] R. Nikoukhah and S.L. Campbell. A multi-model approach to failure detection in uncertain sampled-data systems. *European Journal of Control*, 11(3):255–266, 2005.

[42] R. C. Osborne, R. J. Adams, C. S. Hsu, and S. S. Banda. Reduced-order h, compensator design for an aircraft control problem. *AIAA Journal of Guidance, Control, and Dynamics*, 17(2):341–345, 1994.

[43] I. R. Petersen and D. McFarlane. Optimal guaranteed cost control and filtering for uncertain linear systems. *IEEE Transactions on Automatic Control*, 39(9):1971–1977I, 1994.

[44] E. Polak. *Optimization: Algorithms and Consistent Approximations*. Springer, New York, 1991.

[45] A. Quarteroni, R. Sacco, and F. Saleri. *Numerical Mathematics*. Springer, New York, 2000.

[46] A. H. Sayed. A framework for state-space estimation with uncertain models. *IEEE Transactions on Automatic Control*, 46(7):998–1013, 2001.

[47] M. Simandl, I. Puncochar, and J. Kralovec. Rolling horizon for active fault detection. In *Proc. of Decision and Control and European Control Conf.*, 2005.

[48] J. S. Yang. Mixed H^2 compensator design for an aircraft control problem. In *Proc. of the Conf. on Decision and Control*, pages 1964–1969, 1999.

[49] X. J. Zhang. *Auxiliary Signal Design in Fault Detection and Diagnosis*. Springer, Heidelberg, 1989.

Index