APPLICATION OF DETECTION FILTER THEORY TO LONGITUDINAL CONTROL OF GUIDEWAY VEHICLES

by

Jean-Pierre Augustin Gerard

Ingenieur, Ecole Centrale de Paris

(1977)

SUBMITTED IN PARTIAL FULFILLMENT

OF THE REQUIREMENTS FOR THE

DEGREE OF MASTER OF SCIENCE

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June, 1978

Signature of Author______________________
Department of
Aeronautics and Astronautics

Certified by______________________
Thesis Supervisor

Accepted by______________________
Chairman
Departmental Graduate Committee

APPLICATION OF DETECTION FILTER THEORY TO LONGITUDINAL CONTROL OF GUIDEWAY VEHICLES

by

Jean-Pierre Augustin Gerard

Submitted to the Department of
Aeronautics and Astronautics on June 30, 1978
in partial fulfillment of the requirements for
the degree of Master of Science

ABSTRACT

This paper recalls briefly the main results of the detection filter theory, which, through sophisticated data processing, allows in certain circumstances to detect and identify component failures in a system, by assigning unidirectional or bidirectional error outputs to each failure. The algorithm of a computer program developed to help the design of a detection filter is then detailed. An application of it in the content of longitudinal control of a guideway vehicle was then made to investigate what practical results could be expected.

Thesis Supervisor: Wallace E. VanderVelde
Title: Professor of Aeronautics and Astronautics

ACKNOWLEDGEMENT

I want to express my gratitude to Professor W.E. VanderVelde for his guidance and good advice all along during this study. I enjoyed working under his direction. Special thanks go to Michael Dyment, a fellow graduate student, who did not hesitate to work long hours before he left to make sure that the interface between his guideway vehicle simulation program and a reference model program worked, and that I understood how to use it. Last, but not least, I thank Mrs. Barbara Marks for her patience in the typing of my difficult manuscript.

My financial support was a fellowship of Jean Gaillard Memorial Foundation during the school year, and a research assistantship in June.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

With the decreasing price trend in computation capability and the increasing price trend in hardware components, it will make more and more economic sense to try to achieve high levels of reliability in control systems with less redundancy in material parts, at the expense of more computation. Thus, even on guideway vehicles where weight is not a dominant problem, sophisticated data processing can be envisioned as a way to achieve the required reliability of the longitudinal control system.

For a complex system without any redundancy, a failure in any element entails a failure of the whole system. Redundancy, on the contrary, allows certain component failures without preventing the system as a whole to function. One of the simplest kinds of redundancy is what might be called "standby redundancy." Two or more components which perform the same functions are set in parallel, and in case of failure of the operating component, the system switches to the backup one. The problem is to know which component of the chain is faulting, when the system as a whole fails. This can be done by majority rule with three components in parallel. A detection filter, under certain circumstances, can detect which component is faulting and requires then only two components in parallel. There are some other advantages in the use of detection filters, not discussed in this paper, such as their use as suboptimal filters which could provide partial state estimation for failed systems.

The detection filter theory was first presented by Beard (ref. 1) and was further developed by Jones (ref. 2). This study was made to investigate whether detection filters would be of practical value in longitudinal control of guideway vehicles: detection filter theory assumes a linear time invariant model, which is not the case for a guideway vehicle, subjected to non linear forces such as the aerodynamic force. Furthermore, some inputs of the real system would be difficult to indicate, such as the grade of the track, and some components of the control system would be noisy. To assess the relative values of these effects compared to the effects of failures in the system, a simulation of a representative vehicle was set up and tests were made to evaluate the practical value of a detection filter processing the difference between a real system output and a simplified linearized reference model output. The detection filter was designed with a computer program which was developed, applying algorithms derived from reference 2.

Chapter 2 recalls the theoretical results necessary to understand the application, chapter 3 details the algorithm of the detection filter design program, chapter 4 presents the guideway vehicle simulation, the reference model for the detection filter, and the detection filter which was computed, chapter 5 gives the experimental results, and chapter 6 states the conclusions. Appendix A gives the listing of the detection filter design program, Appendix B the listing of the guideway vehicle simulation together with the reference model simulation, Appendix C presents some problems met in the practical design, and Appendix D the orthogonal reduction procedure used repeatedly in the algorithm.

CHAPTER 2

THEORETICAL BACKGROUND

The concept of the detection filter was first presented by Beard (ref. 1) and was further explored by Jones (ref. 2). Only a brief summary is given here. The aim of this chapter is just to present the results necessary to understand the application; for more details see reference 1 and reference 2.

The detection filter theory assumes that the plant dynamics can be represented by a linear, time invariant set of differential equations. Let $\underline{x}$ be the state vector (n dimensioned)

$\underline{u}$ be the input vector (q dimensioned)

$\underline{y}$ be the output vector (m dimensioned)

The system equations are

$$\dot{\underline{x}} = A\,\underline{x} + B\,\underline{u}$$

$$\underline{y} = C\,\underline{x}$$

where

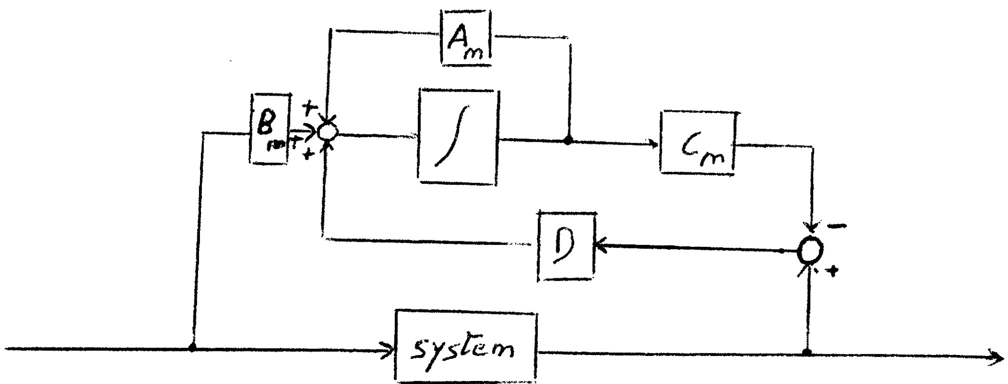
A is a n x n matrix

B is a n x q matrix

C is a m x n matrix

From the comparison between y, the output of the system measured by means of sensors, and $\underline{y}_m$, the output of a simulation run in parallel (real time) with the system, with the same equations, A, B, C having their unfailed values, the detection filter theory allows to detect which component of the system has failed, under certain circumstances.

The general reference model is the following:



General Reference Model

Equations are

$$\begin{cases} \dot{\underline{x}} = A\,\underline{x} + B\,\underline{u} & \\ \underline{y} = C\,\underline{x} & \underline{x}(0) \text{ given} \end{cases}$$

$$\begin{cases} \dot{\underline{x}}_m = A_m\,\underline{x}_m + B_m\,\underline{u} + D(\underline{y} - \underline{y}_m) & \\ \underline{y}_m = C_m\,\underline{x}_m & \underline{x}_m(0) \text{ given} \end{cases}$$

($\underline{x}_m(0)$ is necessary to start the simulation, but as (A - DC) is designed to have no positive real part eigenvalues, $\underline{x}_m(0)$ does not need to be equal to $\underline{x}(0)$ for $\underline{x}(t)$ and $\underline{x}_m(t)$ to be equal in steady state).

In the absence of failure $A_m \equiv A$ $B_m \equiv B$ $C_m \equiv C$.

Introducing the error vector $\underline{\varepsilon} = \underline{x} - \underline{x}_m$ we have

$$\dot{\underline{x}} - \dot{\underline{x}}_m = A(\underline{x} - \underline{x}_m) - D(\underline{y} - \underline{y}_m)$$

$$= A(\underline{x} - \underline{x}_m) - DC\,(\underline{x} - \underline{x}_m)$$

$$\dot{\underline{\varepsilon}} = (A - DC)\,\underline{\varepsilon} \qquad \underline{\varepsilon}(0) = \underline{x}(0) - \underline{x}_m(0)$$

The output error $\underline{\varepsilon}' = \underline{y} - \underline{y}_m$ then follows the equation

$$\underline{\varepsilon}' = C\,\underline{\varepsilon}\,.$$

In the event of a failure in the physical system, some values of parameters in A, B, C change (may become time varying). The usefulness of the detection filter theory comes from the fact that very often failures can be modelized according to one of the two following ways:

- <u>controller failure model</u>

$$\dot{\underline{x}} = A\,\underline{x} + B\,\underline{u} + \underline{b}_i\, n_i(t) \qquad \underline{y} = C\,\underline{x}$$

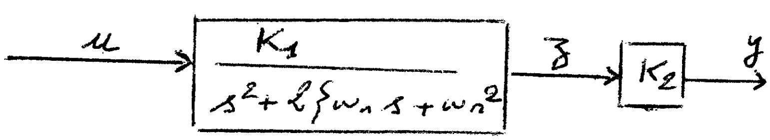
- <u>sensor failure model</u>

$$\dot{\underline{x}} = A\,\underline{x} + B\,\underline{u} \qquad \underline{y} = C\,\underline{x} + \underline{e}_{m_i}\, n_{ci}(t)$$

where $\underline{b}_i$ and $\underline{e}_{m_i}$ are two time-invarying vectors, even if the failures introduce time variations in the matrices.

<u>Example</u>

Consider the simple case below



Equation $\dfrac{z}{u} = \dfrac{K_1}{s^2 + 2\zeta\omega_n s + \omega_n^2} \iff \ddot{z} + 2\zeta\omega_n \dot{z} + \omega_n^2 z = K_1 u$

Using the phase variable $x_1 = z$, $x_2 = \dot{z}$, we have the state equations

$$\begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -\omega_n^2 & -2\xi\omega_n \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} 0 \\ K_1 \end{pmatrix} u$$

$$y = \begin{pmatrix} K_2 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

If K_1 fails and becomes $K_1 \cdot k_1(t)$, the model becomes

$$\begin{cases} \dot{\underline{x}} = A\,\underline{x} + B\,\underline{u} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} (k_1(t) - 1)\, K_1\, u \\ \underline{y} = C\,\underline{x} \end{cases}$$

in this case $\underline{b}_1 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$, $n_i(t) = [k_1(t) - 1]\, K_1\, u$

If K_2 fails and becomes $K_2 \cdot k_2(t)$, the model becomes

$$\begin{cases} \dot{\underline{x}} = A\,\underline{x} + B\,\underline{u} \\ \underline{y} = C\,\underline{x} + (1)\,(K_2)\,[k_2(t) - 1]\, x_1 \end{cases}$$

In this case $\underline{e}_{m_i} = 1$ $\quad n_{ci}(t) = K_2\, [k_2(t) - 1]\, x_1$

In case of failure, the error differential equations become then:

- <u>controller failure model</u> $\quad \dot{\underline{\varepsilon}} = (A - DC)\,\underline{\varepsilon} + \underline{b}_i\, n_i(t)$

$$\underline{\varepsilon}' = C\,\underline{\varepsilon}$$

- <u>sensor failure model</u> $\quad \dot{\underline{\varepsilon}} = (A - DC)\,\underline{\varepsilon} - \underline{d}_i\, n_{c_i}(t)$

$$\underline{\varepsilon}' = C\,\underline{\varepsilon} + \underline{e}_{m_i}\, n_{c_i}(t)$$

where $\underline{d}_i$ is the ith column of the matrix D.

The detection filter theory allows the detection of the faulting component of a system because of three features: under certain circumstances, it allows to find a D such that:

a - eigenvalues of A - DC can be almost arbitrarily assignable

b - outputs associated with a $\underline{b}_i$ (controller failure model) can be constrained to be unidirectional

c - outputs associated with a n_{c_i} (sensor failure model) can be constrained to a plane.

More precisely: definition (Beard)

The event associated with the vector $\underline{b}$ is detectable if there exists a matrix D such that

(1) $C\underline{\varepsilon}$ maintains a fixed direction in the output space, where $\underline{\varepsilon}(t)$ is the settled out solution

(2) at the same time all eigenvalues of A - DC can be specified almost arbitrarily.

It can be shown that the solution to $\dot{\underline{\varepsilon}} = (A - DC)\,\underline{\varepsilon} + \underline{b}\,n(t)$ with A - DC matrix negative definite is, after the vanishing of the transient terms

$$\underline{\varepsilon}(t) = \int_{t_0}^{t} \exp\left[-(A-DC)(t-\tau)\right] \underline{b}\, n(\tau)\, d\tau$$

and this solution lies in the space spanned by the columns of $W_b = [\underline{b}, (A - DC)\,\underline{b}, \ldots, (A - DC)^{n-1}\,\underline{b}]$.

That is to say, $\underline{\varepsilon}$ may be expressed in the form

$\underline{\varepsilon}(t) = W_b \underline{g}(t)$ $\qquad\qquad$ $\underline{g}(t)$ depending on $n(t)$

Then

$$C\,\underline{\varepsilon}(t) = C\,W_b\,\underline{g}(t)$$

Beard showed that condition (1) of the definition is equivalent to the fact that $C\,W_b$ has rank 1.

The most useful property of direction filters, however, is their property, under certain circumstances, to detect without ambiguity several failures at a time. If there are n independent sensors in a n-order plant, it seems intuitive, and it can be shown, that no ambiguity is left in monitoring failures of the plant. If there is only one sensor, on the contrary, it seems obvious that no discrimination between eventual failures can be performed. The following concepts are necessary to understand how the case where there are less than n independent sensors is handled.

° null space: the null space of an operator A is the largest subspace of the space where A is defined, whose image under A is the zero space. It is denoted $\eta(A)$.

° detection equivalent events (Jones)

Two events $\underline{b}_1$ and $\underline{b}_2$ associated with failures in a system are said to be detection equivalent if

(a) every detection filter for $\underline{b}_1$ is a detection filter for $\underline{b}_2$

(b) the unidirectional output error generated by the failure associated with $\underline{b}_2$ is in the same output direction as that associated with $\underline{b}_1$.

° detection space of an event

The detection space of $\underline{b}_1$ contains all events which are detection equivalent to $\underline{b}_1$.

(a) Vector space definition (Jones).

Let $\underline{b}_1$ be an event vector associated with a failure. Assume that $C\,\underline{b}_1 \neq 0$. Detection space for $\underline{b}_1$ is denoted by $\overline{R}_1$ and is the direct sum

$\overline{R}_1 = \underline{b}_1 \oplus R_1$ where $R_1 \subset R^n$ is the largest subspace satisfying the three conditions

(1) $\eta(M) \cap R_1 = \emptyset$

(2) $R_1 \subset \eta(C)$ where $M = \begin{bmatrix} C \\ CA \\ \cdot \\ \cdot \\ CA^{n-1} \end{bmatrix}$

(3) $AR_1 \subset \overline{R}_1$

As $\underline{b}_1 \notin \eta(M)$, condition (1) ensures that $\overline{R}_1 = \underline{b}_1 \oplus R_1$ is an observable subspace for (A,B,C). Condition (2) ensures that for every vector of $\overline{R}_1$, $\overline{\xi} = \alpha \underline{b}_1 + \xi$ where $\xi \in R_1$

$$C\overline{\xi} = \alpha C\underline{b}_1 + C\xi = \alpha C\underline{b}_1 + 0 = \alpha C\underline{b}_1$$

Then every vector of $\overline{R}_1$ has the same output direction as $\underline{b}_1$.

(b) Matrix notation (Beard)

$\overline{R}_1$ can be shown to be the null space of

$$M' = \begin{bmatrix} E_m - C\underline{b}_1[C\underline{b}_1^T C\underline{b}_1]^{-1}(C\underline{b}_1)^T \\ [E_m - C\underline{b}_1[C\underline{b}_1^T C\underline{b}_1]^{-1}(C\underline{b}_1)^T][A - A\underline{b}_1[C\underline{b}_1^T C\underline{b}_1]^{-1}(C\underline{b}_1)^T C] \\ \vdots \\ [E_m - C\underline{b}_1[C\underline{b}_1^T C\underline{b}_1]^{-1}(C\underline{b}_1)^T][A - A\underline{b}_1[C\underline{b}_1^T C\underline{b}_1]^{-1}(C\underline{b}_1)^T C]^{n-1} \end{bmatrix}$$

where E_m is the identity matrix in a m dimension space.

Note: If $C\,\underline{b}_1 = 0$, just replace $\underline{b}_1$ by $A^{\mu-1}\underline{b}_1$ where μ is the smallest integer such that $CA^{\mu-1}\,\underline{b}_1$ is not zero, in the above definition.

° <u>detection generator</u>

It is possible to show that the detection space $\overline{R}_{b_1}$ is cyclic invariant, and that there exists a unique vector $\underline{g}$ in $\bar{R}_{b_1}$ such that the vectors

$$\underline{g},\ A\ \underline{g}, \ldots,\ A^{\nu_{b_1}-1}\ \underline{g} \text{ are a basis for } \bar{R}_{b_1}$$

$$CA^{\nu_{b_1}-1}\ \underline{g} = C\ \underline{b}_1$$

Vector definition (Jones)

Let $d(\bar{R}_{b_1}) = \nu_{b_1} \cdot \underline{g}$ is the detection generator of $\bar{R}_{b_1}$ if

(1) $A^k \underline{g} \in R_{b_1} \qquad k < \nu_{b_1} - 1$

(2) $CA^{\nu_{b_1}-1}\underline{g} = C\ \underline{b}_1$

Beard showed that if the ν_{b_1} eigenvalues of A - DC associated with the controllable subspace of $\underline{b}_1$ are given by the roots of

$$s^{\nu_{b_1}} + p_{\nu_{b_1}}\ s^{\nu_{b_1}-1} + \ldots + p_2\ s + p_1 = 0$$

where the p_i are scalars (which implies that if a desired eigenvalue of $\bar{R}_{b_1}$ is complex, its conjugate must be selected too, hence the almost arbitrarily assignability concept), then D must be a solution of

$$DCA^{\nu_{b_1}-1}\ \underline{g} = p_1\ \underline{g} + p_2\ A\ \underline{g} + \ldots + p_{\nu_{b_1}}\ A^{\nu_{b_1}-1}\ \underline{g} + A^{\nu_{b_1}}\ \underline{g}$$

° <u>Mutually detectable set of events</u> (Jones)

Given the inhomogeneous error equations

$$\dot{\underline{\mathcal{E}}} = (A - DC)\ \underline{\mathcal{E}} + \underline{b}_i\ n_i(t) \qquad\qquad i = 1,\ldots,r$$

$$\underline{\mathcal{E}}' = C\ \underline{\mathcal{E}}$$

The failures associated with the events $\underline{b}_1,\ldots,\underline{b}_r$ are mutually detectable by a single failure detection system if

(1) the output generated by each of $\underline{b}_1 \, n_1(t), \ldots, \underline{b}_r \, n_r(t)$ maintains a fixed direction in the output space, and

(2) the eigenvalues of A - DC can be specified almost arbitrarily by a proper choice of D.

This definition says nothing about the output directions $C \, \underline{b}_i$. From a practical point of view, however, one case can be immediately examined: if two events $\underline{b}_1$ and $\underline{b}_2$ are such that $C \, \underline{b}_1$ is parallel to $C \, \underline{b}_2$, there are 2 possibilities:

(1) If $\underline{b}_2 \in \bar{R}_1$, then $\underline{b}_2$ and $\underline{b}_1$ are detection equivalent, and a detection filter cannot distinguish between failures associated with these two events.

(2) If $\underline{b}_2 \notin \bar{R}_1$ it can be shown that if $C_{\underline{b}_1} /\!/ C_{\underline{b}_2}$ a failure detection system which detects failures associated with $\underline{b}_1$ cannot simultaneously detect failures associated with $\underline{b}_2$. The output error for the second failure cannot be constrained to a single direction.

This can be generalized to the case where one output direction $C \, \underline{b}_i$ can be expressed as a linear combination of others. In that case, to have unidirectional outputs, it can be shown that eigenvalues for each detection space can no longer be arbitrarily assigned. Hence the new concept.

° Output separability

Vectors $\underline{b}_1, \ldots, \underline{b}_r$ are output separable if the rank of $[C \, \underline{b}_1, \ldots, C \, \underline{b}_r] = r$.

It can be shown that output separability is sufficient to guarantee that a D can be found for which failures associated with $\underline{b}_1, \ldots, \underline{b}_r$ produce unidirectional output errors. If the dimension of $\bar{R}_i$ is denoted by ν_i, $\nu_s' = \sum_{i=1,r} \nu_i$ eigenvalues of A - DC can be almost arbitrarily assigned by the choice of D.

Output separability, unfortunately, does not imply mutual detectability: it may happen that eigenvalues of each $\bar{R}_i$ can be almost arbitrarily assigned, when the $\underline{b}_i$ are output separable, but that some eigenvalues of A - DC are determined, and cannot be changed. More precisely:

° Let S be the space spanned by $[\underline{b}_1, \ldots, \underline{b}_r]$. <u>We can define a detection space for S, $\bar{R}_s$</u> such that (Jones)

$$\bar{R}_s = R_s \oplus S$$

where R_s is the largest subspace which satisfies

(1) $\eta(M) \cap R_s = 0$

(2) $R_s \subset \eta(C)$

(3) $AR_s \subset \bar{R}_s$

$\bar{R}_s$ can be shown to be the null space of $\begin{bmatrix} C_s' \\ C_s'(A - D_s C) \\ \vdots \\ C_s'(A - D_s C)^{n-1} \end{bmatrix}$

where

$$D_S = A\bar{B}[(C\bar{B})^T(C\bar{B})]^{-1}(C\bar{B})^T$$

$$C_S' = [E_m - C\bar{B}[(C\bar{B})^T(C\bar{B})]^{-1}(C\bar{B})^T]C$$

and $\bar{B}$ is the matrix $[\underline{b}_1, \ldots, \underline{b}_r]$.

$\bar{R}_s$ is a direct extension of $\bar{R}_i$ defined formerly. In a sense, it is the set of events which are detection equivalent to the set of $\underline{b}_i$, $i = 1, \ldots, r$. If the $\underline{b}_i$'s are output separable, we have

$$\bar{R}_1 \oplus \bar{R}_2 \oplus \ldots \oplus \bar{R}_r \subset \bar{R}_s$$

Let us define $\nu_s' = \nu_1 + \ldots + \nu_r$ where ν_i = dim of $\bar{R}_i$

ν_s = dimension of $\bar{R}_s$

If D is chosen such that it makes outputs associated with $\underline{b}_1, \ldots, \underline{b}_r$ unidirectional, only ν_s' of the eigenvalues associated with $\bar{R}_s$ can be almost arbitrarily assignable, the remaining $\nu_s - \nu_s'$ are unassignable. Therefore

$\bar{R}_1, \ldots, \bar{R}_r$ are mutually detectable $\Leftrightarrow \sum_{i=1,r} \nu_i = \nu_s$

The preceding concepts are sufficient to understand the basic structure of the algorithm written to help the designer develop a detection filter. What follows is necessary to understand how the program can help the designer to find the values of the unassignable eigenvalues if the $\underline{b}_i$'s are not mutually detectable, and a detectable subset if these eigenvalues are not acceptable.

° Excess subspace. If $\underline{b}_1, \ldots, \underline{b}_r$ are not mutually detectable, the excess subspace of $\bar{R}_s$ is any subspace $R_o \subset \eta(C)$ which satisfies $\bar{R}_s = \bar{R}_1 \oplus \ldots \oplus \bar{R}_r \oplus R_o$.

In general, it is not unique. However, it can be shown that the ν_o eigenvalues of A-DC associated with R_o (which are the ν_o unassignable eigenvalues of A-DC for a given set of $\underline{b}_i$'s) are independent of the choice of D. The algorithm used will determine a basis of the unique subspace R_{og} defined by:

- R_{og} excess subspace of $\bar{R}_g$

- $AR_{og} \subset R_{og} \oplus \underline{g}_1 \oplus \cdots \oplus \underline{g}_r$

It can be shown that R_{og} is the null space of the matrix

$$M_o = \begin{bmatrix} \bar{c}_1 \\ \vdots \\ \bar{c}_1 (A - D_g C)^{\nu_1 - 1} \\ \bar{c}_2 \\ \vdots \\ \bar{c}_2 (A - D_g C)^{\nu_2 - 1} \\ \vdots \\ \bar{c}_r (A - D_g C)^{\nu_r - 1} \end{bmatrix}$$

° where the $\bar{c}_i$'s are the rows of the matrix $[(C\bar{B})^T(C\bar{B})]^{-1}(C\bar{B})^T C$

° $D_g = A\bar{B}[(C\bar{B})^T(C\bar{B})]^{-1}(C\bar{B})^T$

° $\bar{B}$ matrix $[\underline{b}_1, \ldots, \underline{b}_r]$

Let us call R_{og} this basis. Once it is determined, as $AR_{og} \subset R_{og} \oplus \underline{g}_1 \oplus \cdots \oplus \underline{g}_r$, we have with $G = [\underline{g}_1, \ldots, \underline{g}_r]$

$$AR_{og} = R_{og} \Pi + G \theta$$

it can be shown that the rows of θ are equal to

$$\theta_i = \bar{c}_i \ (A - D_g \ C)^{\nu_i} \ R_{og} \text{ for } i = 1, \ldots, r \qquad (2\text{-}1)$$

As A, R_{og}, θ are known, Π can be computed, and the <u>unassignable eigenvalues of A - DC associated with $\bar{B}$ are the eigenvalues of Π</u>.

° Property of R_{og}: suppose we have a set of events $\underline{b}_1, \ldots, \underline{b}_r$ which are not mutually detectable for a system (A,C) . If we define R_{og_k} = excess subspace associated with

$$(\underline{b}_1, \ldots, \underline{b}_{k-1}, \underline{b}_{k+1}, \ldots, \underline{b}_r)$$

for k = 1,...,r, Jones showed that

(a) $R_{og_k} \subset R_{og}$ for all k

(b) The excess subspace associated with the set of events $\underline{b}_i$ where $\underline{b}_j$ and $\underline{b}_k$ have been extracted is

$$R_{og_i} = R_{og_j} \cap R_{og_k}$$

The program written uses this property to help the designer to find a subset of the $\underline{b}_i$'s with acceptable unassignable eigenvalues: (it is an option)

° for each event $\underline{b}_i$ it computes the set Λ_i of unassignable eigenvalues associated with $\underline{b}_1, \ldots, \underline{b}_{i-1}, \underline{b}_{i+1}, \ldots, \underline{b}_n$

° the designer knows that if he eliminates $\underline{b}_i$, for $i \in J$ of the set of events, the unassignable eigenvalues of the set of events, $\underline{b}_j$, $j \subset [(1,\ldots,r) - J]$ will be $\bigcap_{i \in J} \Lambda_i$

In particular, if $\bigcap_{i \in J} \Lambda_i = \emptyset$, the remaining events once the events $\underline{b}_i$, $i \in J$, have been extracted are mutually detectable.

° Output stationarity (Reference 2)

Assume $\underline{b}_1, \ldots, \underline{b}_k$ are output separable, and let D* be the class of operators such that for every $D \in D^*$ the output generated by each of $\underline{b}_1, \ldots, \underline{b}_k$ with respect to (A - DC, C) is unidirectional. The subspace $\bar{\mathcal{J}}$ can be made output stationary with $\underline{b}_1, \ldots, \underline{b}_k$ if there exists a $D' \in D^*$ such that the output from every element $\xi_i \in \bar{\mathcal{J}}$ for (A - D'C, C) is unidirectional along $C\xi_i$ (or $CA\xi_i$ if $C\xi_i = 0$, etc).

The cost of using output stationarity to increase the number

<u>of failures which can be detected by a single failure detection system is that certain eigenvalues of A - DC may have to be assigned a multiplicity greater than one</u>.

Suppose we want to make $\underline{h}_i$ output stationary with $\underline{b}_1,\ldots,\underline{b}_r$ where $\underline{b}_1,\ldots,\underline{b}_r$ is a set of output separable vectors, and $\underline{b}_1,\ldots,\underline{b}_r,\underline{h}_i$ are not output separable. This implies that $C\,\underline{h}_i$ is a linear combination of $C\,\underline{b}_1,\ldots,C\,\underline{b}_r$. Suppose that $\underline{b}_1,\ldots,\underline{b}_\ell$ is the smallest subset of $\underline{b}_1,\ldots,\underline{b}_r$ such that

$$C\,F_L^C\,\alpha_L^C = \beta_i\,C\,\underline{h}_i$$

has a solution where

$$\alpha_L^C = [\alpha_1,\ldots,\alpha_\ell]^T$$

$$F_L^C = [\underline{b}_1,\ldots,\underline{b}_\ell]$$

and $\alpha_1,\ldots,\alpha_\ell$ are a set of nonzero coefficients. If $\underline{b}_1,\ldots,\underline{b}_\ell$ are mutually detectable, it can be demonstrated that $\underline{h}_i$ can be made output stationary with $\underline{b}_1,\ldots,\underline{b}_r$ if there exists a solution Ξ to $\bar{R}_L^C\,\Xi = \bar{S}_i$ where

$$\bar{R}_L^C = [\bar{R}_1,\ldots,\bar{R}_\ell]$$

and where $\bar{S}_i = [\underline{h}_i : S_i]$, detection space of $\underline{h}_i$.

CHAPTER 3

DETECTION FILTER DESIGN ALGORITHM

A - Principle of the algorithm

There are two basic steps in the algorithm:

During the first part, the designer has to find an acceptable set of events, either output separable and mutually detectable—that is to say without unassignable eigenvalues—or output separable and with acceptable unassignable eigenvalues. The program first tests the output separability of the events, and, if the events are output separable, goes on to compute the detection space $\bar{R}_{\mathcal{S}}$ associated with the whole set of events, and the detection space $\bar{R}_i$ associated with each $\underline{b}_i$. In the process, it computes the detection generator $\underline{g}_i$ associated with each $\bar{R}_i$.

If $rk(\bar{R}_{\mathcal{S}}) = \sum_i rk(\bar{R}_i)$, the events are mutually detectable. Then a detection filter D can be designed such that all the eigenvalues of A - DC are almost arbitrarily assignable.

If $rk(\bar{R}_{\mathcal{S}}) > \sum_i rk(\bar{R}_i)$, there are $rk(\bar{R}_{\mathcal{S}}) - \sum_i rk(\bar{R}_i)$ unassignable eigenvalues in A - DC, independently of the choice of D. The program goes on to compute these unassignable eigenvalues. Three possibilities are then offered to the designer:

- ° accept the unassignable eigenvalues, and go to the next step;
- ° find a subset of the $\underline{b}_i$'s with no unassignable eigenvalues.

To do this, the program computes for each $\underline{b}_i$ the set Λ_i of unassignable eigenvalues associated with the events $(\underline{b}_1, \ldots, \underline{b}_{i-1}, \underline{b}_{i+1}, \ldots, \underline{b}_r)$. The designer then takes out the events b_j, $j \in I$, I a set of indices, $(I \subset [1,..,r])$ such that

$\bigcap_{j \in I} \Lambda_j$ contains only acceptable unassignable eigenvalues. If, for example, one of the Λ_i's, assume Λ_1, is the null set, the designers know that the events $\underline{b}_2, \ldots, \underline{b}_r$ are mutually detectable.

° increase the dimension of state space. (Not yet operational, he has to go back to the beginning with the new A, B, C, and the new set of events $\bar{B}$).

Once the designer has an acceptable set of events, the program goes on to compute the detection filter with the desired eigenvalues. This is done in <u>base normal canonical form</u> (Jones). The transformation matrix from the given coordinate system to base normal canonical form is defined to be T^{-1}. The general structure for T^{-1} is

$$T^{-1} = \left[\underline{g}_1, \ldots, A^{\nu_1 - 1}\underline{g}_1, \underline{g}_2, \ldots, A^{\nu_2 - 1}\underline{g}_2, \ldots, \underline{g}_r, \ldots, A^{\nu_r - 1}\underline{g}_r, T_o \right] \qquad (3\text{-}1)$$

where the $\underline{g}_i$'s are the detection generators of the $\bar{R}_i$'s

ν_i is the dimension of $\bar{R}_i$

T_o is a matrix such that T is nonsingular. The choice made for T_o is:

$$T_o = \left[\underline{z}_1, \ldots, \underline{z}_{\gamma_o}, \underline{w}_{r+1}, \ldots, (A - D_s C)^{q_{r+1} - 1}\underline{w}_{r+1}, \ldots, (A - D_s C)^{q_m - 1}\underline{w}_m \right]$$

where $\underline{z}_1, \ldots, \underline{z}_{\gamma_o}$ is a basis for R_{oG} (If the events are mutually detectable, R_{oG} has dimension 0).

The $\underline{w}_{r+1},\ldots,\underline{w}_m$ are chosen so that the vectors

$$\underline{w}_{r+1},\ldots,(A-D_sC)^{q_{r+1}-1}\underline{w}_{r+1},\ldots,(A-D_sC)^{q_m-1}\underline{w}_m$$

complete the set $\underline{g}_1,\ldots,A^{\gamma_1-1}\underline{g}_1,\ldots,A^{\gamma_r-1}\underline{g}_r,\underline{z}_1,\ldots,\underline{z}_{\gamma_0}$ to form a basis in R^n. If $\bar{R}_s$ is already of dimension n, where n is the state space dimension, there is no need for the vectors $\underline{w}_{r+1},\ldots,\underline{w}_m$. The set

$$\underline{g}_1,\ldots,A^{\gamma_1-1}\underline{g}_1,\ldots,A^{\gamma_r-1}\underline{g}_r,\underline{z}_1,\ldots,\underline{z}_{\gamma_0}$$

a basis of $\bar{R}_s$, is also a basis for R^n.

More precisely, the $\underline{w}_i$ are the auxiliary vectors associated with $C'_i(A - D\ C)^{q_i-1}$ in the orthogonal reduction of

$$M_c = \begin{bmatrix} C_s' \\ C_s'(A-D_sC) \\ \vdots \\ C_s'(A-D_sC)^{n-1} \end{bmatrix}$$

starting with the identity matrix where C'_i is a row of C'_s

$$C_s' = \left[E_m - C\bar{B}\left[(C\bar{B})^T(C\bar{B})\right]^{-1}(C\bar{B})^T\right]C$$

$$D_s = A\bar{B}\left[(C\bar{B})^T(C\bar{B})\right]^{-1}(C\bar{B})^T$$

q_i is the largest integer such that all of $C'_i,\ldots,C'_i(A - D\ C)^{q_i-1}$ have a nonzero auxiliary vector (See appendix D on orthogonal reduction procedure for the definition of auxiliary vectors).

All but m of the vectors of the right hand side of (3-1) are in the null space of C. These m columns are used to define a transformation of the output space compatible with T^{-1}.

$$T_m^{-1} = \left[CA^{\gamma_1-1}\underline{g}_1,\ldots,CA^{\gamma_r-1}\underline{g}_r,\ C_s'(A-D_sC)^{q_{r+1}-1}\underline{w}_{r+1},\ldots,C_s'(A-D_sC)^{q_m-1}\underline{w}_m\right]^{-1}$$

T^{-1} transforms the state vector to base normal form and T transforms it back to the original coordinate system. We have the relations

Base normal form | original base

$$\hat{A} = T^{-1} A T$$
$$\hat{B} = T^{-1} B$$
$$\hat{C} = T_m^{-1} C T$$
$$\hat{\underline{x}}_m = T^{-1} \underline{x}_m$$
$$\hat{\underline{y}}_m = T_m^{-1} \underline{y}_m$$

This transformation is used because the design of the detection filter in the canonical basis is straightforward: due to the two relations

$$0 \subset A^{\nu_i - 1} \underline{g}_i = \rho_{i_1} \underline{g}_i + \rho_{i_2} A \underline{g}_i + \cdots + \rho_{i_{\nu_i}} A^{\nu_i - 1} \underline{g}_i + A^{\nu_i} \underline{g}_i \qquad (3\text{-}2)$$

where the ρ_{i_k} are the coefficients of the polynomial $\prod_{i=1,\nu_i} (s - \lambda_i)$ of the eigenvalues of $\bar{R}_i$: $s^{\nu_i} + \rho_{i_{\nu_i}} s^{\nu_i - 1} + \cdots + \rho_{i_2} s + \rho_{i_1}$ and

$$AR_{og} = R_{og} \Pi + G\Theta \text{ , we have:} \qquad (3\text{-}3)$$

$\hat{A}$ is of the form

$$\hat{A} = \begin{bmatrix} \hat{A}'_{11} & \hat{A}_{12} & \cdots & \hat{A}_{1r} & \hat{\theta}_1 & \hat{\Gamma}'_1 \\ \hat{A}_{21} & \hat{A}'_{22} & & \vdots & \vdots & \vdots \\ \vdots & & \ddots & \vdots & \vdots & \vdots \\ \vdots & & & \ddots & & \\ \hat{A}_{r1} & - & - & \hat{A}'_{rr} & \hat{\theta}_r & \hat{\Gamma}'_r \\ \hat{A}_{o1} & - & - & \hat{A}_{or} & \Pi & \Gamma'_o \\ \hat{A}_{r1} & - & - & \hat{A}_{rr} & 0 & \hat{A}'_{rr} \end{bmatrix}$$

where $\hat{\theta}_i = \begin{bmatrix} \theta_i \\ 0 \end{bmatrix}$ matrices with only one nonzero row, defined in Chapter 2, equation (2-1)

Π is a $\nu_o \times \nu_o$ matrix associated with R_{og}, given by relation (3-3)

$$\hat{A}_{ij} = \begin{bmatrix} 0 & \cdots & 0 & \hat{a}_{ij_1} \\ \vdots & & \vdots & \vdots \\ 0 & \cdots & 0 & \hat{a}_{ij\gamma_i} \end{bmatrix} = \left[0 \,\vdots\, \underline{\hat{a}}_{ij} \right]$$ is a $\gamma_i \cdot \gamma_j$ matrix, for $i \neq j$

$$\hat{A}'_{ii} = \begin{bmatrix} 0 & \cdots & 0 & -\rho'_{i1} \\ 1 & \ddots & & \vdots \\ 0 & \ddots & & \vdots \\ 0 & \cdots 0 & 1 & -\rho'_{i\gamma_i} \end{bmatrix}$$ is a $\gamma_i \cdot \gamma_i$ matrix (3-4)

$\hat{A}'_{rr}$ is $(n-\gamma_s) \times (n-\gamma_s)$

$\hat{\Gamma}'_i$ is $\gamma_i \times (n-\gamma_s)$

$\hat{\Gamma}'_0$ is $\gamma_0 \times (n-\gamma_s)$

$\hat{C}$ is of the form

$$\hat{C} = \begin{bmatrix} \underline{\hat{c}}_1 & 0 & \cdots & 0 \\ 0 & \underline{\hat{c}}_2 & \cdots & 0 \\ 0 & \cdots & \hat{c}_n & 0 \\ 0 & \cdots & 0 & \hat{c}_{rm} \end{bmatrix}$$ where $\underline{\hat{c}}_i = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}$ is a γ_i vector

In order for D to be a detection filter for $[\underline{b}_1, \ldots, \underline{b}_r]$, each space $\bar{R}_i$ has to be invariant for A - DC, or, equivalently, for $\hat{A} - \hat{D}\hat{C}$. If the ρ_{ik} are the coefficients of the desired set of eigenvalues of $\bar{R}_i$, $\hat{A} - \hat{D}\hat{C}$ is equal to

$$\hat{A} - \hat{D}\hat{C} = \begin{bmatrix} \hat{A}_{11} & 0 & \cdots & 0 & \hat{\theta}_1 & \hat{\Gamma}'_1 \\ 0 & \hat{A}_{22} & & & \hat{\theta}_2 & \hat{\Gamma}'_2 \\ \vdots & & \ddots & & \vdots & \vdots \\ 0 & \cdots & & \hat{A}_{nn} & \hat{\theta}_n & \hat{\Gamma}'_n \\ 0 & \cdots & & 0 & \pi & \hat{\Gamma}'_0 \\ 0 & & & 0 & 0 & \hat{A}'_{rr} \end{bmatrix}$$ where $$\hat{A}_{ii} = \begin{bmatrix} 0 & \cdots & 0 & -\rho_{i1} \\ 1 & \ddots & & \vdots \\ 0 & \cdots 0 & 1 & -\rho_{i\gamma_i} \end{bmatrix}$$

If a $\hat{D}$ is selected so that $\hat{A} - \hat{D}\hat{C}$ is of this form, outputs associated with each $\underline{b}_i$ will be unidirectional and the eigenvalues associated with each $\bar{R}_i$ can be chosen (if the dimension of $\bar{R}_\ell$ is less than n, the state space dimension, n - ν_ℓ eigenvalues of A - D C will be the same as those of A. They could be selected too, but the program does not do it. See Ref. 2 for more details).

To set $\hat{A} - \hat{D}\hat{C}$ in the desired form, $\hat{D}$ is selected to be the sum of two terms

$$\hat{D} = \hat{D}_{FR} + \hat{D}_{\psi}$$

with

$$\hat{D}_{FR} = \begin{bmatrix} 0 & \underline{\hat{a}}_{12} & \cdots & \underline{\hat{a}}_{1n} & 0 & \cdots & 0 \\ \underline{\hat{a}}_{21} & 0 & \cdots & \underline{\hat{a}}_{2n} & 0 & \cdots & 0 \\ \vdots & & & \vdots & & & \\ \underline{\hat{a}}_{n1} & \underline{\hat{a}}_{n2} & \cdots & 0 & & & \\ \underline{\hat{a}}_{01} & \underline{\hat{a}}_{02} & \cdots & \underline{\hat{a}}_{0n} & & & \\ \underline{\hat{a}}_{r1} & \underline{\hat{a}}_{r2} & \cdots & \underline{\hat{a}}_{rn} & 0 & \cdots & 0 \end{bmatrix}$$

$$\hat{D}_{\psi} = \begin{bmatrix} \underline{d}_{\psi_1} & 0 & \cdots & & 0 \\ 0 & \underline{d}_{\psi_2} & \cdots & & 0 \\ \vdots & & \ddots & & \\ 0 & \cdots & & \underline{d}_{\psi_n} \cdots & 0 \\ 0 & \cdots & & & 0 \\ 0 & \cdots & & & 0 \end{bmatrix}$$

with

$$\underline{d}_{\psi_i} = \begin{bmatrix} -p'_{i1} + p_{i1} \\ -p'_{i2} + p_{i2} \\ \vdots \\ -p'_{i\nu_i} + p_{i\nu_i} \end{bmatrix}$$

$i = 1, \ldots, r$ with the $p'_{i_1}, \ldots, p'_{i\nu_i}$ defined in (3-4).

Finally, the program computes

$$D = T \hat{D} T_m^{-1}$$

B - Algorithm of the detection filter program

BA) Reading of data

Step 1. Enter A,B,C, $\underline{b}_i$ dimensions A(n,n), B(n,q), C(m,n), $\bar{B}$(n,r)

Step 2. Compute rank of C.

Step 3. If there are more than rank C events to be detected, divide the $\underline{b}_i$ into groups of no more than rank C vectors in a group.

BB) Output separability

Step 4. Test each group for output separability. Given a group $\underline{b}_1, \ldots, \underline{b}_r$ compute C $\underline{b}_1, \ldots,$ C$\underline{b}_r$ and check whether they are linearly independent. If not, the group has to be changed.

BC) Mutual detectability

Step 5. For each group of events selected, mutual detectability will be investigated. For each group, $\bar{R}_\gamma$, $\bar{R}_1$, $\bar{R}_2, \ldots, \bar{R}_r$ will be computed, as well as $\underline{g}_1, \ldots, \underline{g}_r$.

5.a Apply orthogonal reduction to C, starting with $\Omega^{(1)} = E_n$. Let Ω_c be the terminating matrix.

5.b Compute

$$D_\gamma = A\bar{B}\left[(C\bar{B})^T(C\bar{B})\right]^{-1}(C\bar{B})^T$$

$$C'_\gamma = \left[E_m - C\bar{B}\left[(C\bar{B})^T(C\bar{B})\right]^{-1}(C\bar{B})^T\right]C$$

Compute

$$M_{D'} = \begin{bmatrix} C'_\gamma \\ C'_\gamma(A - D_\gamma C) \\ \vdots \\ C'_\gamma(A - D_\gamma C)^{n-1} \end{bmatrix}$$

Apply orthogonal reduction to $M_{D'}$ starting with $\Omega^{(1)} = \Omega_c$.

Let Ω_s be the terminating matrix, $\nu_s' = rk(\Omega_s)$.

(Note: an option allows to start the orthogonal reduction with the identity matrix to compute the $\underline{w}_i$ needed in the final steps, and the integers q_i, such that $c_i'(A - D_s c)^{q_i - 1}$ has a nonzero auxiliary vector, q_i largest integer for which this is true).

5.c For each $\underline{b}_i$ i = 1,r

Compute

$$D_{b_i} = A\underline{b}_i[(C\underline{b}_i)^T(C\underline{b}_i)]^{-1}(C\underline{b}_i)^T$$

$$C' = [E_m - (C\underline{b}_i)[(C\underline{b}_i)^T(C\underline{b}_i)]^{-1}(C\underline{b}_i)^T]C$$

Reduce

$$M_{D_i} = \begin{bmatrix} C' \\ C'(A - D_{b_i}C) \\ \vdots \\ C'(A - D_{b_i}C)^{n-1} \end{bmatrix} \qquad \text{starting with } \Omega^{(1)} = \Omega_s$$

Let Ω_i be the terminating matrix $rk(\Omega_i) = \nu_i'$

5.d Compute

$$M = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix}$$

Apply orthogonal reduction to M, starting with each Ω_i, i = 1,...,r. Each reduction ends on a zero matrix. The last vector to be removed from the range space of Ω_i is a multiple of the detection generator $\underline{g}_i$.

5.e Compute $\nu_s = \nu_s' + r$ $\qquad\qquad$ ν_s = dimension of $\bar{R}_s$

$\nu_i = \nu_i' + 1$ for all i = 1,...,r. ν_i = dimension of $\bar{R}_i$.

Check whether $\nu_s = \sum_{i=1,r} \nu_i$

If equality holds, the events are mutually detectable, go to Step 6. If not go to step 5.f.

5.f (determines excess subspace R_{og}). If $\nu_o = \nu_s - \sum_{i=1,r} \nu_i$, a total of ν_o eigenvalues of A - DC are unassignable; they will be computed.

° Compute $[(C\,\bar{B})^T (C\,\bar{B})]^{-1} (C\,\bar{B})^T C = \begin{bmatrix} \bar{c}_1 \\ \vdots \\ \bar{c}_r \end{bmatrix}$

(partly done in step 5.b)

° Compute

$$M_o = \begin{bmatrix} \bar{c}_1 \\ \bar{c}_1 (A - D_s C) \\ \vdots \\ \bar{c}_1 (A - D_s C)^{\nu_1 - 1} \\ \vdots \\ \bar{c}_r (A - D_s C)^{\nu_2 - 1} \end{bmatrix}$$

° Apply orthogonal reduction to M_o starting with Ω_s. Let Ω_{og} be the terminating matrix $rk(\Omega_{og}) = \nu_s$

° Define $R_{og} = [z_1, \ldots, z_{\nu_o}]$ where the z_i's are ν_o linearly independent columns of Ω_{og}

° Compute $\theta_i = \bar{c}_i\ (A - D_s\ C)^{\nu_i}\ R_{og}$ $\qquad$ for i = 1,...,r

Form matrix θ whose rows are the θ_i's

° Compute $\Pi = [R_{og}^T R_{og}]^{-1} R_{og}^T [A\ R_{og} - G\theta]$

where $G = [g_1, \ldots, g_r]$ matrix of the generators.

° Compute eigenvalues of Π which are the unassignable eigenvalues associated with the set of events.

5.g Three options open:

° accept the eigenvalues of Π ; go to step 6

° find a subset of $\underline{b}_1, \ldots, \underline{b}_r$ with acceptable unassignable eigenvalues ; go to step 5h

° increase dimension of state space ; go to step 1.

5.h (indicate unassignable eigenvalues associated with each $\underline{b}_i$). For each i, i = 1,r compute

$G_i = G$ with the ith column deleted (G defined in step 5f)

$\theta_i = \theta$ with the ith row deleted (θ defined in step 5f)

θ_i^c = ith row of θ

$$M_{oi} = \begin{bmatrix} \theta_i^c \\ \theta_i^c \Pi \\ \vdots \\ \theta_i^c \Pi^{r_i - 1} \end{bmatrix}$$

Apply orthogonal reduction to M_{oi}. Let β_i' be the terminating matrix. Find a matrix β_i whose columns span the column space of β_i'. Find a matrix Δ whose columns span the row space of M_{oi}. Form $[\Delta \vdots \beta]_i$

Compute

$$\Pi_i^{\circ} = [\Delta \vdots \beta]_i^{-1} \Pi [\Delta \vdots \beta]_i = \begin{bmatrix} \Pi_i^c & \vdots & 0 \\ \hline \Pi_{io}^c & \vdots & \Pi_i \end{bmatrix}$$

Find eigenvalues of Π_i. Let Λ_i be this set. Λ_i is the set of unassignable eigenvalues associated with $[\underline{b}_1, \ldots, \underline{b}_{i-1}, \underline{b}_{i+1}, \ldots, \underline{b}_r]$.

5.i Test for output stationarity: determine whether it is possible to make an event $\underline{b}_a$ output stationary with $\underline{b}_1, \ldots, \underline{b}_r$.

° compute $\bar{S}_a$, the detection space associated with $\underline{b}_a$ (same computation as in 5.c except that orthogonal reduction of M_D associated with $\underline{b}_a$ starts with identity matrix)

° find the subset of $\underline{b}_1, \ldots, \underline{b}_r$ of which $\underline{b}_a$ is a linear combination. Call J the set of indices of $\underline{b}_i$ such that

$$\underline{b}_a = \sum_{k \in J} \alpha_k \underline{b}_k \qquad J \subset [1, \ldots, r]$$

° Form matrix $\bar{R}_i^c = [\ldots \bar{R}_k \ldots]$ for $k \in J$. Output stationarity will be possible if there exists some Ξ such that

$$\bar{R}_i^c \, \Xi = \bar{S}_a$$

If $\nu_a = rk(\bar{S}_a)$, ν_a eigenvalues associated with each $\bar{R}_i$, $i \in J$, are equal to the eigenvalues selected for $\bar{R}_a$. If, for some $i \in J$, $rk(\bar{R}_i) = \nu_i > \nu_a$, $\nu_i - \nu_a$ eigenvalues of $\bar{R}_a$ are unassignable.

Note: the step 5.i is only partly implemented in the program, as it is in June 78, and not fully tested out.

BD) Filter design

Step 6. Let $\{\underline{b}_i\}$ i = 1,...,r be the set of events at this point.

6.a Use R_{og} defined in step 5.f. Use results of step 5.b: $\underline{w}_i$ and q_i. Form the matrix

$$T_o = \left[R_{og}, \underline{w}_{r+1}, \ldots, (A - D_s C)^{q_{r+1}-1} \underline{w}_{r+1}, \ldots, (A - D_s C)^{q_m - 1} \underline{w}_m \right]$$

6.b Form the matrix

$$T = \left[\underline{g}_1, --, A^{\gamma_1 - 1} \underline{g}_1, --, A^{\gamma_r - 1} \underline{g}_r : T_o \right]$$

Compute T^{-1}

Form $T_m = \left[c A^{\gamma_1 - 1} \underline{g}_1, --, c A^{\gamma_r - 1} \underline{g}_r, c_s'(A - \underline{D}_s c)^{q_{r+1} - 1} \underline{w}_{r+1}, --, c_s'(A - \underline{D}_s c)^{q_m - 1} \underline{w}_m \right]$

Compute T_m^{-1}

6.c Compute $\hat{A} = T^{-1}AT$, $\hat{B} = T^{-1}B$, $\hat{C} = T_m^{-1} CT$

$\hat{A}$ is of the form

$$\hat{A} = \begin{bmatrix} \hat{A}'_{11} & \hat{A}_{12} & -- & \hat{A}_{1r} & \hat{\theta}_1 & \hat{\Gamma}'_1 \\ \hat{A}_{21} & & & & \vdots & \vdots \\ \vdots & & & & & \\ \hat{A}_{r1} & & & \hat{A}'_{rr} & \hat{\theta}_r & \hat{\Gamma}'_r \\ \hat{A}_{o1} & - & - & \hat{A}_{or} & \Pi & \hat{\Gamma}'_o \\ \hat{A}_{r1} & - & - & \hat{A}_{rr} & 0 & \hat{A}'_{rr} \end{bmatrix}$$

with $\hat{A}'_{rr}$ $(n - \nu_s)(n - \nu_s)$ matrix

Π a $\nu_o \cdot \nu_o$ matrix

$\hat{\theta}_i = \begin{bmatrix} \theta_i \\ 0 \end{bmatrix}$ only one nonzero row

$\hat{A}_{ij} = \begin{bmatrix} 0 & \cdots & 0 & \hat{a}_{ij1} \\ \vdots & & & \\ 0 & \cdots & 0 & \hat{a}_{ij\gamma_i} \end{bmatrix}$ $\nu_i \cdot \nu_j$ matrix for $i \neq j$

$\hat{A}'_{ii} = \begin{bmatrix} 0 & \cdots & 0 & -p'_{i1} \\ 1 & \ddots & & \vdots \\ \vdots & & & \\ 0 & & 1 & -p'_{i\gamma_i} \end{bmatrix}$ $\nu_i \cdot \nu_i$ matrix

6.d Compute

$$\hat{D}_{FR} = \begin{bmatrix} 0 & \hat{\underline{a}}_{12} & \cdots & \hat{\underline{a}}_{1r} & 0 & 0 \\ \hat{\underline{a}}_{21} & 0 & \cdots & \hat{\underline{a}}_{2r} & \vdots & \vdots \\ \vdots & & & & \vdots & \vdots \\ \hat{\underline{a}}_{r1} & & & 0 & \vdots & \vdots \\ \hat{\underline{a}}_{c1} & & & \hat{\underline{a}}_{cr} & \vdots & \vdots \\ \hat{\underline{a}}_{r1} & & \cdots & \hat{\underline{a}}_{rr} & 0 & 0 \end{bmatrix}$$

° Enter the desired $\lambda_{i1}, \ldots, \lambda_{i\gamma_i}$ (eigenvalues for $\overline{R}_i$), i = 1,r .

Compute

$$\psi_i(\lambda) = (\lambda - \lambda_{i1}) \cdots (\lambda - \lambda_{i\gamma_i})$$

$$= \lambda^{\gamma_i} + p_{i\gamma_i} \lambda^{\gamma_i - 1} + \cdots + p_{i2} \lambda + p_{i1}$$

Compute

$$\underline{d}_{\psi_i} = \begin{bmatrix} -p'_{i1} + p_{i1} \\ \vdots \\ -p'_{i\gamma_i} + p_{i\gamma_i} \end{bmatrix}$$

Compute

$$\hat{D}_{\psi} = \begin{bmatrix} \underline{d}_{\psi_1} & 0 & \cdots & 0 & 0 \\ 0 & \underline{d}_{\psi_2} & \cdots & 0 & \vdots \\ \vdots & & & & \vdots \\ 0 & \cdots & & \underline{d}_{\psi_r} & \vdots \\ 0 & & & 0 & \\ 0 & & & 0 & 0 \end{bmatrix}$$

° Compute $\hat{D} = \hat{D}_{\psi} + \hat{D}_{FR}$

° Compute $D = T \hat{D} T_m^{-1}$.

C - Examples

The following examples are pure mathematical transformations of matrices, described here just in order to illustrate the theoretical notions introduced earlier. No physical background is to be searched for the A, B, C matrices used in this part.

Example 1.

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 6 & 7 \\ 0 & -1 & 0 & 6 & 7 \\ -3 & 0 & -4 & -.8 & 0 \\ -1.5 & 0 & -2 & 0 & -.9 \end{bmatrix} \qquad B = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 3 \\ 1.5 \end{pmatrix}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad \underline{b}_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \qquad \underline{b}_2 = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

Running the program, we find:

- dimension of R_g is 1
- dimension of R_1 is 0
- dimension of R_2 is 0

$$\underline{g}_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \qquad \underline{g}_2 = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

Then, dimension of $\bar{R}_g = 3$ (as $\bar{R} = [\underline{b}_1 \vdots \underline{b}_2] \oplus R_g$)

dimension of $\bar{R}_1 = 1$ (as $\bar{R}_1 = \underline{b}_1 \oplus R_1$)

dimension of $\bar{R}_2 = 1$ (as $\bar{R}_2 = \underline{b}_2 \oplus R_2$)

we have

$$\nu_s > \nu_1 + \nu_2 \quad \text{and} \quad \nu_s - (\nu_1 + \nu_2) = 1$$

There is one unassignable eigenvalue. The program checks its value, which is - 1.

The designer accepts this value, and goes on. The vectors $\underline{w}_i$ obtained in the orthogonal reduction of M_D, starting with the identity matrix (step 5b of the algorithm) turn out to be

$$\underline{w}_3 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \qquad \underline{w}_4 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

Then

$$T_c = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad T = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_m = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \hat{A} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 6 & 7 \\ 0 & 0 & -1 & 6 & 7 \\ -3. & -4 & 0 & -.8 & 0 \\ -1.5 & -2 & 0 & 0 & -.9 \end{bmatrix}$$

Then

$$\hat{D}_{FR} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -3 & -4 & 0 & 0 \\ -1.5 & -2 & 0 & 0 \end{bmatrix}$$

If the designer wants the eigenvalue -8 associated with $\bar{R}_1$ and -9 associated with $\bar{R}_2$,

$$\hat{D}_4 = \begin{bmatrix} 8 & 0 & 0 & 0 \\ 0 & 9 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Then

$$\hat{D} = \begin{bmatrix} 8 & 0 & 0 & 0 \\ 0 & 9 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -3 & -4 & 0 & 0 \\ -1.5 & -2 & 0 & 0 \end{bmatrix} \qquad D = \begin{bmatrix} 8 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 9 & 0 & 0 \\ -3 & -4 & 0 & 0 \\ -1.5 & -2 & 0 & 0 \end{bmatrix}$$

As a further check

$$A - DC = \begin{bmatrix} -8 & -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 6 & 7 \\ 0 & -1 & -9 & 6 & 7 \\ 0 & 0 & 0 & -.8 & 0 \\ 0 & 0 & 0 & 0 & -.9 \end{bmatrix}$$

We see that $(A - DC)\underline{g}_1 = -8\,\underline{g}_1$

$(A - DC)\underline{g}_2 = -9\,\underline{g}_2$

Example 2.

Same matrices A, B, C

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 6 & 7 \\ 0 & -1 & 0 & 6 & 7 \\ -3 & 0 & -4 & -.8 & 0 \\ -1.5 & 0 & -2 & 0 & -.9 \end{bmatrix} \qquad B = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 3. \\ 1.5 \end{pmatrix}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

But, we shall use the events

$$\underline{b}_1 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \qquad \underline{b}_2 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \qquad \underline{b}_3 = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} \qquad \underline{b}_4 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Running the program, we find

- dimension of R_ℓ is 1
- dimension of R_1 is 0
- dimension of R_2 is 0
- dimension of R_3 is 0
- dimension of R_4 is 0

$$\underline{g}_1 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \qquad \underline{g}_2 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \qquad \underline{g}_3 = \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} \qquad \underline{g}_4 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Then dimension of $\bar{R}_s = 5$

dimension of $\bar{R}_1 = 1$

dimension of $\bar{R}_2 = 1$

dimension of $\bar{R}_3 = 1$

dimension of $\bar{R}_4 = 1$

We have

$$\nu_s > \nu_1 + \nu_2 + \nu_3 + \nu_4$$

and

$$\nu_s - (\nu_1 + \nu_2 + \nu_3 + \nu_4) = 1$$

There is one unassignable eigenvalue, the program checks its value, which is 0.

The designer accepts this eigenvalue and goes on. There is no $\underline{w}_i$ in this case ($\dim \bar{R}_s = 5 = \dim$ of state space).

$$T_c = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \qquad T = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

$$T_m = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

$$\hat{A} = \begin{bmatrix} -.8 & 0 & -4 & -3 & 0 \\ 0 & -.9 & -2 & -1.5 & 0 \\ 6 & 7 & -1 & 0 & -1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Then

$$\hat{D}_{FR} = \begin{bmatrix} 0 & 0 & -4 & -3 \\ 0 & 0 & -2 & -1.5 \\ 6 & 7 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

If the designer wants the eigenvalue -7 associated with $\overline{R}_1$, -8 associated with $\overline{R}_2$, -9 associated with $\overline{R}_3$, -10 associated with $\overline{R}_4$)

$$\hat{D}_4 = \begin{bmatrix} 6.2 & 0 & 0 & 0 \\ 0 & 7.1 & 0 & 0 \\ 0 & 0 & 8 & 0 \\ 0 & 0 & 0 & 10 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Then

$$\hat{D} = \begin{bmatrix} 6.2 & 0 & -4 & -3 \\ 0 & 7.1 & -2 & -1.5 \\ 6 & 7 & 8 & 0 \\ 0 & 0 & 1 & 10 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

And

$$D = \begin{bmatrix} 10 & 1 & 0 & 0 \\ 0 & 8 & 6 & 7 \\ 0 & 8 & 6 & 7 \\ -3 & -4 & 6.2 & 0 \\ -1.5 & -2 & 0 & 7.1 \end{bmatrix}$$

As a further check

$$A - DC = \begin{bmatrix} -10 & 1 & -1 & 0 & 0 \\ 0 & -1 & -8 & 0 & 0 \\ 0 & -1 & -8 & 0 & 0 \\ 0 & 0 & 0 & -7 & 0 \\ 0 & 0 & 0 & 0 & -8 \end{bmatrix}$$

We see that $(A - DC)\underline{g}_1 = -7\underline{g}_1$

$(A - DC)\underline{g}_2 = -8\underline{g}_2$

$(A - DC)\underline{g}_3 = -9\underline{g}_3$

$(A - DC)\underline{g}_4 = -10\underline{g}_4$

If the designer had decided not to accept the unassignable eigenvalue, the program can help him in selecting the subset of $\underline{b}_i$'s, in computing the Λ_i associated with each $\underline{b}_i$. In this case

$$\Lambda_1 = \{0\} \qquad \Lambda_2 = \{0\} \qquad \Lambda_3 = \emptyset \qquad \Lambda_4 = \emptyset$$

which means that

- with the set $\{\underline{b}_2, \underline{b}_3, \underline{b}_4\}$, the unassignable eigenvalue is 0
- with the set $\{\underline{b}_1, \underline{b}_3, \underline{b}_4\}$, the unassignable eigenvalue is 0
- with the set $\{\underline{b}_3, \underline{b}_4\}$ the unassignable eigenvalue is 0,
- the sets of events $\{\underline{b}_1, \underline{b}_2, \underline{b}_3\}$, $\{\underline{b}_1, \underline{b}_2, \underline{b}_4\}$,

$\{\underline{b}_1, \underline{b}_3\}$, $\{\underline{b}_1, \underline{b}_4\}$, $\{\underline{b}_2, \underline{b}_3\}$, $\{\underline{b}_2, \underline{b}_4\}$, $\{\underline{b}_1\}$, $\{\underline{b}_2\}$, $\{\underline{b}_3\}$, $\{\underline{b}_4\}$

have no unassignable eigenvalues associated with them.

CHAPTER IV

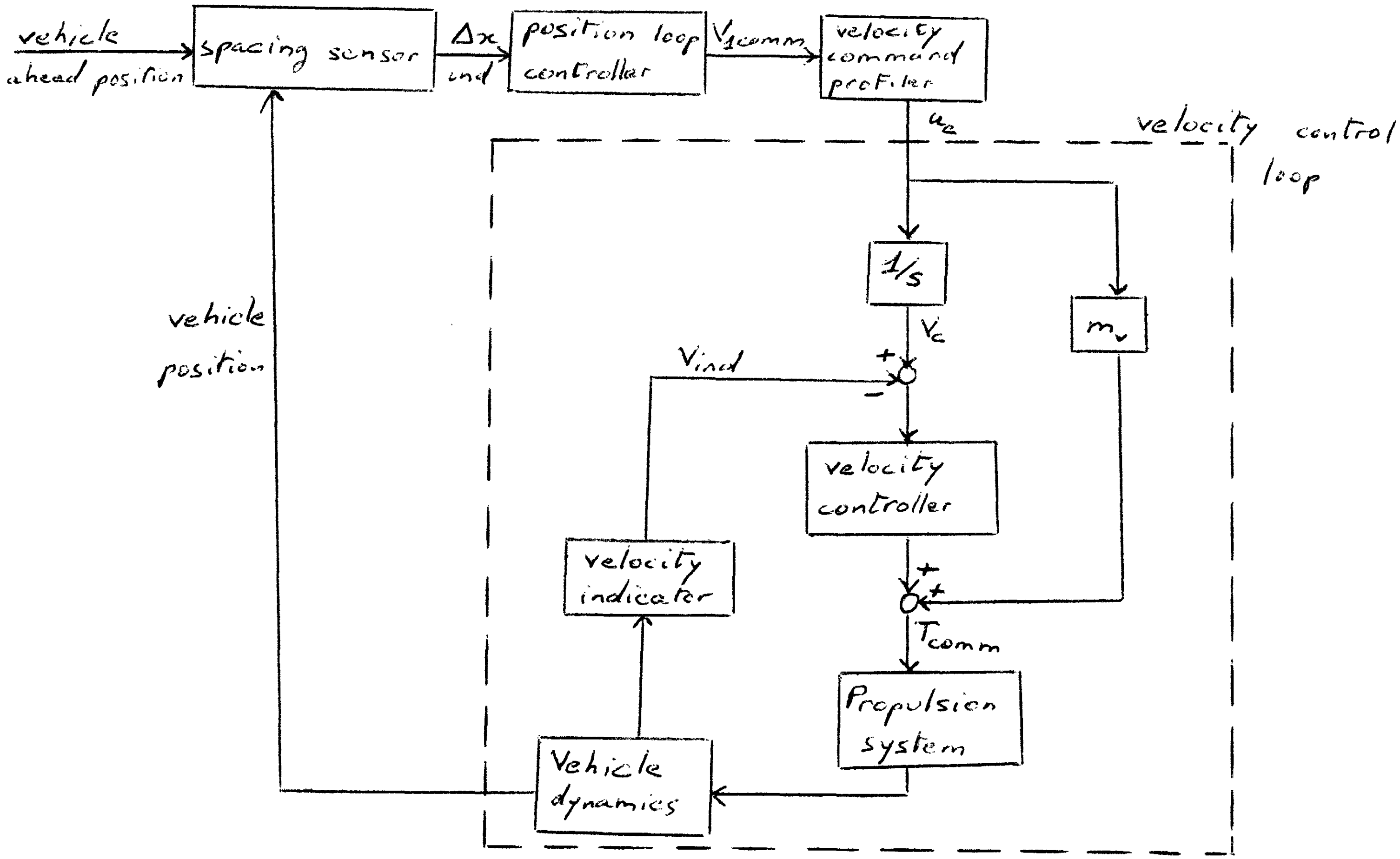
GUIDEWAY VEHICLE SIMULATION AND DETECTION FILTER DESIGN

(A) Background

It was determined that a study of the applicability of detection filter theory to guideway vehicle control would be most useful if it were done in the context of a typical guideway vehicle rather than in the context of any specific system. Two different approaches are possible: one where the spacing of different vehicles on the guideway is monitored by a wayside controller and one where each vehicle has an onboard spacing sensor which measures the distance from the vehicle ahead. Figure 4.1 and Fig. 4.2 show the block diagrams of the control systems with a spacing sensor and with a wayside controller.

The velocity profiler and the velocity control loop are common to both cases. To achieve a high level of reliability, it is preferable to implement a detection filter on board the vehicle, so that the filter could be used even in the case of a failure occurring in the communications between the wayside and the vehicle. In the case of a control system with a wayside controller, information on the spacing between the vehicle and the vehicle ahead is not available onboard if one does not wish to implement a special communication link for the detection filter only. Hence, it would not be possible to design a detection filter to monitor failures on the whole system.

Fig. 4.1 Vehicle-Follower System



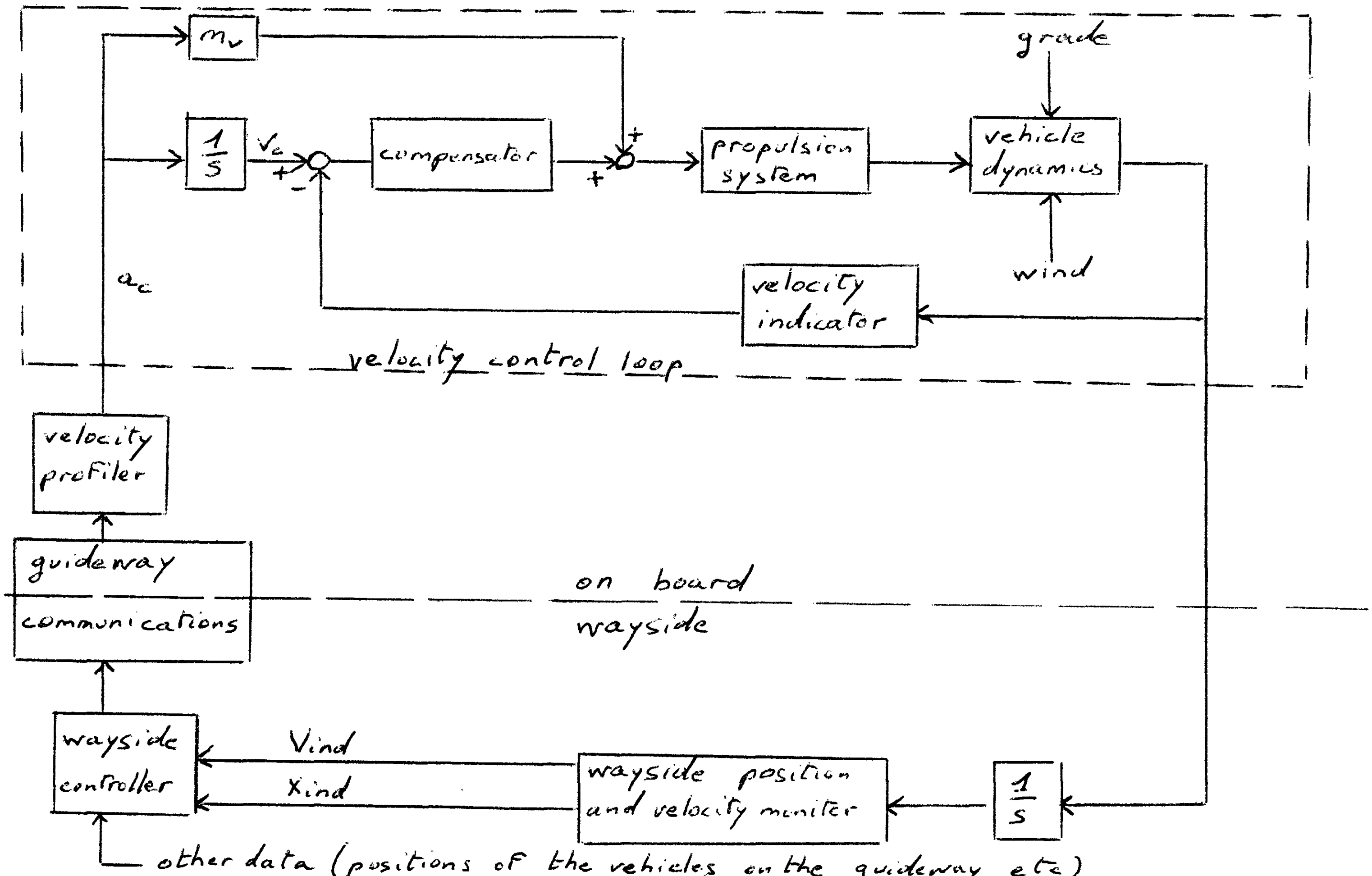


Fig. 4.2 Guideway vehicle control system with wayside controller

Only failures occurring in the velocity control loop could be monitored—as the velocity profiler, whose function is to limit the jerk and the acceleration commanded to the vehicle within bounds compatible with passenger comfort, is essentially non-linear and could not be accurately modeled in the linear reference model of the detection filter.

In the case of an autonomous vehicle-follower system, the position of the vehicle ahead is not available as a signal. The whole system cannot then be monitored by a detection filter. The velocity command profiler and the position loop controller would very likely be implemented in a digital computer, and the velocity command loop could be thought of as implemented with analog equipment in a preliminary feasibility study. In this case too, then, a detection filter would be designed only for the velocity control loop.

Figure 4.3 shows the velocity command loop, common to both systems; it is the part of the system for which a detection filter would be designed.

(B) Generic system parameters

We shall deal only with the components describing the velocity command loop. Figure 4.4 shows the component dynamics. The generic system parameters used were those found in a contractor documentation.

(1) Velocity indicator: onboard indication of vehicle velocity. No data given on noise. We used the following model:

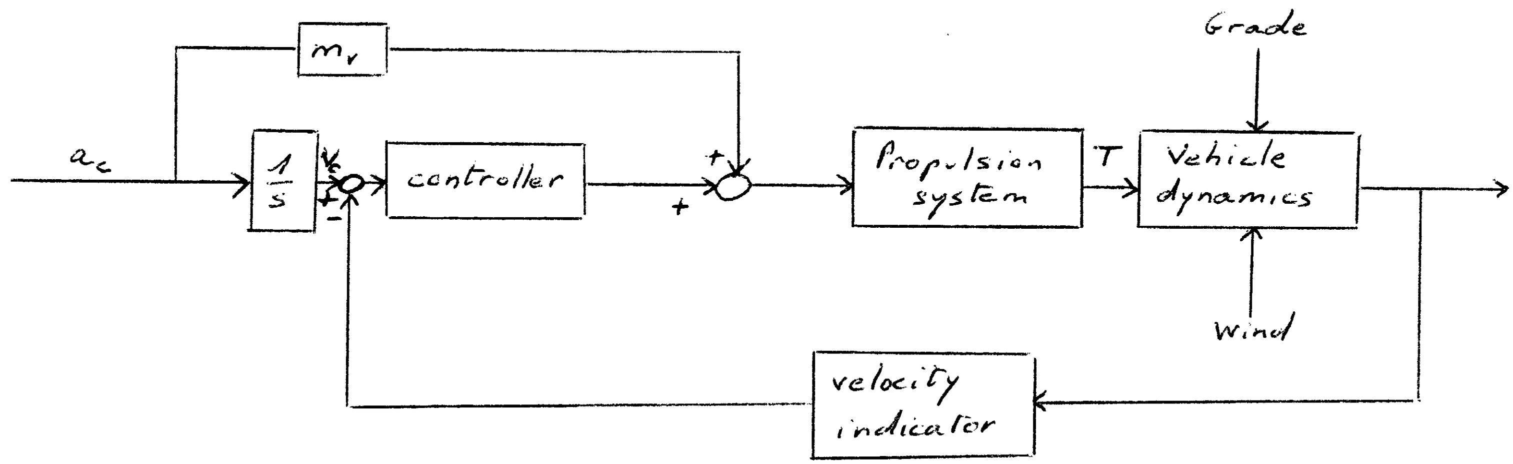


Fig. 4.3 Velocity control loop

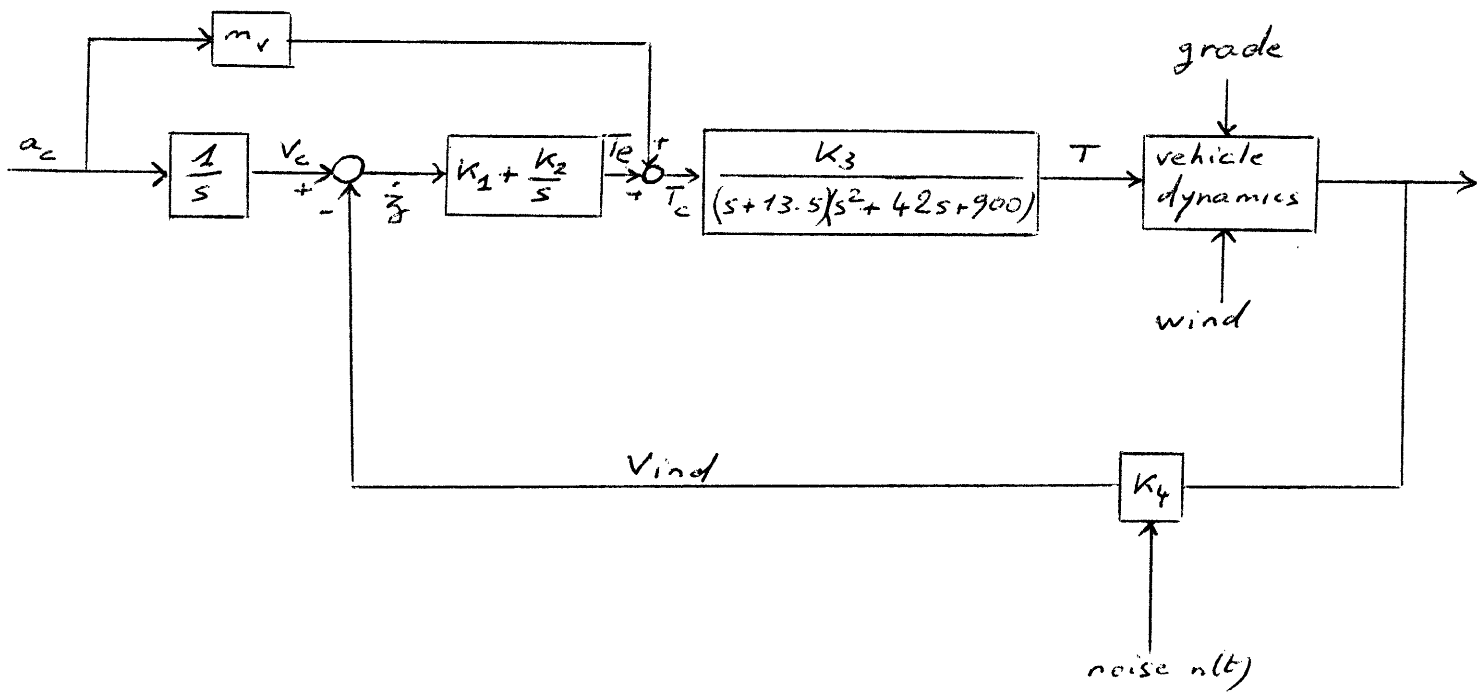
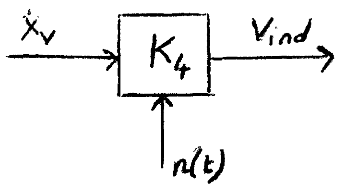


Fig. 4.4 Dynamics of the velocity control loop



$K_4 = 1.2$

$V_{ind} = K_4[\dot{x}_v + n(t)]$

(Appendix C shows why a value of 1 is not advisable for the detection filter design).

(2) Vehicle dynamics $\quad m = F_p + F_C + F_G + F_A$

where

F_p = propulsion system force = T

F_C = Coulomb friction force = $-100 \cdot \mathrm{sgn}(\dot{X}_v)$ lbs
$= - f_c \ \mathrm{sgn}\ (\dot{X}_v)$

F_G = grade force

$= -mg \dfrac{\%\ \text{grade}}{100}$ lbs, up to 6%

F_A = aerodynamic forces

$= -(0.03)(\dot{X}_v - V_w)|\dot{X}_v - V_w|$ lbs $= -c_{aero}\ (\dot{X}_v - V_w)|\dot{X}_v - V_w|$

with V_w = wind velocity in ft/sec

(3) Compensator: proportional + integral

$K_1 + K_2/s$ with $K_1 = 1500 \quad K_2 = 1000$ (Units lbs ft sec)

(4) Propulsion system

The propulsion system is modeled with a 3rd order dynamics transfer function.

$$\frac{T}{T_{comm}} = \frac{1}{(1 + \frac{s}{13.5})(1 + \frac{2(.7)s}{30} + \frac{s^2}{30^2})}$$

We shall use the form

$$T = a_2\ T + a_3\ T + a_4\ T + K_3\ T_{comm}$$

with

$a_2 = -55.5 \qquad a_4 = -12150$

$a_3 = -1467 \qquad K_3 = 12150$

(5) Acceleration feedforward

The commanded acceleration is fed forward through a gain m_v, which should be equal to the mass of the vehicle. More realistically, in this simulation, m and m_v are not equal, but close numbers: $m = 373 \quad m_v = 350$ (slugs)

(C) System equations

The variables used are:

T: realized thrust, $\dot{T}$, $\ddot{T}$,

x_v: vehicle position

$\dot{x}_v$: vehicle velocity

$\dot{z}$: input to the compensator ($\dot{z} = v_c - v_{ind}$)

The inputs are

a_c acceleration command

$(\dot{x}_v - V_w)|\dot{x}_v - V_w|$ where V_w is the wind velocity

$g \sin\theta$ grade effect

n(t) noise

The outputs (later compared with those of the filter simulation) are

T_c (commanded thrust)

V_{ind}

These were the only physically accessible signals

We have the relations

(1) $(x_v)^{\cdot} = \dot{x}_v$

(2) $$(\dot{x}_v)^{\cdot} = \frac{T}{m} - \frac{f_c}{m}\frac{\dot{X}_v}{|\dot{X}_v|} - \frac{c_{aero}}{m}(\dot{X}_v - V_w)|\dot{x}_v - V_w| - g \sin\theta$$

where θ is the grade

(3) $(T)^{\cdot} = \dot{T}$

(4) $(\dot{T})^{\cdot} = \ddot{T}$

$$\dddot{T} = a_2 \ddot{T} + a_3 \dot{T} + a_4 T + K_3 T_c$$

But

$$T_c = T_e + m_v\, a_c$$

$$= K_1\, \dot{z} + K_2\, z + m_v a_c$$

$$= K_1 (V_c - V_{ind}) + K_2\, z + m_v a_c$$

$$= K_1 (V_c - K_4 (\dot{x}_v + n(t)) + K_2\, z + m_v a_c$$

Then

(5) $$\dddot{T} = - K_1 K_3 K_4\, \dot{X}_v + a_4 T + a_3 \dot{T} + a_2 \ddot{T} + K_2 K_3\, z + K_1 K_3\, V_c + K_3 m_v a_c - K_1 K_3 K_4\, n(t)$$

(6) $$\dot{z} = V_c - V_{ind} = V_c - K_4\, \dot{x}_v - K_4 n(t)$$

(7) $$\dot{V}_c = a_c$$

In matrix form this gives

$$\begin{pmatrix} \dot{x}_v \\ \ddot{x}_v \\ \dot{T} \\ \ddot{T} \\ \dddot{T} \\ \dot{z} \\ \dot{V}_c \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/m & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & -K_1K_3K_4 & a_4 & a_3 & a_2 & K_2K_3 & K_1K_3 \\ 0 & -K_4 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} x_v \\ \dot{x}_v \\ T \\ \dot{T} \\ \ddot{T} \\ z \\ V_c \end{pmatrix}$$

$$+ \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ -f_c/m & 0 & -\frac{C_{aero}}{m} & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & K_3 m_v & 0 & 0 & -K_1K_3K_4 \\ 0 & 0 & 0 & 0 & -K_4 \\ 0 & 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \frac{\dot{x}_v}{|\dot{x}_v|} \\ a_c \\ (\dot{x}_v - v_w)|\dot{x}_v - v_w| \\ g \sin\theta \\ n(t) \end{pmatrix}$$

(8) $T_c = -K_1K_4 \dot{x}_v + K_2z + K_1V_c + m_va_c - K_1K_4 n(t)$

(9) $V_{ind} = K_4\dot{X}_v + K_4 n(t)$

In matrix form

$$\begin{pmatrix} T_c \\ V_{ind} \end{pmatrix} = \begin{pmatrix} 0 & -K_1K_4 & 0 & 0 & 0 & K_2 & K_1 \\ 0 & K_4 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} X_v \\ \dot{X}_v \\ T \\ \dot{T} \\ \ddot{T} \\ z \\ V_c \end{pmatrix}$$

$$+ \begin{pmatrix} 0 & m_v & 0 & 0 & -K_1K_4 \\ 0 & 0 & 0 & 0 & K_4 \end{pmatrix} \begin{pmatrix} \frac{\dot{X}_v}{|\dot{X}_v|} \\ a_c \\ (\dot{X}_v - V_w)|\dot{X}_v - V_w| \\ g \sin\theta \\ n(t) \end{pmatrix}$$

Figure 4.5 shows the action of the velocity profiler on a way-side velocity command, to keep the jerk and the acceleration under values compatible with passenger comfort.

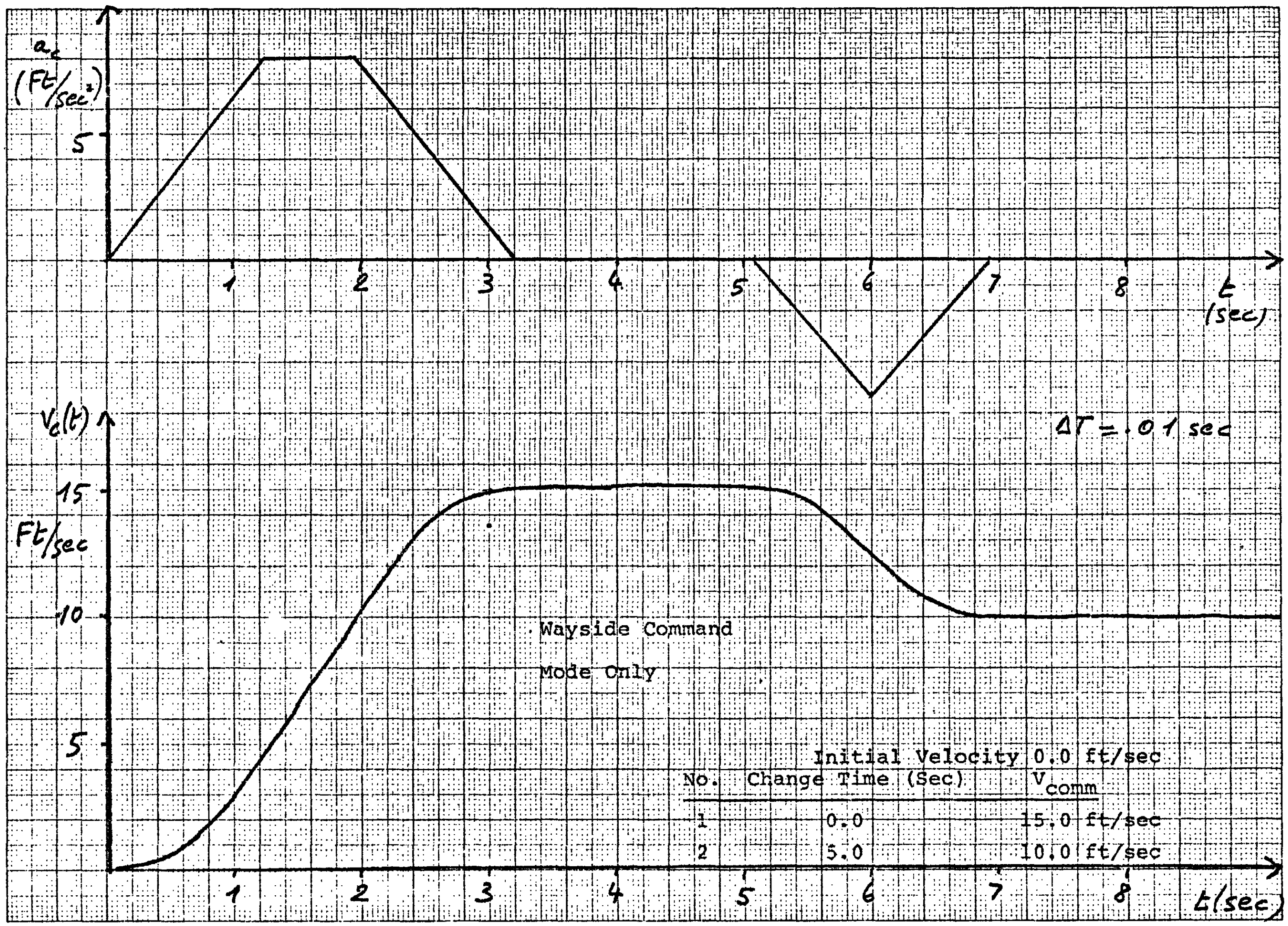
Figure 4.6 shows the response of the system in velocity regulation mode, plotting the velocity error versus time, V_c given by Fig. 4.5.

(D) Reference model of the system for the detection filter

The elements whose failures were to be monitored by a detection filter were: (see Fig. 4.3)

- the controller
- the velocity indicator
- the propulsion system.

Fig. 4.5



a_c (Ft/sec²)
5
1 2 3 4 5 6 7 8
t (sec)
$V_c(t)$
15
Ft/sec
10
5
ΔT = .01 sec
Wayside Command
Mode Only
Initial Velocity 0.0 ft/sec
No. Change Time (Sec) V comm
1 0.0 15.0 ft/sec
2 5.0 10.0 ft/sec
1 2 3 4 5 6 7 8
t (sec)

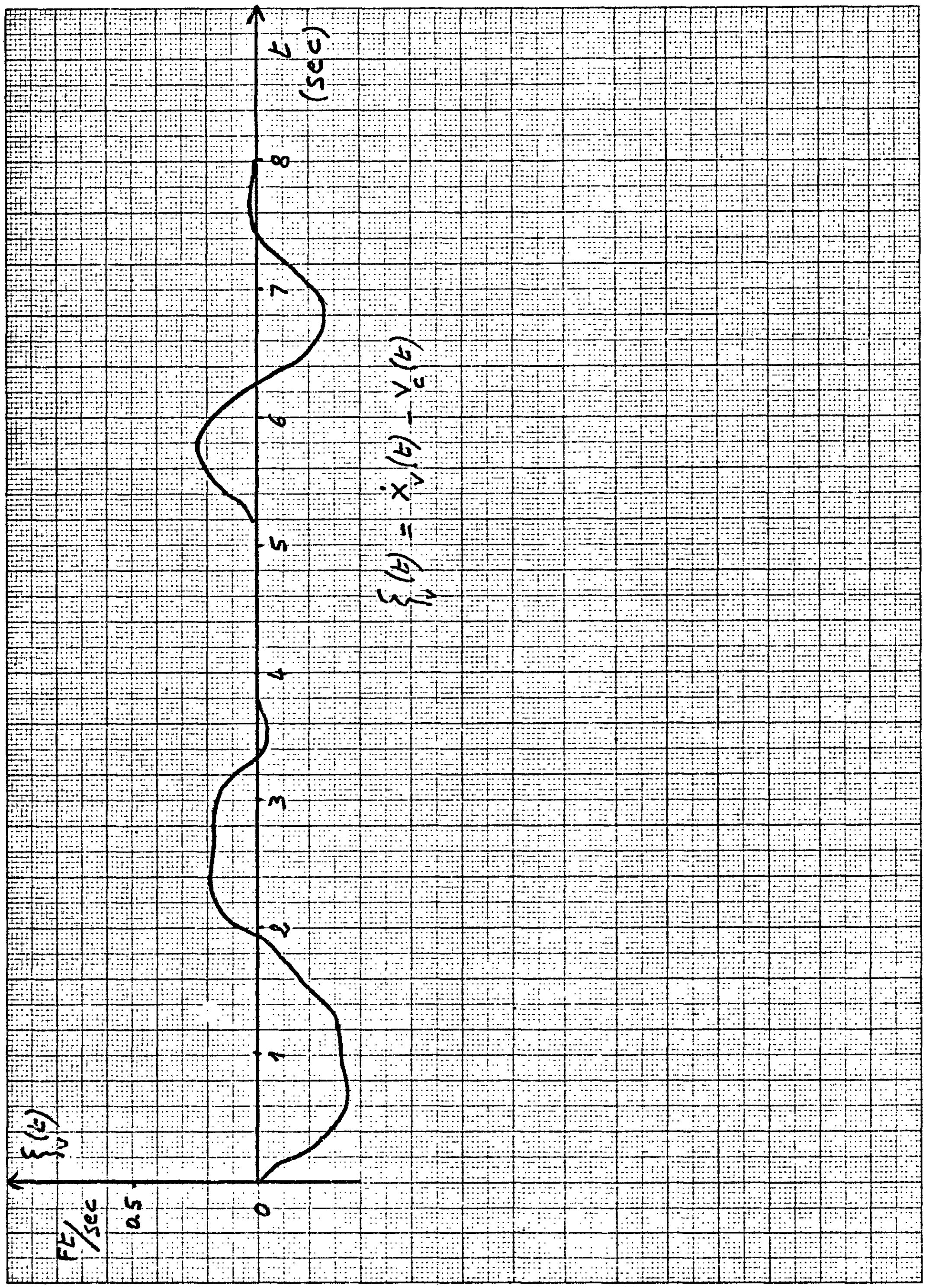
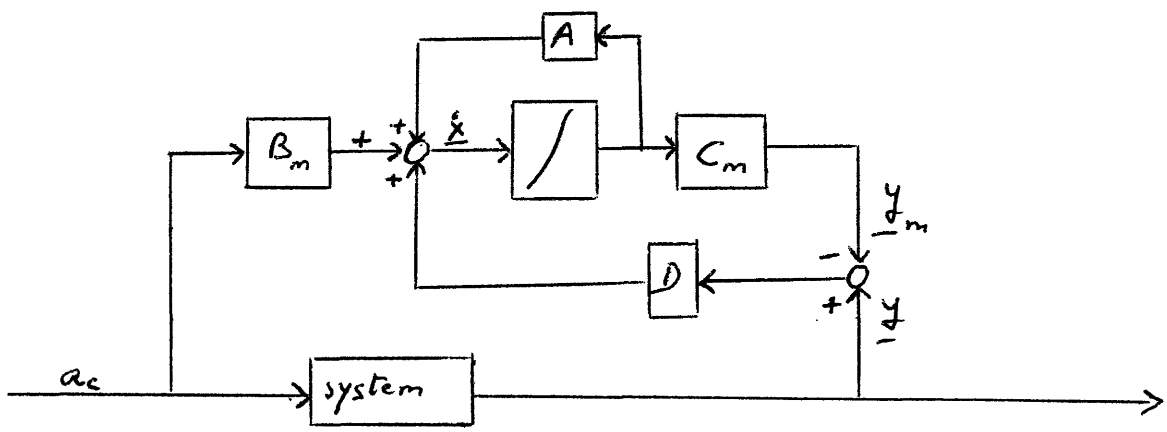


Fig. 4.6

The two first elements, on a real system, would be duplicated, and the system would switch to the backup component in case of a failure indicated by the detection filter. The propulsion system would be made of two parallel modules; in case of a failure, the vehicle would continue with half of its propulsion capability. The velocity control loop functional block diagram would be in fact the one indicated by Fig. 4.7

The detection filter works only on a linear model. We have



To derive the matrices A_m, B_m, C_m for the reference model of the detection filter, all the nonlinearities were neglected in the system, the grade component was ignored, and the simplified model of Fig. 4.8 was used (notice that the Coulomb frinction was modeled in the reference model as a bias).

The states selected are $\dot{X}_V$, T, $\dot{T}$, $\ddot{T}$, T_e, V_c

(vehicle velocity, thrust, first and second derivatives of thrust, compensator output and velocity command)

The outputs are T_c and V_{ind}

We have the relations

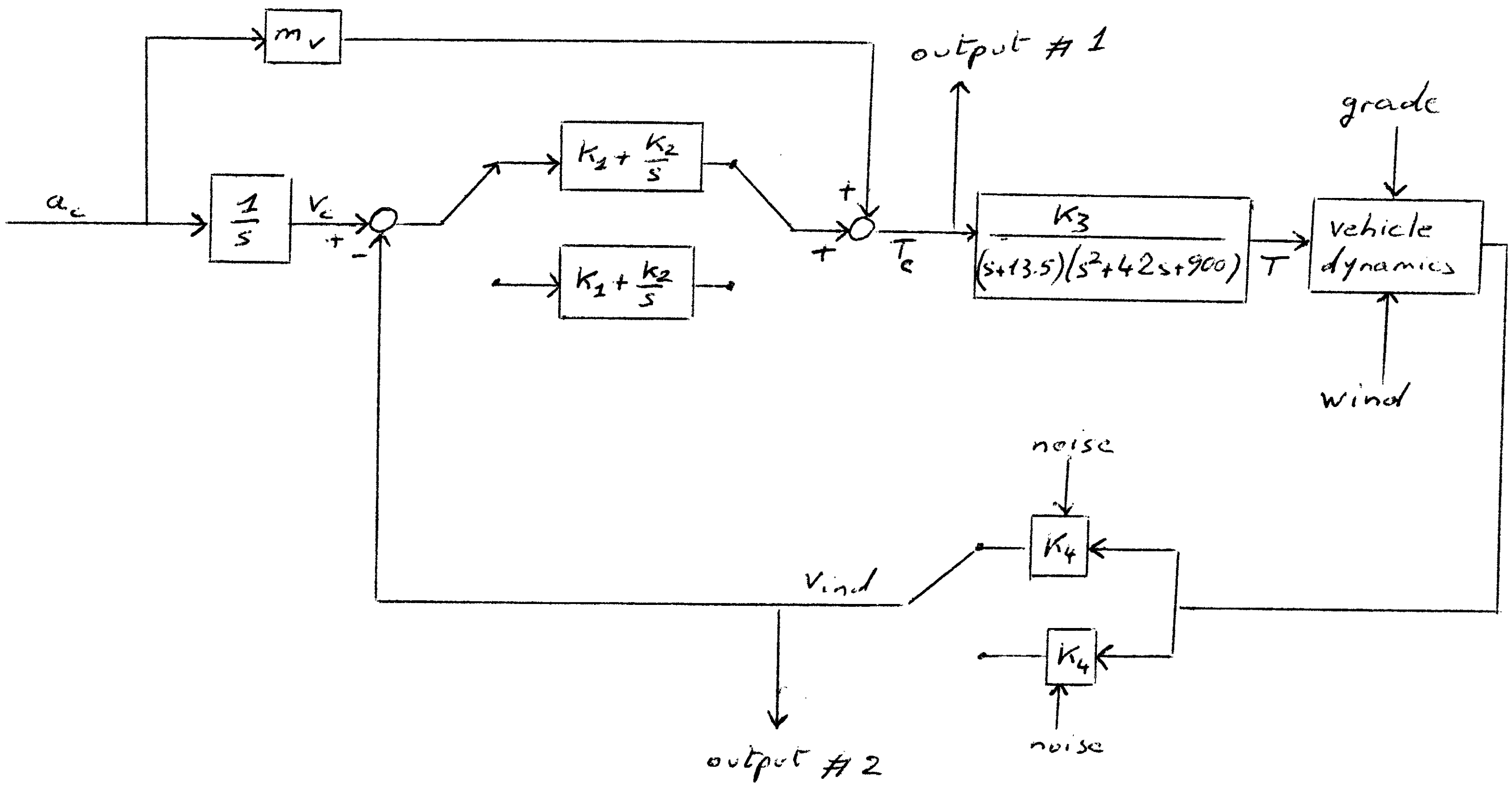


Fig. 4.7 Redundancies in velocity control loop

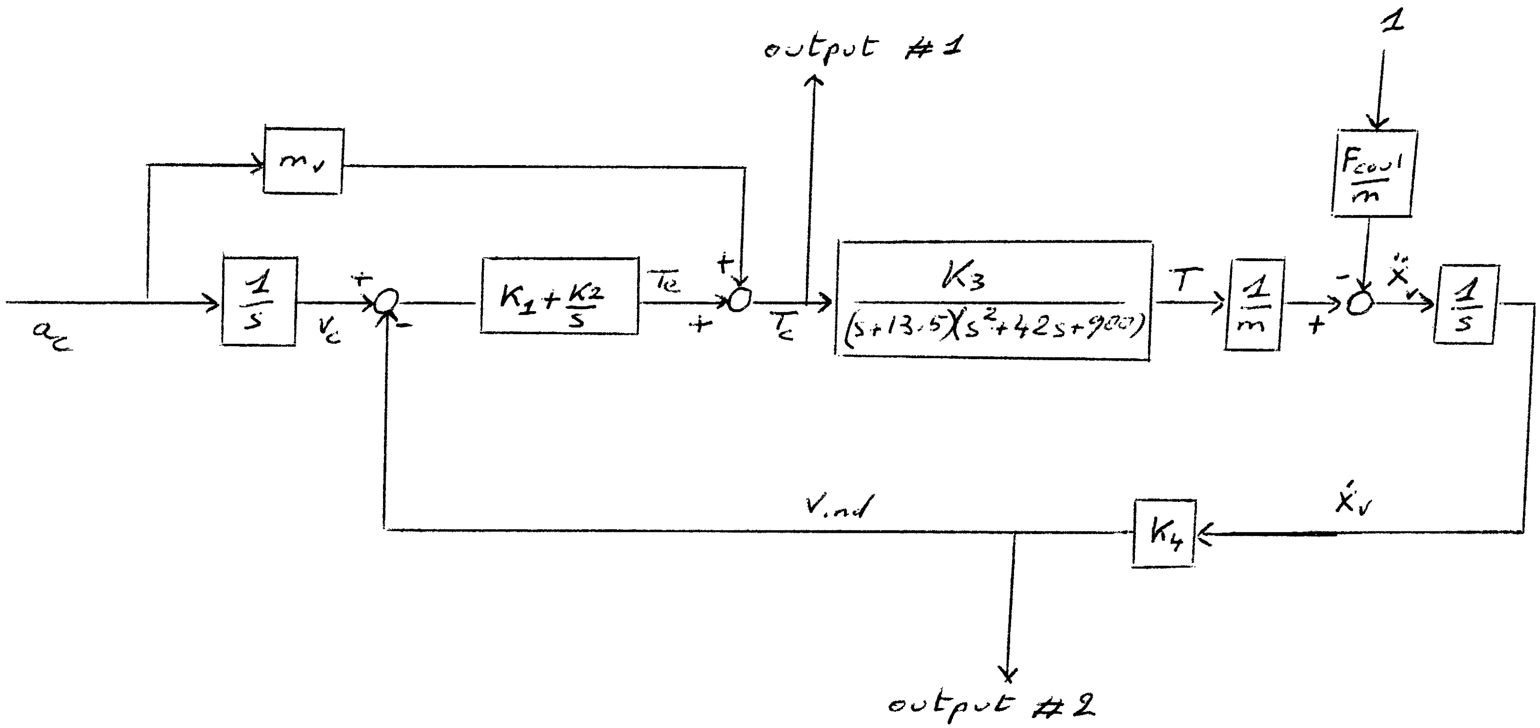


Fig. 4.8 Detection filter reference model

$$T_c = T_c + m_v a_c$$

$$\dot{T}_e = K_1(\dot{V}_c - \dot{V}_{ind}) + K_2(V_c - V_{ind})$$

$$= K_1(a_c - K_4\ddot{X}_v) + K_2(V_c - V_{ind})$$

$$= K_1 a_c - K_1 K_4 \frac{T}{m} + K_1 K_4 \frac{F_{coul}}{m} + K_2 V_c$$

$$(\dot{X}_v)' = \frac{T}{m} - \frac{F_{coul}}{m}$$

$$(T)' = \dot{T}$$

$$(\dot{T})' = \ddot{T}$$

$$(\ddot{T})' = K_3 T_c + a_2\ddot{T} + a_3\dot{T} + a_4 T$$

$$= K_3\, T_e + K_3 m_v a_c + a_2\ddot{T} + a_3\dot{T} + a_4 T$$

$$(\dot{V}_c) = a_c$$

In matrix form

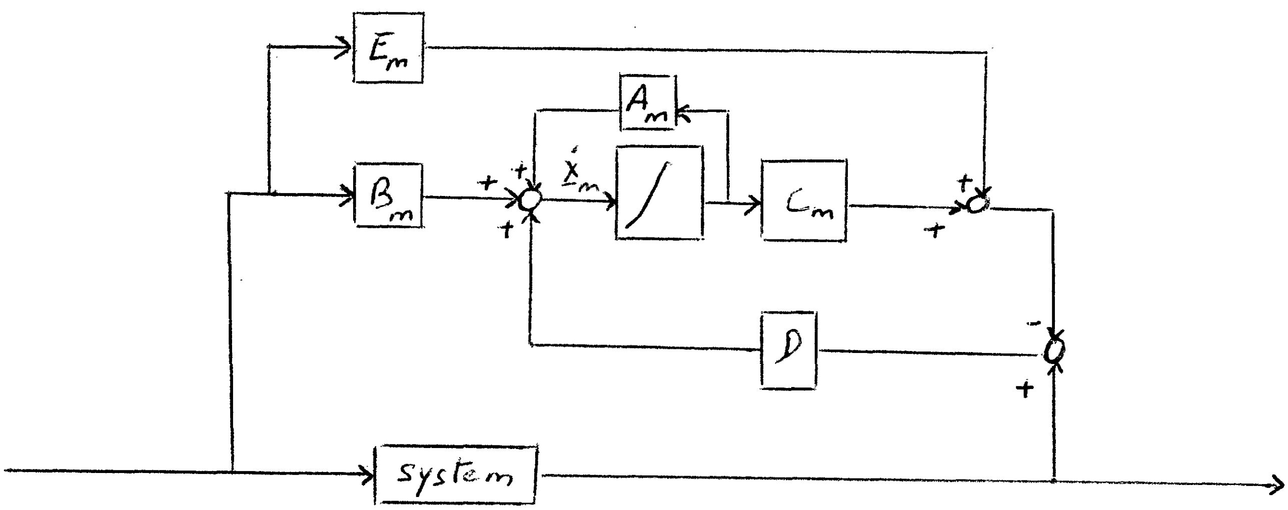
$$\begin{pmatrix} \ddot{X}_v \\ \dot{T} \\ \ddot{T} \\ \dddot{T} \\ \dot{T}_e \\ \dot{V}_c \end{pmatrix} = \begin{pmatrix} 0 & 1/m & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & a_4 & a_3 & a_2 & K_3 & 0 \\ -K_2K_4 & -\frac{K_1K_4}{m} & 0 & 0 & 0 & K_2 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \dot{X}_v \\ T \\ \dot{T} \\ \ddot{T} \\ T_e \\ V_c \end{pmatrix} + \begin{pmatrix} 0 & -\frac{F_{coul}}{m} \\ 0 & 0 \\ 0 & 0 \\ K_3 m_v & 0 \\ K_1 & K_1K_4\frac{F_{coul}}{m} \\ 1 & 0 \end{pmatrix} \begin{pmatrix} a_c \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} T_c \\ V_{ind} \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 \\ K_4 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \dot{X}_v \\ T \\ \dot{T} \\ \ddot{T} \\ T_e \\ V_c \end{pmatrix} + \begin{pmatrix} m_v & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} a_c \\ 1 \end{pmatrix}$$

This is not of the form
$$\begin{cases} \dot{\underline{x}}_m = A_m \underline{x}_m + B_m u \\ \underline{y}_m = C \underline{x}_m \end{cases}$$

There is a matrix E_m such that
$$\begin{cases} \dot{\underline{x}}_m = A_m \underline{x}_m + B_m \underline{u} \\ \underline{y}_m = C_m \underline{x}_m + E_m \underline{u} \end{cases}$$

The system configuration, with the filter, is in fact



We have then the equations

$$\dot{\underline{x}} = A\,\underline{x} + B\,\underline{u} \qquad \underline{y} = C\,\underline{x} + E\,\underline{u}$$

$$\dot{\underline{x}}_m = A_m\underline{x}_m + B_m\underline{u} + D(\underline{y} - \underline{y}_m) \qquad \underline{y}_m = C_n\underline{x}_m + E\,\underline{u}$$

In the absence of failures, $A \equiv A_n \quad B \equiv B_m \quad C \equiv C_m$

Then $\quad \dot{\underline{x}} - \dot{\underline{x}}_m = \dot{\underline{\varepsilon}} = A\underline{x} + B\underline{u} - A\underline{x}_m - B\underline{u} - D(C\underline{x} + E\underline{u} - C\underline{x}_m - E\underline{u})$

Then $\quad \dot{\underline{\varepsilon}} = (A - DC)\underline{\varepsilon}$ in the absence of failure. So long as the failures considered entail no variation in the E matrix, the turns $DE\underline{u}$ and $DE_m\underline{u}$ simplify out in the equation of $\dot{\underline{\varepsilon}}$, and the theory of the detection filter, as it was presented in Chapter 2, is applicable without modification to this case. It would be applicable too in the case where, even if a failure caused a modification in E, it could be modeled as a sensor failure.

In this case,

$$E_m = \begin{pmatrix} m_v & 0 \\ 0 & 0 \end{pmatrix}$$

As a failure in m_v is not to be monitored, this problem does not arise.

Note: In fact, even in the absence of failures, we do not have $A \equiv A_m$ $B \equiv B_m$ $C \equiv C_m$ because the simulation of the system, represented by A_m, B_m, C_m is linearized, and because the grade component is ignored. This study was partly made to check that the neglected nonlinearities and the grade component produced outputs along the C $\underline{b}_i$'s associated with the failures to be monitored much smaller than the failure outputs.

(E) Detection filter design

The first step in the detection filter design is to derive the event vectors associated with the failures to be monitored. Failures in compensator, motor, tachometer, can be modeled as failures in K_1, K_2, K_3, K_4.

Event associated with a failure in K_1.

If K_1 fails and becomes arbitrarily time-varying, $K_1 k_1(t)$ being its new value, the system equations become

$$\begin{cases} \dot{\underline{x}} = A\underline{x} + B\underline{u} + \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \left(-\frac{Tk_4}{m} + a_c + K_4 \frac{F_{coul}}{m} \right) \left(k_1(t) - 1 \right) K_1 \\ \underline{y} = C\underline{x} + E\underline{u} \end{cases}$$

The event associated with a failure in K_1 is then $\underline{e}_{65} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$

We have $C\underline{e}_{65} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$

Event associated with a failure in K_2. If K_2 fails and becomes arbitrarily time-varying, its new value being $K_2k_2(t)$, the system equations become

$$\begin{cases} \dot{\underline{x}} = A\underline{x} + B\underline{u} + \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \left(-K_4 \dot{x}_v + v_c \right) \left(k_2(t) - 1 \right) K_2 \\ \underline{y} = C\underline{x} + E\underline{u} \end{cases}$$

The event associated with a failure in K_2 is $\underline{e}_{65}$.

Event associated with a failure in K_3. If K_3 fails and becomes arbitrarily time-varying, its new value being $K_3k_3(t)$, the system equations become

$$\begin{cases} \dot{\underline{x}} = A\underline{x} + B\underline{u} + \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \left(T\varrho + m_v a_c \right) \left(k_3(t) - 1 \right) K_3 \\ \underline{y} = C\underline{x} + E\underline{u} \end{cases}$$

The event associated with a failure in K_3 is $\underline{e}_{64} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$

But, $C\,\underline{e}_{64} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$

In this case, we shall use the first vector of the series $A\,\underline{e}_{64}$, $A^2\,\underline{e}_{64}, \ldots, A^n\,\underline{e}_{64}$ for which $C\,A^i\,\underline{e}_{64} \neq \begin{pmatrix} 0 \\ 0 \end{pmatrix}$

$$A\underline{e}_{64} = \begin{pmatrix} 0 \\ 0 \\ 1 \\ a_2 \\ 0 \\ 0 \end{pmatrix} \qquad CA\,\underline{e}_{64} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$A^2\underline{e}_{64} = \begin{pmatrix} 0 \\ 1 \\ a_2 \\ a_3 + a_2^2 \\ 0 \\ 0 \end{pmatrix} \qquad CA^2\underline{e}_{64} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$A^3\underline{e}_{64} = \begin{pmatrix} 1/m \\ a_2 \\ a_3 + a_2^2 \\ a_4 + 2a_3a_2 + a_2^3 \\ -\dfrac{K_1K_4}{m} \\ 0 \end{pmatrix} \qquad CA^3\underline{e}_{64} = \begin{pmatrix} -\dfrac{K_1K_4}{m} \\ \dfrac{K_4}{m} \end{pmatrix}$$

The event associated with K_3 is then $\begin{pmatrix} 1/m \\ a_2 \\ a_3 + a_2^2 \\ a_4 + 2a_3a_2 + a_2^3 \\ -\dfrac{K_1K_4}{m} \\ 0 \end{pmatrix}$

Events associated with a failure in K_4. If K_4 fails and becomes arbitrarily time-varying, its new value being $K_4 k_4(t)$, the system equations become

$$\begin{cases} \dot{\underline{x}} = A\underline{x} + B\underline{u} + \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \left(-K_2 \dot{x}_v - K_1 \frac{T}{m} + K_1 \frac{F_{coul}}{m}\right)(k_4(t) - 1) K_4 \\ \underline{y} = C\underline{x} + E\underline{u} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} (\dot{x}_v)(k_4(t) - 1) K_4 \end{cases}$$

Let us call

$$\underline{e}_{65} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \qquad \underline{e}_{22} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

$$n_5(t) = \left(-K_2 \dot{x}_v - K_1 \frac{T}{m} + K_1 \frac{F_{coul}}{m}\right)(k_4(t) - 1) K_4$$

$$n_{c2}(t) = \dot{x}_v (k_4(t) - 1) k_4$$

The system equations, after failure in K_4, can be rewritten

$$\begin{cases} \underline{x} = A\underline{x} + B\underline{u} + n_5(t)\, \underline{e}_{65} \\ \underline{y} = C\underline{x} + E\underline{u} + n_{c2}(t)\, \underline{e}_{22} \end{cases}$$

The error equations will then be

$$\dot{\underline{\xi}} = (A - DC)\underline{\xi} + n_5(t)\, \underline{e}_{65} - \underline{d}_2\, n_{c2}(t)$$

$$\underline{\xi}' = C\underline{\xi} + n_{c2}(t)\, \underline{e}_{22}$$

where $\underline{d}_2$ is the 2nd column of the filter D

$\underline{e}_{22}$ corresponds to a variation in C. Then (see ref 2 page 193-194), the error output of a failure associated with $\underline{e}_{22}$ can only be contained to the plane spanned by $(C\underline{f}, CA\underline{f})$ where $\underline{f}$ is such that $C\underline{f} = \underline{e}_{22}$.

If

$$\underline{e}_{61} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

we notice that $C\underline{e}_{61} = \begin{pmatrix} 0 \\ K_4 \end{pmatrix} = K_4 \underline{e}_{22}$

A failure in K_4 cannot be modeled under a controller failure model or a sensor model failure, because it involves variations in C and in A. It can be seen as the superposition of a sensor failure and a controller failure.

The controller failure will be represented by the event $\underline{e}_{65}$. The sensor failure will be represented by the couple of events $(\underline{e}_{61}, A\,\underline{e}_{61})$. However

$$A\,\underline{e}_{61} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ -K_2 K_4 \\ 0 \end{pmatrix} = -K_2 K_4 \,\underline{e}_{65}$$

then

$A\,\underline{e}_{61}$ is parallel to $\underline{e}_{65}$

In short, we have

- event associated with failure in K_1 is $\underline{e}_{65} = \underline{b}_1$
- event associated with failure in K_2 is $\underline{e}_{65} = \underline{b}_1$

- event associated with failure in K_3 is $\underline{b}_2 = \begin{pmatrix} 1/m \\ a_2 \\ a_3 + a_2^2 \\ a_4 + 2a_3 a_2 + a_2^3 - k_1 k_4/m \\ 0 \end{pmatrix}$

- events associated with failure in K_4 are $\underline{e}_{65}$ and $\underline{e}_{61}$

It appears that

- failures in K_1 and K_2 are detection equivalent. A filter cannot be designed which will distinguish between the two failures. At most, it would allow to determine if the compensator has failed, but not which part of the compensator has failed.
- A failure in K_4 cannot be constrained to generate unidirectional outputs. In this case, error output can only be constrained to the plane spanned by ($C\,\underline{e}_{65}$, $C\,\underline{e}_{61}$). C is of dimension 2; this means that a failure in K_4 will span the whole output space ($C\,\underline{e}_{65}$ and $C\,\underline{e}_{61}$ are independent).

It was decided to design a detection filter for the events $\underline{b}_1$ and $\underline{b}_2$. A failure in K_1 or K_2 would generate outputs along $C\underline{b}_1$, a failure in K_3 along $C\underline{b}_2$, a failure in K_4 along $C\underline{b}_1$ and $C\underline{b}_2$. This way, a filter could distinguish among failures in the three elements to be monitored (Using more logic, it is possible to distinguish between failures in K_4 and other failures which would generate outputs along $C\underline{b}_1$ and $C\underline{b}_2$: both tachometers could be used in parallel, with only the output of one fed back to the velocity controller. The two outputs would be compared, a failure would be declared if the difference between the two did not stay in an allowable margin. The detection filter could then identify the faulting tachometer.)

The system for which the filter is to be designed is represented by

$$A = \begin{pmatrix} 0 & .2681 \times 10^{-2} & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & -12150. & -1466.5 & -55.5 & 12150. & 0 \\ -1200 & -4.826 & 0 & 0 & 0 & 1000 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$B = \begin{pmatrix} 0 & -.268 \\ 0 & 0 \\ 0 & 0 \\ 4.25 \times 10^{5} & 0 \\ 1500 & 482.6 \\ 1 & 0 \end{pmatrix} \qquad C = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 \\ 1.2 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

$$E = \begin{pmatrix} 350 & 0 \\ 0 & 0 \end{pmatrix}$$

$$\underline{b}_1 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \qquad \underline{b}_2 = \begin{pmatrix} .268 \times 10^{-2} \\ -55.5 \\ 1613\ 75 \\ -20\,322.37 \\ -4.826 \\ 0 \end{pmatrix}$$

Using the computer program developed from the algorithm presented in Chapter 3, it was found that:

$R_{\mathcal{S}}$ has a dimension 4

R_1 has a dimension 1

R_2 has a dimension 3

$$\underline{g}_1 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \qquad \underline{g}_2 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

Then $\bar{R}_{\mathcal{S}}$ has a dimension 6

$\bar{R}_1$ has a dimension 2

$\bar{R}_2$ has a dimension 4

$\mathcal{V}_1 + \mathcal{V}_2 = \mathcal{V}_{\mathcal{S}}$ The events $\underline{b}_1$ and $\underline{b}_2$ are mutually detectable. We assigned the eigenvalues - 10 and - 10 to $\bar{R}_1$, - 10, - 10, - 10, - 10 to $\bar{R}_2$

The filter computed was

$$D = \begin{pmatrix} 0 & -12.9166 \\ 0 & -1942.937 \\ 0 & 4\,639\,965. \\ 12150 & -193\,023\,664 \\ 20 & 52\,252.0625 \\ 0.1 & 150. \end{pmatrix}$$

CHAPTER 5

EXPERIMENTAL RESULTS

According to the detection filter theory, a failure in K_1, K_2, or K_3 should produce unidirectional outputs along $C\underline{b}_1$ and $C\underline{b}_2$, where $\underline{b}_1$ is the event associated with a failure in K_1 or K_2, and $\underline{b}_2$ an event associated with a failure in K_3.

From the comparison between the outputs of the guideway vehicle simulation and of the guideway vehicle reference model, one has access to $\underline{y} - \underline{y}_m$ whose two components are respectively $T_c - T_{cm}$ and $V_{ind} - V_{ind_m}$. This vector of the output space is expressed in the basis $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$. In this basis, $C\underline{b}_1$ has the value $\begin{pmatrix} 1.2 \\ 0 \end{pmatrix}$ and $C\underline{b}_2 = \begin{pmatrix} -4.826 \\ 3.217\times 10^{-3} \end{pmatrix}$. It is necessary to change the basis of the output space to measure the error outputs along $C\underline{b}_1$ and $C\underline{b}_2$. The new basis $\left(\frac{1}{1.20} C\underline{b}_1, C\underline{b}_2\right)$ has the components $\begin{pmatrix} 1 & -4.826 \\ 0 & 3.217\times 10^{-3} \end{pmatrix}$ along the old one. We have then $\underline{x}_{new} = P^{-1}\underline{x}_{old}$ with

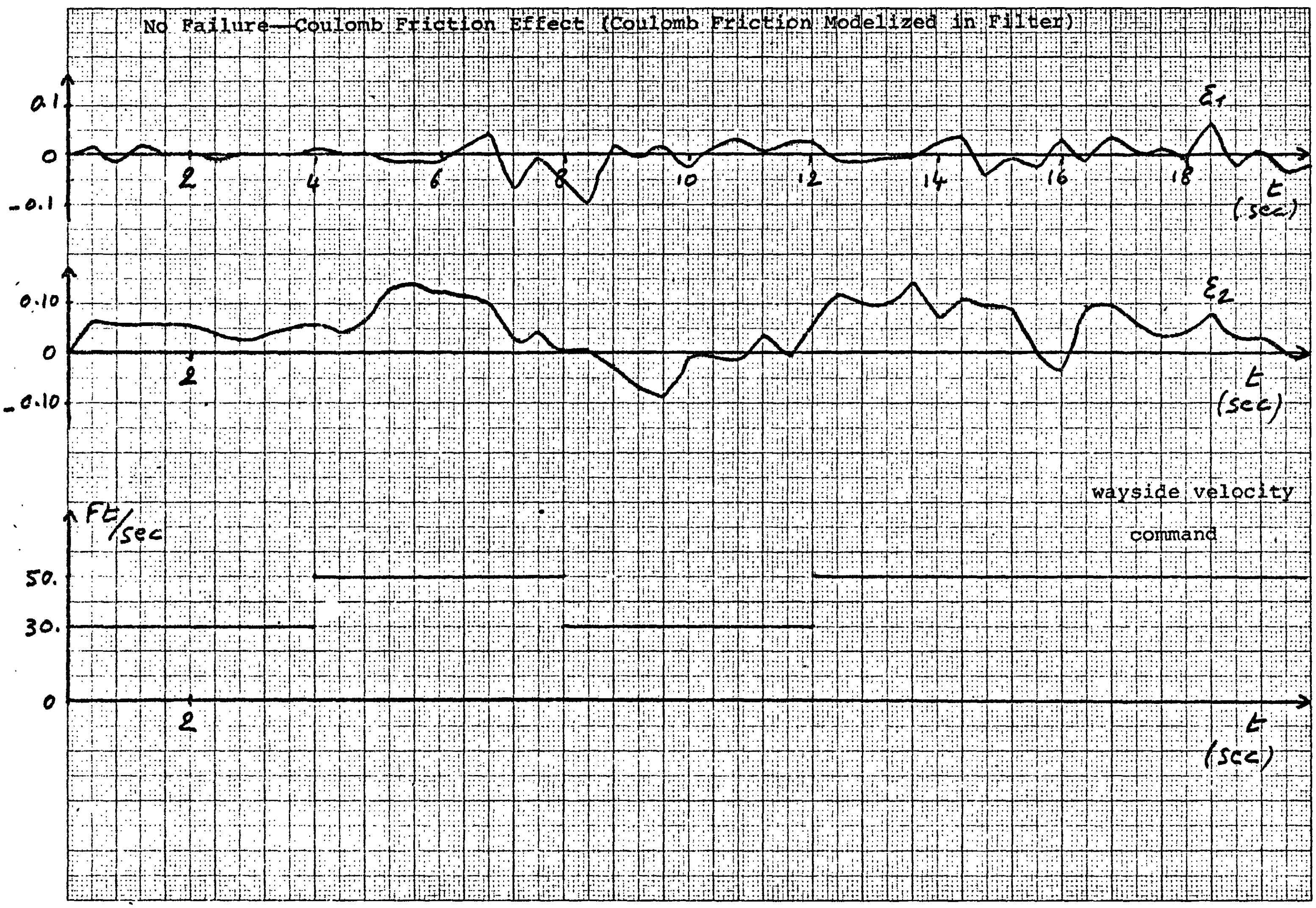
$$P = \begin{pmatrix} 1 & -4.826 \\ 0 & 3.217\times 10^{-3} \end{pmatrix} \quad \text{Hence } P^{-1} = \begin{pmatrix} 1 & 1500 \\ 0 & 310.83 \end{pmatrix}$$

In the following plots, $\mathcal{E}_1 = [P^{-1}(\underline{y}-\underline{y}_m)]_1$ is the error output along $\frac{1}{1.20} C\underline{b}_1$. $\mathcal{E}_2 = [P^{-1}(\underline{y}-\underline{y}_m)]_2$ is the error output along $C\underline{b}_2$

One first series of tests was made to determine the order of magnitude of the outputs due to the unmodeled effects and nonlinearities.

Figure 5.1 shows the outputs when the simulation is run with no

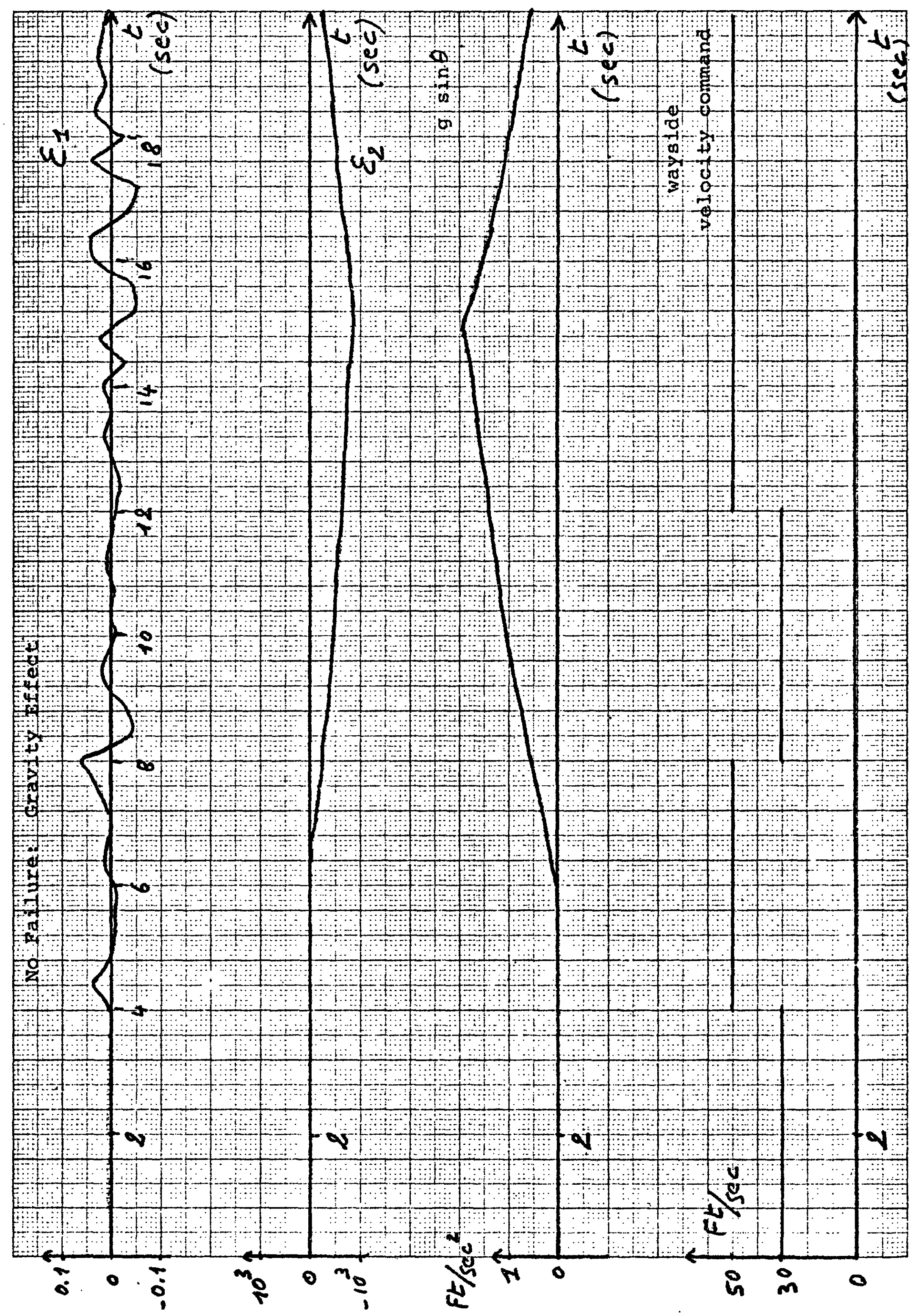
Fig. 5.1


No Failure—Coulomb Friction Effect (Coulomb Friction Modelized in Filter)
0.1
0
-0.1
ε_1
2
4
6
8
10
12
14
16
18
t (sec)
0.10
0
-0.10
ε_2
t (sec)
wayside velocity command
Ft/sec
50.
30.
0
t (sec)

grade effect, and with all the nonlinearities equal to zero, except for the coulomb friction which is modeled in the reference model. As the coulomb friction component is the same in the simulation and the reference model, the error outputs should be identically equal to zero. What appears is then just the results of numerical precision loss. It is the numerical difference of the outputs of two physically equivalent systems modeled in different ways, integrated with a finite difference method, with a time interval of 0.02s. It appears that an output along ε_1 of less than 0.1 in absolute value is not significant, and that an output along ε_2 of less than 0.15 in absolute value is not significant. As one would expect, these values are smaller than the output errors due to the nonlinearities and due to the failures.

Figure 5.2 shows the error outputs when the simulation is run with the grade effect only (no noise, no aerodynamic forces, no Coulomb friction). It appears that the output along ε_1 is not significant. There is only an output along ε_2, of magnitude $1 \cdot 10^3$ at its maximum value along the test track. The third plot shows the value of g $\sin\theta$, the projection of gravity along the track. The assigned wayside velocity command changes 3 times during the test period without any particular effect. It makes physical sense that the unmodeled grade effect results in error output along the channel associated with a failure in the propulsion system: the grade effect is equivalent to a change of response of the propulsion system. Figure 5.3 shows the error outputs when the simulation

Fig. 5.2


No Failure: Gravity Effect
ε1
ε2
g sin θ
wayside
velocity command
t (sec)
Ft/sec²
Ft/sec
0.1
0
-0.1
10³
-10³
1
50
30
2
4
6
8
10
12
14
16
18

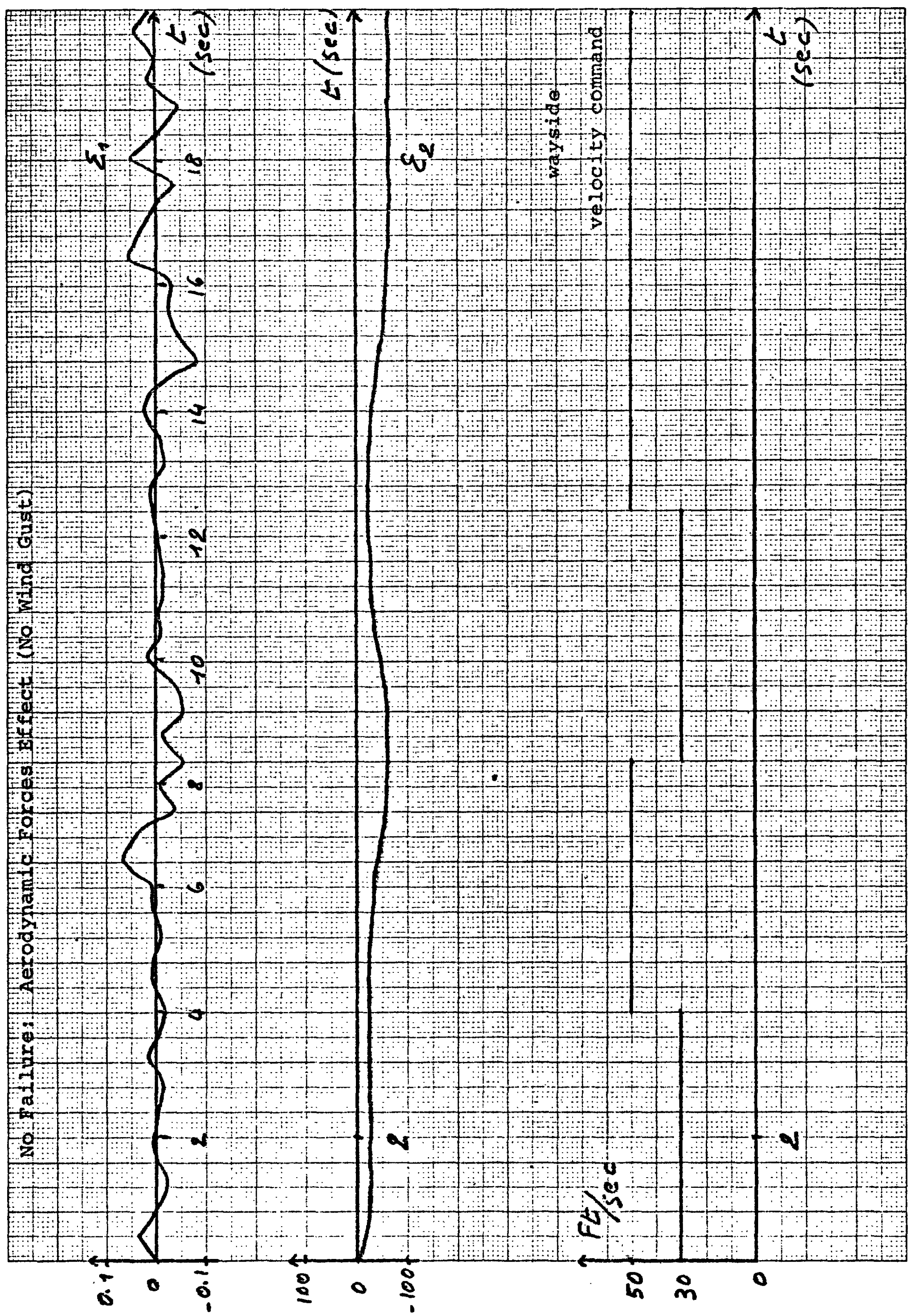

No Failure: Aerodynamic Forces Effect (No Wind Gust)
ε1
ε2
t (sec)
wayside velocity command
Ft/sec
0.1
0
-0.1
100
0
-100
50
30
0
2
4
6
8
10
12
14
16
18

Fig. 5.3

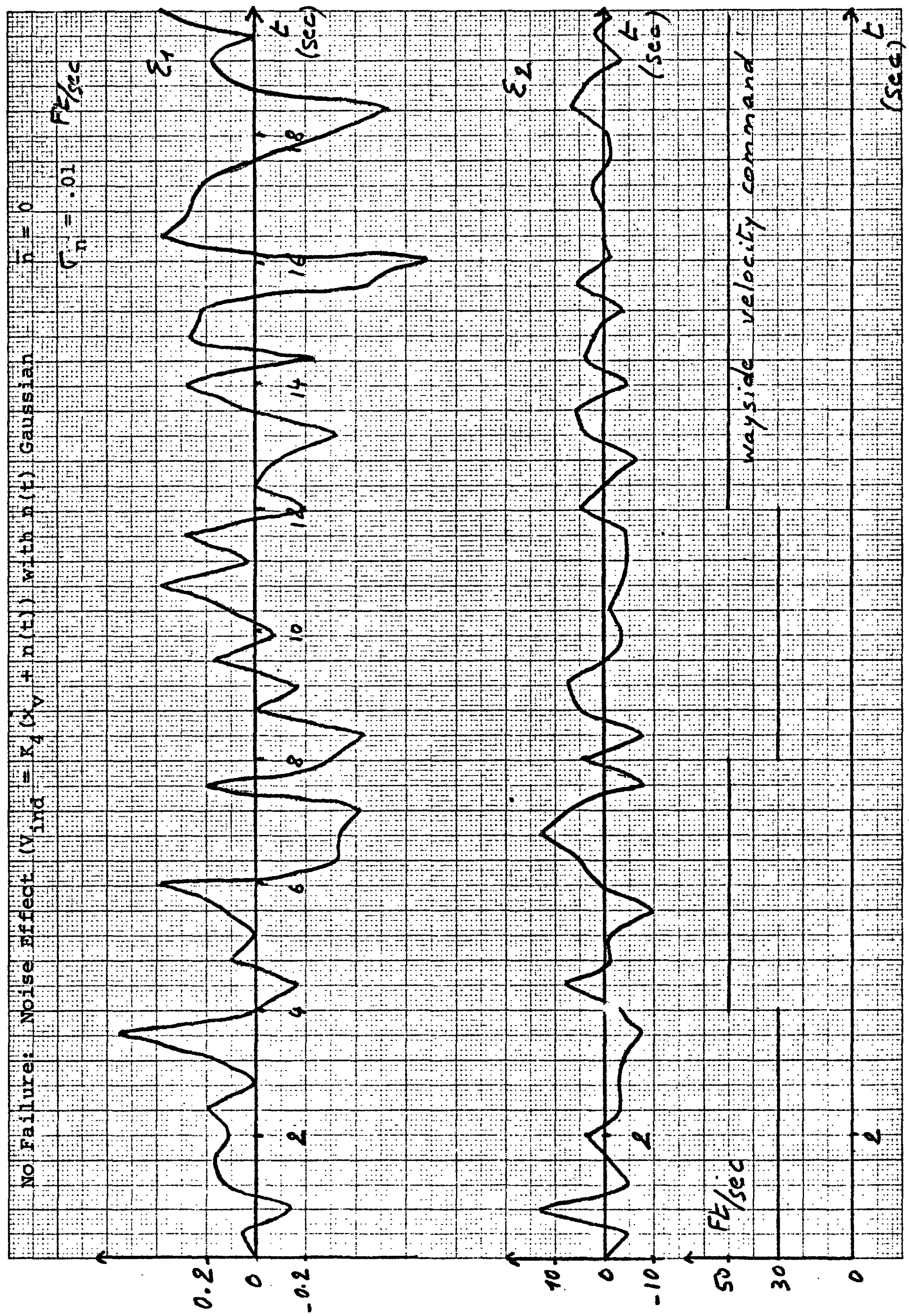
is run with the aerodynamic force effect only, with no wind gust (no noise, no Coulomb friction, no grade effect). It appears that the output along $\mathcal{E}_1$ is not significant; there is an output along $\mathcal{E}_2$ of magnitude less than 100 at its maximum value during the test. The assigned wayside velocity command changes 3 times during the test, resulting in changes of $\mathcal{E}_2$. At steady state, the error along $\mathcal{E}_2$ associated with a velocity of 30 ft/sec is equal to -23, the error associated with a velocity of 50 ft/sec is equal to -63. It makes physical sense that the aerodynamic force effect results only in an error output along the channel associated with a failure in the propulsion system; the aerodynamic force is equivalent to a change of response in the propulsion system.

Figure 5.4 shows the error outputs when the simulation is run with noise in the tachometer output only (no Coulomb friction; no grade component, no aerodynamic force effect). It appears that with a gaussian zero mean noise of standard deviation 0.01 ft/sec in the tachometer, there is an output along $\mathcal{E}_1$ whose maximum value happened to be .68, and an output along $\mathcal{E}_2$, whose maximum value happened to be 13. The assigned wayside velocity command changes 3 times during the test without any particular effect.

In summary, the main neglected effect in the reference model is the grade effect, which produces an output along $\mathcal{E}_2$ only, whose maximum value is roughly 6.10^3 times greater than the residue due to numerical precision loss. The only neglected effect in the reference model which produces an output along $\mathcal{E}_1$ is the noise in the tachom-

Fig. 5.4

No Failure: Noise Effect ($V_{ind} = K_4(x_v + n(t))$ with n(t) Gaussian $\bar{n} = 0$, $\sigma_n = .01$



eter, whose maximum value is roughly 7 times greater than the residue due to numerical precision loss.

A test was then conducted to check the transient response of the outputs due to incorrect initial conditions when there is no failure, no nonlinearity or neglected effect, i.e., when the reference model and the vehicle simulation are physically equivalent. The error on $\mathcal{E}_1$ was initialized to 3000; the error on $\mathcal{E}_2$ was initialized to 622. No change in wayside velocity command was issued. Figure 5.5 shows the error decay as a function of time. As the eigenvalues assigned with $\bar{R}_1$ are -10 and -10, and the eigenvalues assigned with $\bar{R}_2$ are -10, -10, -10, -10, the initial errors should decay in several tenths of a second. As can be seen from Fig. 5.5, $\mathcal{E}_1$ decays to 5 per cent of its initial value within 0.2 sec and $\mathcal{E}_2$ does within 0.7 sec. It appears that even if $\underline{x}_m$ is not initialized in the reference model with the initial values of $\underline{x}$, i.e., $\underline{x}_m(0) \neq \underline{x}(0)$, after 1 sec the error outputs will be less than 1/300 of their initial values.

A third series of tests was finally conducted to investigate the effects of failures in K_1, K_2, K_3, K_4. Figure 5.6 shows the specifications of this test:

- a wind gust, between t = 6 and t = 14 s, headwind of 30 ft/sec
- a variation of the grade. The second plot of Fig. 6 shows the value of $g \sin\theta$, the component of gravity along the track
- 3 changes in wayside velocity command. We have

t =	0 - 4	4 - 8	8 - 12	12 - 20	(sec)
wayside velocity command	30	50	30	50	(ft/sec)

Fig. 5.5

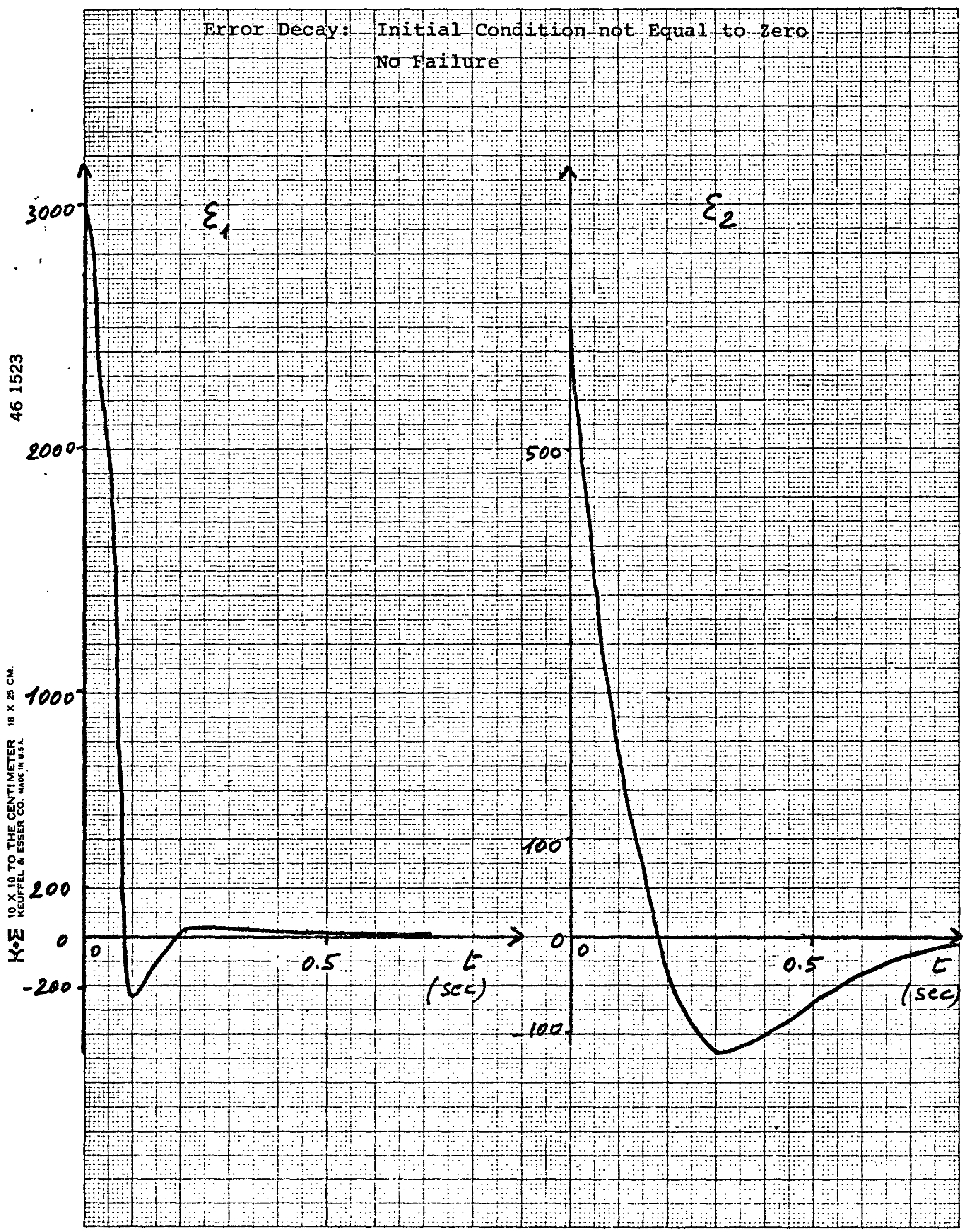
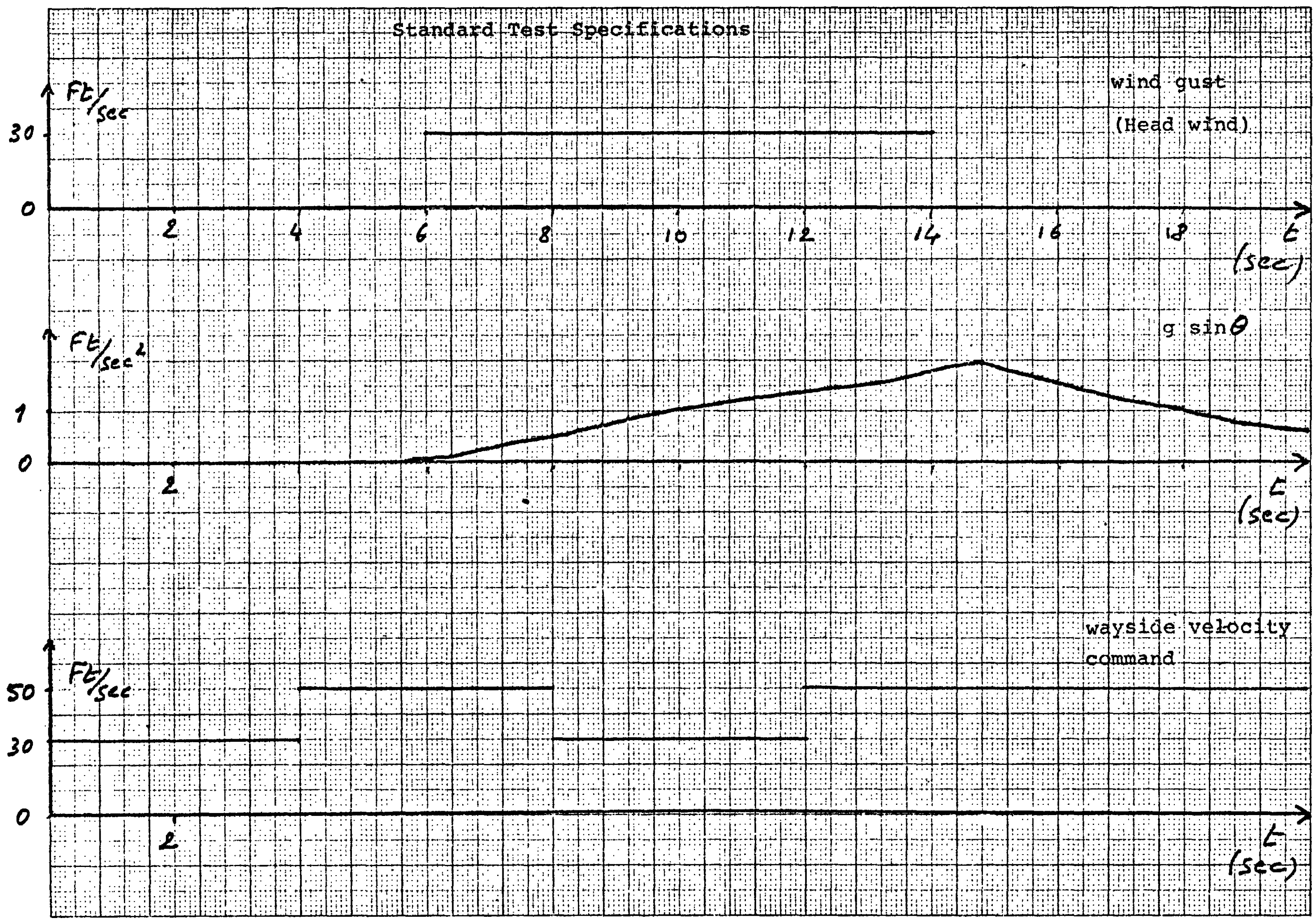

Error Decay: Initial Condition not Equal to Zero
No Failure
3000
2000
1000
200
0
-200
0
0.5
t
(sec)
ε1
500
100
0
-100
0
0.5
t
(sec)
ε2
46 1523
K+E 10 X 10 TO THE CENTIMETER 18 X 25 CM.
KEUFFEL & ESSER CO. MADE IN U.S.A.

Fig. 5.6


Standard Test Specifications
wind gust
(Head wind)
Ft/sec
30
0
2
4
6
8
10
12
14
16
18
t
(sec)
g sinθ
Ft/sec²
1
0
2
t
(sec)
wayside velocity
command
Ft/sec
50
30
0
2
t
(sec)

- tachometer noise was input into the simulation
- Coulomb friction was taken into account in both the simulation and the reference model.

In this series of tests, the worst case was considered with all the nonlinearities and neglected effects running up.

Figure 5.7 shows the outputs along both channels in the absence of failure

Figure 5.8 shows the outputs in case of a failure in K_1

Figure 5.9 shows the outputs in case of a failure in K_2

Figure 5.10 shows the outputs in case of a failure in K_3

Figure 5.11 shows the outputs in case of a failure in K_4

A "failure" in each case means that the gain changed, the new gain being equal to half of its initial value. This is a completely arbitrary failure mode to simulate. It should be recalled that the detection filter does not depend on the manner in which components fail; the filter simply recognizes that the modeled function is no longer being performed. If a total failure were simulated, in the sense that the gain was set to zero rather than half its nominal value, the detection filter outputs signifying failure would be even larger.

It appears (1) that outputs generated by failures can easily be distinguished from outputs due to neglected effects or nonlinearities: failure outputs along $\mathcal{E}_1$ are roughly 10 times larger than residual outputs along $\mathcal{E}_1$ without failures; failure outputs along $\mathcal{E}_2$ are roughly 5 times larger than residual outputs along $\mathcal{E}_2$ without failures;

(2) that simulation shows that the detection filter performs accordingly to theory. Failures in K_1 and K_2 generate unidirectional outputs along $\mathcal{E}_1$, failures in K_3 along $\mathcal{E}_2$, failures in K_4 along $\mathcal{E}_1$ and $\mathcal{E}_2$.

Fig. 5.7

Standard Test—No Failure—Noise and Gravity and Coulomb Friction and Aerodynamic Forces Effect

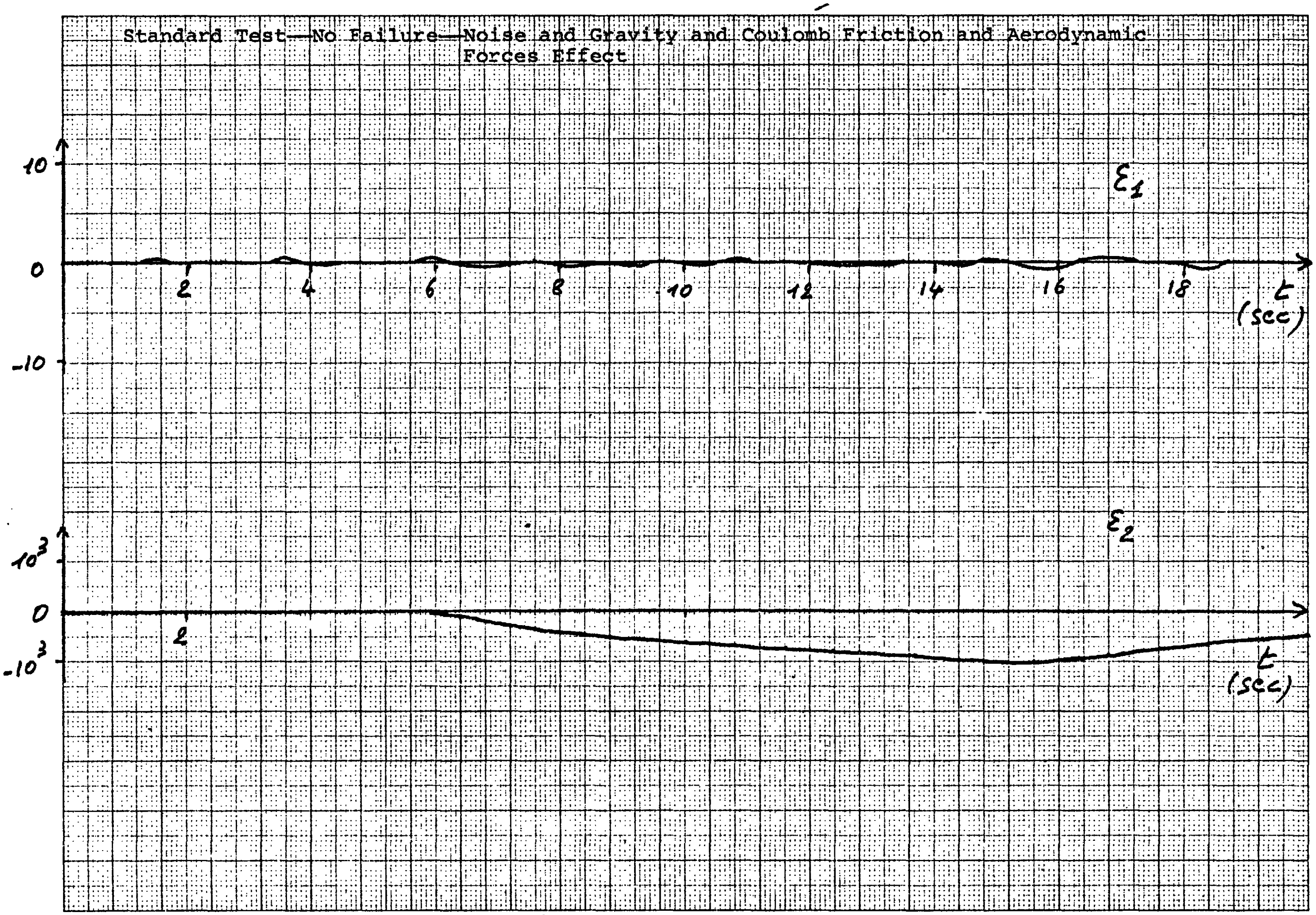


Fig. 5.8

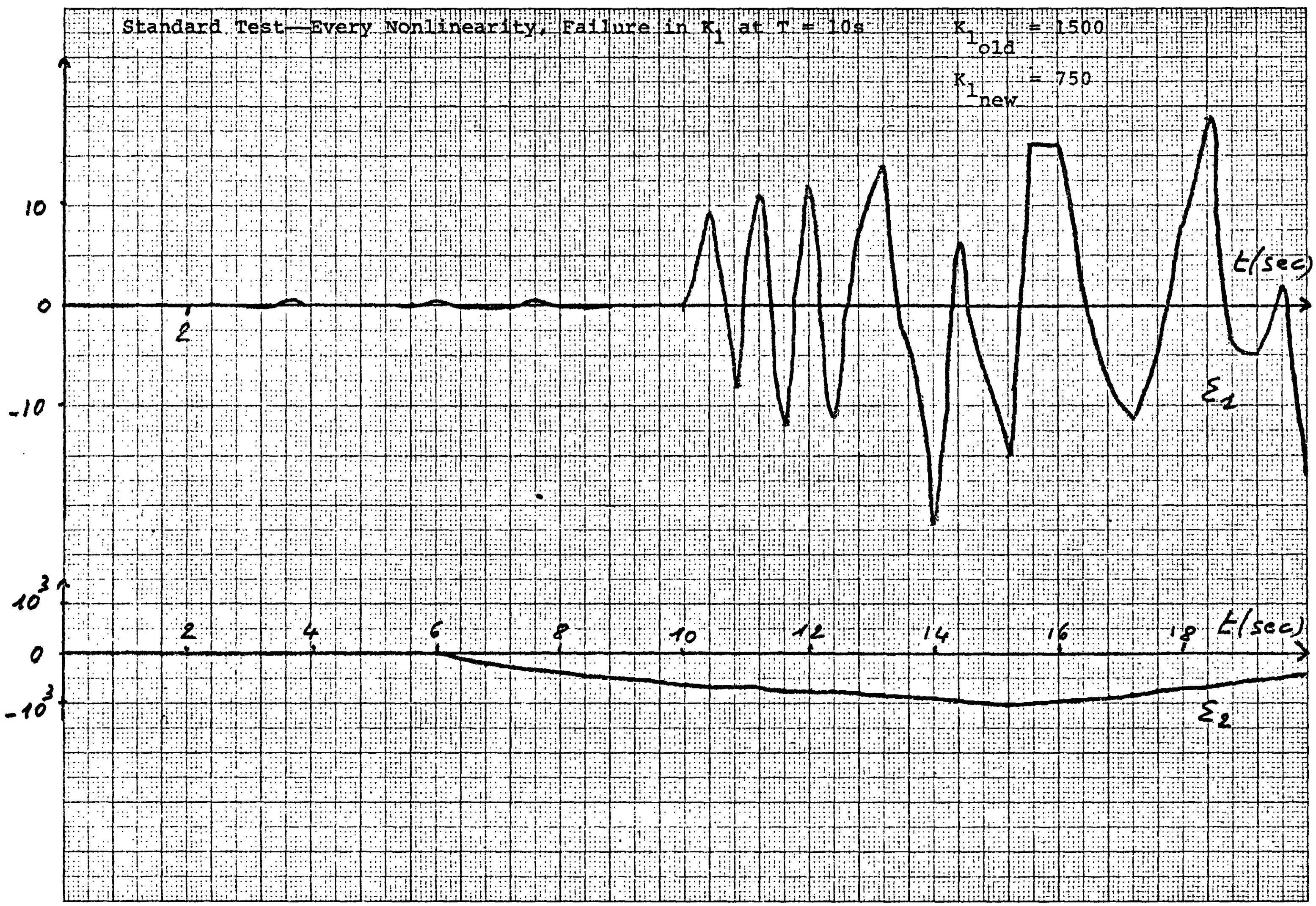

Standard Test—Every Nonlinearity, Failure in K_1 at T = 10s
$K_{1_{old}}$ = 1500
$K_{1_{new}}$ = 750
10
0
-10
t(sec)
ε_1
10^3
0
-10^3
2
4
6
8
10
12
14
16
18
t(sec)
ε_2

Fig. 5.9

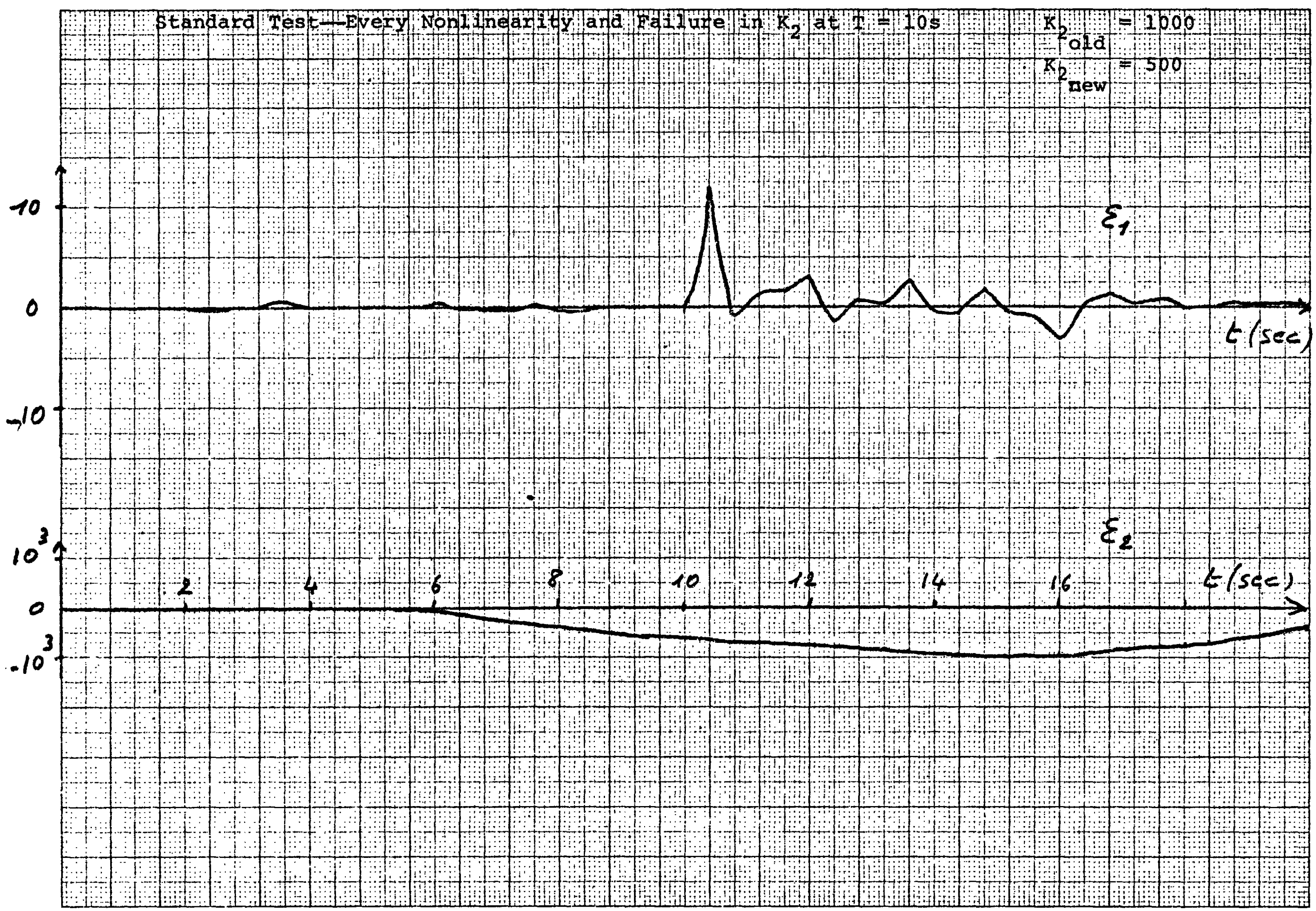

Standard Test—Every Nonlinearity and Failure in K_2 at T = 10s
$K_{2_{old}}$ = 1000
$K_{2_{new}}$ = 500
ε_1
ε_2
10
0
-10
10^3
0
-10^3
2
4
6
8
10
12
14
16
t (sec)
t (sec)

Fig. 5.10

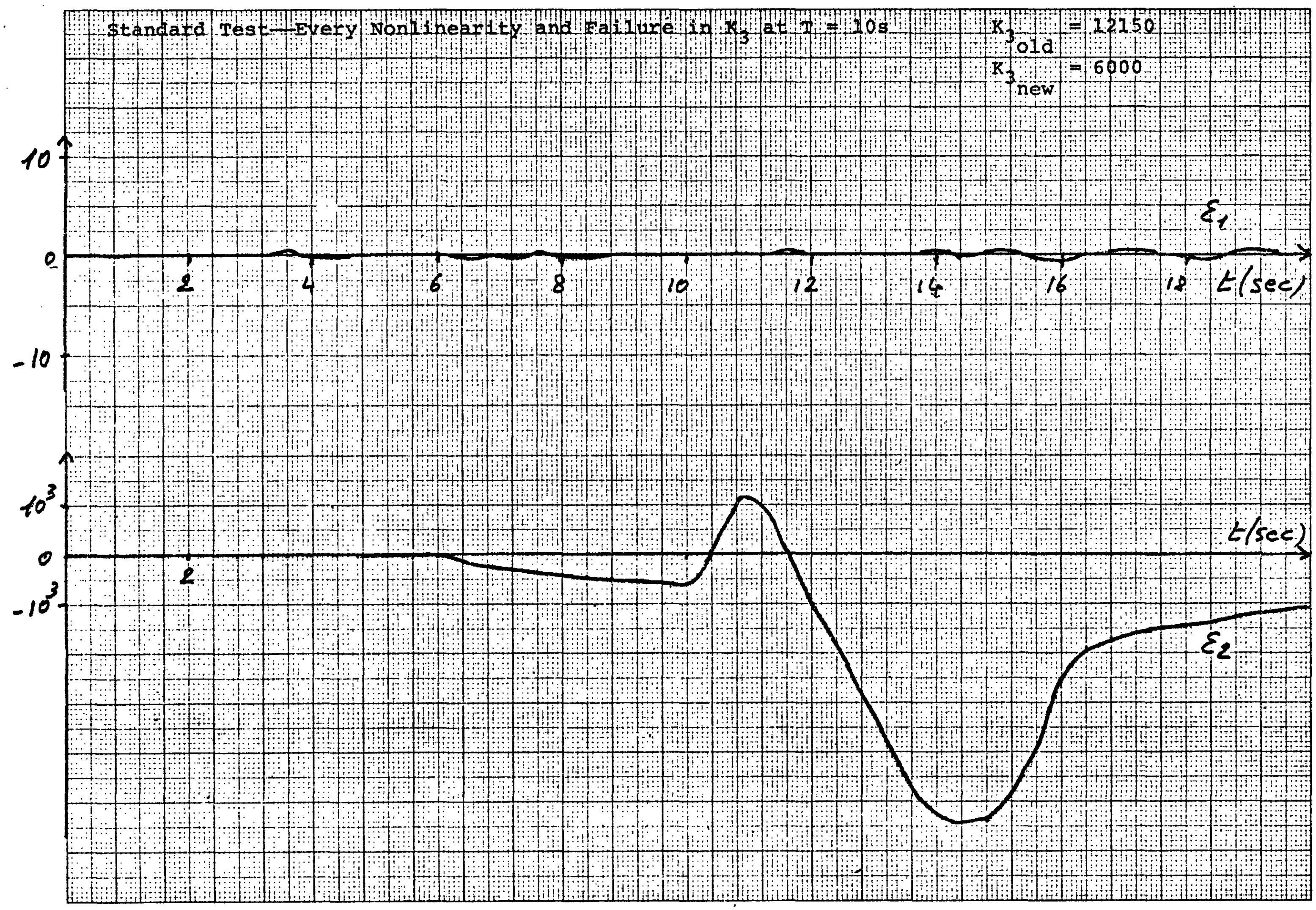
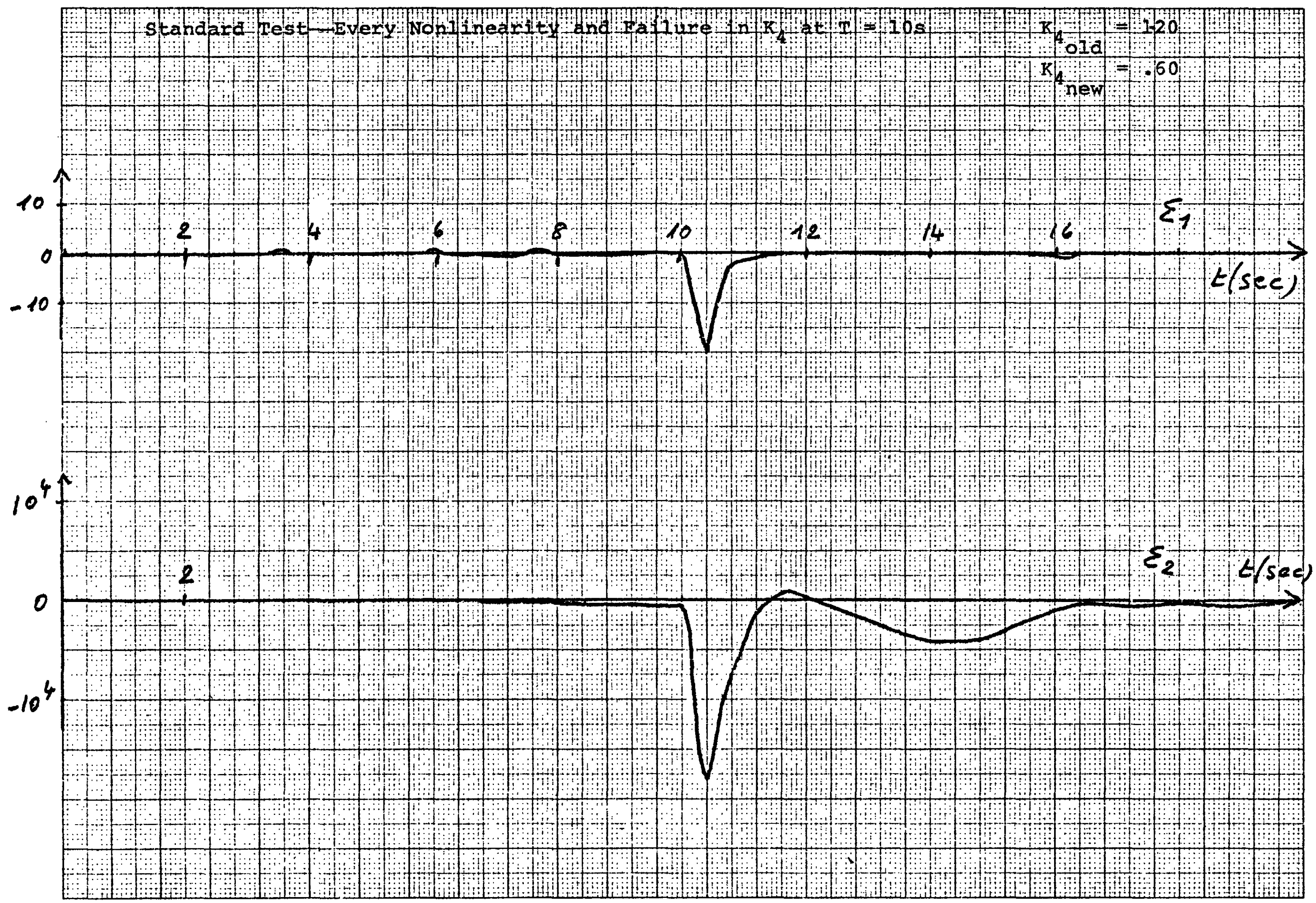

Standard Test—Every Nonlinearity and Failure in K3 at T = 10s
K3old = 12150
K3new = 6000
10
0
-10
10^3
0
-10^3
2
4
6
8
10
12
14
16
18
t (sec)
t (sec)
ε1
ε2

Fig. 5.11


Standard Test—Every Nonlinearity and Failure in K4 at T = 10s
K4old = 1.20
K4new = .60
10
0
-10
2
4
6
8
10
12
14
16
ε1
t(sec)
10^4
0
-10^4
2
ε2
t(sec)

CHAPTER 6

CONCLUSIONS

This study has shown the applicability of detection filter theory to the detection of failures in longitudinal control systems for guideway vehicles. Even though the detection filter theory was developed in the context of a linear time invariant system (prior to failures), the first tests have shown that error outputs due to nonlinearities, or noise, or neglected effects in the reference model can be easily distinguished from error outputs due to component failures. It was further shown that it is easy to distinguish between the three kinds of failures most likely to occur in the velocity control loop. This feature allows to achieve high levels of reliability with less hardware redundancy. To detect which element is failing, it is not needed to triplicate every component susceptible of failure, as it would be if majority rule were the detection law.

This study, however, has not addressed the problem of the detection law. It was just shown that for a particular choice of filter eigenvalues, the problem can be solved, but no attempt was made to determine an optimal choice of eiganvalues with respect to the differentiation between normal error outputs and failure outputs, and with respect to the time delay before a failure can be detected. This would be the object of a further study.

REFERENCES

1. Beard, Richard V., "Failure Accommodation in Linear Systems through Self-Reorganization," PhD Thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, February 1971; also Man-Vehicle Laboratory Report MVT-71-1.
2. Jones, Harold L., "Failure Detection in Linear Systems," PhD Thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, August 1973; also C.S. Draper Laboratory Report T-608.

APPENDIX A

LISTING OF DETECTION FILTER DESIGN PROGRAM

```
COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12)
DIMENSION CBI(12,12),BUFF(12,12)
INTEGER P,Q,R
CALL MAIN1
IECR=0
CALL MAIN2
IECR=0
CALL MAIN3
IECR=0
CALL MAIN4
IECR=0
CALL MAIN5
IECR=3
CALL MAIN6
IECR=0
CALL MAIN7
IECR=0
CALL MAIN8
IECR=6
CALL MAIN9
CALL MAIN9C
CALL MAIN9D
STOP
END
```

```
      SUBROUTINE MAIN1
       COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
       COMMON/MANI3/IROW,ICOL
       COMMON/MANI4/OMEGC,DS,CS
       DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12)
      DIMENSION BUFF(12,12)
       DIMENSION IROW(12),ICOL(12)
       DIMENSION OMEGC(12,12),DS(12,12),CS(12,12)
       INTEGER P,Q,R
1      WRITE(6,100)
100    FORMAT(1X,'WRITE N,P,Q,R,IECR,IC,WHERE A(N,N),B(N,Q),C(P,N),'
     1,'BI(N,R),IECR WRITING INDEX,TYPICAL IECR IS 0,',/,'IC DIMENSION'
     2,'OF DECLARED COLUMN OF C,6I3')
       READ(5,1000)N,P,Q,R,IECR,IC
1000    FORMAT(6I3)
       WRITE(6,1000)N,P,Q,R,IECR,IC
       WRITE(6,101)
101    FORMAT(1X,'IF DATA ARE CORRECTLY ENTERED,TYPE11,ELSE,TYPE00')
       READ(5,1001)ISIGN1
1001   FORMAT(I2)
       IF(IECR.GT.5)WRITE(6,1001)ISIGN1
       IF(ISIGN1.EQ.0)GO TO 1
2      WRITE(6,102)
102    FORMAT(1X,'WRITE EPS,PRECISION FOR THE COMPUTATION OF THE '
     1,'RANK OF C,TYPICAL EPS IS 0.0001')
       READ(5,1002)EPS
1002   FORMAT(F10.4)
       WRITE(6,1002)EPS
       WRITE(6,101)
       READ(5,1001)ISIGN2
       IF(IECR.GT.5)WRITE(5,1001)ISIGN2
       IF(ISIGN2.EQ.0) GO TO 2
       CALL LEOFA(A,N,IECR)
       CALL LEOFB(B,N,Q,IECR)
       CALL LEOFC(C,N,P,IECR)
       DO 3 I=1,P
       DO 3 J=1,N
3      BUFF(I,J)=C(I,J)
       CALL MFGR(C,IC,P,N,IRANK,IROW,ICOL,EPS,IER)
       WRITE(6,104) IER
104    FORMAT(1X,'ERROR CODE IN COMPUTATION OF RANK OF C IS IER'
     1,' ',I6)
       DO 4 I=1,P
       DO 4 J=1,N
4      C(I,J)=BUFF(I,J)
       WRITE(6,105)IRANK
105    FORMAT(1X,'RANK OF C IS',I4)
       RETURN
       END
       SUBROUTINE PROCBI(C,BI,CBI,N,P,R,IECR)
       DIMENSION C(12,12),BI(12,12),CBI(12,12)
       INTEGER P,Q,R
       DO 1 I=1,P
       DO 1 J=1,R
1     CBI(I,J)=0.E0
       DO 2 I=1,P
       DO 2 J=1,R
       DO 2 K=1,N
2     CBI(I,J)=CBI(I,J)+C(I,K)*BI(K,J)
       IF(IECR.GT.3) WRITE(6,100)((CBI(I,J),J=1,R),I=1,P)
100    FORMAT(5E10.4)
```

```
      WRITE(6,101)                                                      MAI00620
  101 FORMAT(1X,'THE PRODUCT OF C AND BI HAS BEEN COMPUTED AND'         MAI00630
     1,'PUT IN MATRIX CBI(P,R)')                                        MAI00640
      RETURN                                                            MAI00650
      END                                                               MAI00660
```

```
      SUBROUTINE LEOFA(A,N,IECR)
      DIMENSION A(12,12)
      INTEGER P,Q,R
1     WRITE(6,100)
100   FORMAT(1X,'TYPE ((A(I,J),J=1,N),I=1,N,FREE FORMAT')
      READ(5,*)((A(I,J),J=1,N),I=1,N)
      WRITE(6,101)
101   FORMAT(1X,'A=')
      WRITE(6,*)((A(I,J),J=1,N),I=1,N)
      WRITE(6,102)
102   FORMAT(1X,'IF SATISFIED,TYPE 11.OTHERWISE,TYPE 00')
      READ(5,103)ISIGN
103   FORMAT(I2)
      IF(ISIGN.EQ.0) GO TO 1
      RETURN
      END
      SUBROUTINE LEOFB(B,N,Q,IECR)
      DIMENSION B(12,12)
      INTEGER P,Q,R
      IQ=Q
1     WRITE(6,100)
100   FORMAT(1X,'TYPE B(I,J),J=1,Q,I=1,N,FREE FORMAT')
      READ(5,*)((B(I,J),J=1,IQ),I=1,N)
      WRITE(6,101)
101   FORMAT(1X,'B=')
      WRITE(6,*)((B(I,J),J=1,IQ),I=1,N)
      WRITE(6,102)
102   FORMAT(1X,'IF SATISFIED,TYPE 11.OTHERWISE,TYPE 00')
      READ(5,103)ISIGN
103   FORMAT(I2)
      IF(ISIGN.EQ.0) GO TO 1
      RETURN
      END
      SUBROUTINE LEOFC(C,N,P,IECR)
      DIMENSION C(12,12)
      INTEGER P,Q,R
      IP=P
1     WRITE(6,100)
100   FORMAT(1X,'TYPE((C(I,J),J=1,N),I=1,P),FREE FORMAT')
      READ(5,*)((C(I,J),J=1,N),I=1,IP)
      WRITE(6,101)
101   FORMAT(1X,'C=')
      WRITE(6,*)((C(I,J),J=1,N),I=1,IP)
      WRITE(6,102)
102   FORMAT(1X,'IF SATISFIED,TYPE 11.OTHERWISE,TYPE 00')
      READ(5,103)ISIGN
103   FORMAT(I2)
      IF(ISIGN.EQ.0) GO TO 1
      RETURN
      END
      SUBROUTINE LEOFBI(BI,N,R,IECR)
      DIMENSION BI(12,12)
      INTEGER P,Q,R
      IR=R
1     WRITE(6,100)
100   FORMAT(1X,'TYPE((BI(I,J),J=1,R),I=1,N),FREE FORMAT')
      READ(5,*)((BI(I,J),J=1,IR),I=1,N)
      WRITE(6,101)
101   FORMAT(1X,'BI=')
      WRITE(6,*)((BI(I,J),J=1,IR),I=1,N)
      WRITE(6,102)
```

```
102  FORMAT(1X,'IF SATISFIED,TYPE 11.OTHERWISE,TYPE 00')            BUF00620
     READ(5,103)ISIGN                                               BUF00630
103  FORMAT(I2)                                                     BUF00640
     IF(ISIGN.EQ.0) GO TO 1                                         BUF00650
     RETURN                                                         BUF00660
     END                                                            BUF00670
```

```
      SUBROUTINE MAIN2
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
      COMMON/MANI3/IROW,ICOL
      DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12),IROW(12)
      DIMENSION ICOL(12),BUFF(12,12)
      INTEGER P,Q,R
      WRITE(6,100)
100   FORMAT(1X,'IF THE NUMBER OF EVENTS IS GREATER THAN RANK OF C.'/
     1'YOU HAVE TO DIVIDE THE BI IN SETS OF NO MORE THAN RANKC VECTORS'
     1,/,'IN A GROUP.IF THIS IS THE CASE ,TYPE 00,ELSE TYPE 11')
      READ(5,1000) ISIGN1
1000  FORMAT(I2)
      IF(IECR.GT.5) WRITE(6,1000) ISIGN1
      IF(ISIGN1.NE.0)GO TO 1
      STOP
1     WRITE(6,101)
101   FORMAT(1X,'TYPE THE NEW VALUE OF R,I3')
      READ(5,1001) R
1001  FORMAT(I3)
      WRITE(6,1001)R
      WRITE(6,102)
102   FORMAT(1X,'IF THE DATA ARE CORRETLY ENTERED,TYPE 11,ELSE TYPE 00'
     1)
      READ(5,1000) ISIGN2
      IF(IECR.GT.5) WRITE(6,1000) ISIGN2
      IF(ISIGN2.EQ.0) GO TO 1
      IF(IECR.GT.5)WRITE(6,1001)N
      CALL LEOFBI(BI,N,R,IECR)
      CALL PROCBI(C,BI,CBI,N,P,R,IECR)
      ICBI=12
      DO 2 I=1,P
      DO 2 J=1,R
2     BUFF(I,J)=0.E0
      DO 3 I=1,P
      DO 3 J=1,R
3     BUFF(I,J)=CBI(I,J)
      CALL MFGR(BUFF,ICBI,P,R,IRANK,IROW,ICOL,EPS,IER)
      WRITE(6,103) IER,IRANK
103   FORMAT(1X,'ERROR CODE IN THE COMPUTATION OF CBI RANK IS',I3,/,
     1'RANK OF CBI IS',I4)
      WRITE(6,104)(IROW(I),I=1,N),(ICOL(I),I=1,N)
104   FORMAT(1X,'IROW=',I3,'ICOL=',I3,'I=',I3)
      WRITE(6,105)
105   FORMAT(1X,'IF THE RANK OF CBI MATRIX IS NOT EQUAL TO THE ',/
     1,'NUMBER OF EVENTS BI,YOU HAVE TO CHANGE THE GROUP OF BI,TO DO',/
     3,'THIS,ENTER 00.ON THE CONTRARY,IF THE CBI ARE LINEARLY INDEPEN-'
     4,/,'DENT,ENTER 11')
      READ(5,1002) ISIGN3
1002  FORMAT(I2)
      IF(IECR.GT.5) WRITE(6,1002) ISIGN3
      IF(ISIGN3.EQ.0) GO TO 1
      WRITE(6,1003)
1003  FORMAT(1X,'A DETECTION FILTER WILL BE DESIGNED FOR THE EVENTS',/
     1,'BI YOU HAVE KEYED IN AT THIS POINT')
      RETURN
      END
```

```
      SUBROUTINE MAIN3
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
       COMMON/MANI4/OMEGC,DS,CS
      COMMON/MANI6/BUFF1
      COMMON/TRASH2/CBVEC,CBVIV
      DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12)
      DIMENSION BUFF(12,12),BUFF1(12,12)
      DIMENSION OMEGC(12,12),CBVEC(78),CBVIV(78),DS(12,12),CS(12,12)
      INTEGER P,Q,R
      IF(IECR.GT.2)WRITE(6,1000)
 1000 FORMAT(1X,'SUBROUTINE MAIN3')
C**COMPUTATION OF OMEGC
      EPSI1= 1.E-9
      DO 11 I=1,N
      DO 12 J=1,N
 12   OMEGC(I,J)=0. E0
 11   OMEGC(I,I)=1.E0
       IC=12
       CALL ORTRED(C,OMEGC,P,N,IC,EPSI1,IECR)
      IF(IECR.GT.3)WRITE(6,100)((OMEGC(I,J),J=1,N),I=1,N)
 100  FORMAT(4(1X,'OMEGC=',E10.4))
C**COMPUTATION OF CBIT*CBI
      DO 1 I=1,R
      DO 1 J=1,R
 1    BUFF(I,J)=0. E0
      DO 2 I=1,R
      DO 2 J=1,R
      DO 2 K=1,P
 2    BUFF(I,J)=BUFF(I,J)+CBI(K,I)*CBI(K,J)
      IF(IECR.GT.5)WRITE(6,200)((BUFF(I,J),J=1,R),I=1,R)
 200  FORMAT(4(1X,'CBIT*CBI=',E10.4))
C**TRANSITION TO SYMMETRIC STORAGE
      IB=12
      CALL VCVTFS(BUFF,R,IB,CBVEC)
      IF(IECR.GT.5)WRITE(6,300)(CBVEC(I),I=1,10)
 300  FORMAT(5(1X,'CBVEC=',F10.4))
C**COMPUTATION OF INVERSE OF CBIT*CBI
      CALL LINV1P(CBVEC,R,CBVIV,IDGT,D1,D2,IER)
      IF(IECR.GT.5)WRITE(6,400)(CBVIV(I),I=1,10)
 400  FORMAT(4(1X,'CBVIV=',F10.4))
C** TRANSITION TO FULL MODE (CBIT*CBI)-1=BUFF
      CALL VCVTSF(CBVIV,R,BUFF,IB)
      IF(IECR.GT.5)WRITE(6,500)((BUFF(I,J),J=1,R),I=1,R)
 500  FORMAT(4(1X,'C-1=',E10.4))
C** COMPUTATION OF BI*(CBIT*CBI)-1*CBIT=BUFF
      DO 3 I=1,R
      DO 3 J=1,P
 3    BUFF1(I,J)=0. E0
      DO 4 I=1,R
      DO 4 J=1,P
      DO 4 K=1,R
 4    BUFF1(I,J)=BUFF(I,K)*CBI(J,K)+BUFF1(I,J)
C******'*****************
C** BUFF1=(CBIT*CBI)-1*CBIT IN CONMON MANI6
C******'*****************
      IF(IECR.GT.5)WRITE(6,600)((BUFF1(I,J),J=1,P),I=1,R)
 600  FORMAT(4(1X,'BUFF1=',F10.4))
      DO 5 I=1,N
      DO 5 J=1,P
 5    BUFF(I,J)=0. E0
      DO 6 I=1,N
```

```
      DO 6 J=1,P
      DO 6 K=1,R
 6    BUFF(I,J)=BI(I,K)*BUFF1(K,J)+BUFF(I,J)
      IF(IECR.GT.5)WRITE(6,700)((BUFF(I,J),J=1,P),I=1,N)
 700  FORMAT(4(1X,'BCBTCB-1CBT=',E10.4))
C** COMPUTATION OF DS
      DO 7 I=1,N
      DO 7 J=1,P
 7    DS(I,J)=0. E0
      DO 8 I=1,N
      DO 8 J=1,P
      DO 8 K=1,N
 8    DS(I,J)=DS(I,J)+A(I,K)*BUFF(K,J)
      IF(IECR.GT.4)WRITE(6,900)((DS(I,J),J=1,P),I=1,N)
 900  FORMAT(4(1X,'DS=',E10.4))
C ** COMPUTATION OF CS
      DO 9 I=1,P
      DO 9 J=1,P
 9    CS(I,J)=0. E0
      DO 10 I=1,P
      DO 10 J=1,P
      DO 10 K=1,N
 10    CS(I,J)=CS(I,J)+C(I,K)*BUFF(K,J)
      IF(IECR.GT.5)WRITE(6,800)((CS(K,J),J=1,P),K=1,P)
 800  FORMAT(4(1X,'CS=',E10.4))
      RETURN
      END
```

```
      SUBROUTINE MAIN4
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
       COMMON/MANI3/IROW,ICOL
       COMMON/MANI4/OMEGC,DS,CS
      COMMON/MANI5/INU,INUS,INJO
      COMMON/MANI6A/ADSC
      COMMON/TRASH2/XMD,BUFF1,BUFF2,BUFF3
      DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12)
      DIMENSION BUFF(12,12)
      DIMENSION BUFF1(12,12),XMD(144,12),CS(12,12),DS(12,12)
      DIMENSION BUFF2(12,12)
      DIMENSION BUFF3(12,12),IROW(12),ICOL(12),OMEGC(12,12)
      DIMENSION INU(12),ADSC(12,12)
      INTEGER P,Q,R
      IF(IECR.GT.2)WRITE(6,1100)
 1100 FORMAT(1X,'SUBROUTINE MAIN4')
C**COMPUTATION OF CS(CONTINUED)
      DO 1 I=1,P
      DO 1 J=1,P
    1 BUFF1(I,J)=-CS(I,J)
      DO 2 I=1,P
    2 BUFF1(I,I)=1.E0+BUFF1(I,I)
      IF(IECR.GT.6)WRITE(6,100)((BUFF1(I,J),J=1,P),I=1,P)
  100 FORMAT(4(1X,'BUFF1=',E10.4))
      DO 3 I=1,P
      DO 3 J=1,N
      CS(I,J)=0.E0
      DO 3 K=1,P
    3 CS(I,J)=CS(I,J)+BUFF1(I,K)*C(K,J)
      EPSIL=1.E-4
      DO 17 I=1,P
      DO 17 J=1,N
      IF(ABS(CS(I,J)).LT.EPSIL)CS(I,J)=0.
   17 CONTINUE
      IF(IECR.GT.5)WRITE(6,200)((CS(I,J),J=1,N),I=1,P)
  200 FORMAT(4(1X,'CS=',E10.4))
C**COMPUTATION OF A-DS*C
      DO 4 I=1,N
      DO 4 J=1,N
      BUFF2(I,J)=0.E0
      DO 4 K=1,P
    4 BUFF2(I,J)=BUFF2(I,J)+DS(I,K)*C(K,J)
      IF(IECR.GT.6)WRITE(6,300)((BUFF2(I,J),J=1,N),I=1,N)
  300 FORMAT(4(1X,'DSC=',E10.4))
      DO 5 I=1,N
      DO 5 J=1,N
    5 BUFF1(I,J)=A(I,J)-BUFF2(I,J)
      DO 55 I=1,N
      DO 55 J=1,N
   55 ADSC(I,J)=BUFF1(I,J)
      IF(IECR.GT.5)WRITE(6,400)((BUFF1(I,J),J=1,N),I=1,N)
  400  FORMAT(4(1X,'A-DS*C=',E10.4))
C**COMPUTATION OF XMD=MD'
      DO 6 I=1,P
      DO 6 J=1,N
    6 XMD(I,J)=CS(I,J)
      DO 7 I=1,P
      DO 7 J=1,N
    7 BUFF3(I,J)=CS(I,J)
      NN=N-1
      IF(NN.EQ.0) GO TO 11
```

```
      DO 11 L=1,NN
      NN1=L*P
      DO 8 I=1,P
      DO 8 J=1,N
      BUFF2(I,J)=0.E0
      DO 8 K=1,N
 8    BUFF2(I,J)=BUFF2(I,J)+BUFF3(I,K)*BUFF1(K,J)
      IF(IECR.GT.7)WRITE(6,500)((BUFF3(I,J),J=1,N),I=1,P)
 500  FORMAT(4(1X,'BUFF3=',E10.4))
      DO 9 I=1,P
      DO 9 J=1,N
 9    BUFF3(I,J)=BUFF2(I,J)
        DO 10 I=1,P
       DO 10 J=1,N
       II=I+L*P
 10    XMD(II,J)=BUFF2(I,J)
      IF(IECR.GT.7)WRITE(6,600)((BUFF2(I,J),J=1,N),I=1,P)
 600  FORMAT(4(1X,'BUFF2=',E10.4))
 11   CONTINUE
      NP=N*P
      IF(IECR.GT.3) WRITE(6,700)((XMD(I,J),J=1,N),I=1,NP)
 700  FORMAT(4(1X,'XMD=',E10.4))
C** ORTHOGONAL REDUCTION OF MD'=XMD
      IXMD=144
      EPSI1=1.E-9
      WRITE(6,2100)
 2100 FORMAT(1X,'TYPE 11 IF YOU WANT TO DESIGN THE FILTER,',/,1X,
     1'00 IF YOU DO NOT')
      READ(5,1101)ISIGN
 1101 FORMAT(I2)
      CALL ORTRED(XMD,OMEGC,NP,N,IXMD,EPSI1,IECR)
      IF(IECR.GT.1)WRITE(6,800)((OMEGC(I,J),J=1,N),I=1,N)
 800  FORMAT(4(1X,'OMEGS=',E10.4))
      IF(ISIGN.EQ.11) GO TO 15
      GO TO 16
 15   CONTINUE
      DO 14 I=1,N
      DO 13 J=1,N
 13   BUFF2(I,J)=0.E0
 14   BUFF2(I,I)=1.E0
      ICRI=4
      CALL ORTRED(XMD,BUFF2,NP,N,IXMD,EPSI1,ICRI)
      IF(IECR.GT.1)WRITE(6,2101)((BUFF2(I,J),J=1,N),I=1,N)
 2101 FORMAT(4(1X,'OMEGSP=',E10.4))
 16   CONTINUE
C**COMPUTATION OF RANKOMEGS AND LINEAR DEPENDENCY
      DO 12 I=1,N
      DO 12 J=1,N
 12   BUFF2(I,J)=OMEGC(I,J)
      IOMEGC=12
      CALL MFGR(BUFF2,IOMEGC,N,N,IRANK,IROW,ICOL,EPS,IER)
      IF(IECR.GT.1)WRITE(6,900)IER
 900  FORMAT(1X,'ERROR CODE IN COMPUTATION OF OMEGS RANK IS',I3)
      WRITE(6,1000)IRANK
 1000 FORMAT(1X,'RANK OF OMEGS IS',I3)
      INUS=IRANK
      WRITE(6,1003)
 1003 FORMAT(1X,'IF YOU WANT TO OVERCOME THE RANK OF OMEGS',/,1X,
     1'GIVEN BY AUTOMATIC COMPUTATION,TYPE 00.OTHERWISE,TYPE 11')
      READ(5,1004)ISIGN
 1004 FORMAT(I2)
```

```
      IF(ISIGN.NE.0) GO TO 18
      WRITE(6,1005)
 1005 FORMAT(1X,'TYPE THE RANK OF OMEGS')
      READ(5,*) INUS
      WRITE(6,*)INUS
 18   CONTINUE
      IF(IECR.GT.0)WRITE(6,1001)(IROW(I),ICOL(I),I=1,N)
 1001 FORMAT(2(1X,'IROW=',I3,'ICOL=',I3))
      IF(IECR.GT.0)WRITE(6,1002)((BUFF2(I,J),J=1,N),I=1,N)
 1002 FORMAT(4(1X,'BUFF2=',E10.4))
      RETURN
      END
```

```
      SUBROUTINE MAIN5
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
      COMMON/MANI3/IROW,ICOL
      COMMON/MANI4/OMEGC,DS,CS
      COMMON/MANI5/INU,INUS,INJO
      COMMON/MANI5A/GG
      COMMON/TRASH2/XMD,XMD2,BUFF1,BUFF2,BUFF3
      DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12)
      DIMENSION BUFF(12,12)
      DIMENSION CS(12,12),DS(12,12),OMEGC(12,12),IROW(12),ICOL(12)
      DIMENSION VECBU(12),BUFF1(12,12),BUFF2(12,12),BUFF3(12,12)
      DIMENSION XMD(144,12),XMD2(144,12)
      DIMENSION INU(12),GG(12,12)
C**COMPUTATION OF M
      CALL MAIN5P(XMD2)
      INTEGER P,Q,R
C**COMPUTATION FOR EACH BI,I=1,R
      DO 15 JJ=1,R
C** COMPUTATION OF DBI
      CC=0.
      DO 1 I=1,P
 1    CC=CC+CBI(I,JJ)**2
      CC=1./CC
      IF(IECR.GT.6) WRITE(6,100)CC,JJ
 100  FORMAT(1X,'(CBITCBI)-1=',E10.4,'FOR I =',I3)
      DO 2 I=1,N
      VECBU(I)=0.
      DO 2 K=1,N
 2    VECBU(I)=VECBU(I)+A(I,K)*BI(K,JJ)
      IF(IECR.GT.8)WRITE(6,200)(VECBU(I),I=1,N)
 200  FORMAT(4(1X,'VECBU=',E10.4))
      DO 3 I=1,N
      DO 3 J=1,P
 3    BUFF(I,J)=VECBU(I)*CBI(J,JJ)*CC
      IF(IECR.GT.5)WRITE(6,300)((BUFF(I,J),J=1,P),I=1,N)
 300  FORMAT(4(1X,'DBI=',E10.4))
C**COMPUTATION OF C'
      DO 4 I=1,P
      DO 4 J=1,P
 4    BUFF1(I,J)=-CBI(I,JJ)*CBI(J,JJ)*CC
      DO 5 I=1,P
 5    BUFF1(I,I)=1.+BUFF1(I,I)
      IF(IECR.GT.8)WRITE(6,400)((BUFF1(I,J),J=1,P),I=1,P)
 400  FORMAT(4(1X,'BUFF1= ',E10.4))
      DO 6 I=1,P
      DO 6 J=1,N
      BUFF2(I,J)=0.
      DO 6 K=1,P
 6    BUFF2(I,J)=BUFF2(I,J)+BUFF1(I,K)*C(K,J)
      IF(IECR.GT.5)WRITE(6,500)((BUFF2(I,J),J=1,N),I=1,P)
 500  FORMAT(4(1X,'CPRIME=',E10.4))
C** COMPUTATION OF A-DBI*C
      DO 7 I=1,N
      DO 7 J=1,N
      BUFF3(I,J)=0.
      DO 7 K=1,P
 7    BUFF3(I,J)=BUFF3(I,J)+BUFF(I,K)*C(K,J)
      IF(IECR.GT.8)WRITE(6,600)((BUFF3(I,J),J=1,N),I=1,N)
 600  FORMAT(4(1X,'BUFF3=',E10.4))
      DO 8 I=1,N
      DO 8 J=1,N
```

```
 8    BUFF1(I,J)=A(I,J)-BUFF3(I,J)
      IF(IECR.GT.5)WRITE(6,700)((BUFF1(I,J),J=1,N),I=1,N)
 700  FORMAT(4(1X,'A-DBI*C=',E10.4))
C** COMPUTATION OF XMD=MD'
      DO 9 I=1,P
      DO 9 J=1,N
      XMD(I,J)=BUFF2(I,J)
 9    BUFF3(I,J)=BUFF2(I,J)
      NM=N-1
      DO 13 L=1,NN
      NM1=L*P
      DO 10 I=1,P
      DO 10 J=1,N
      BUFF2(I,J)=0.
      DO 10 K=1,N
 10   BUFF2(I,J)=BUFF2(I,J)+BUFF3(I,K)*BUFF1(K,J)
      IF(IECR.GT.8)WRITE(6,800)((BUFF3(I,J),J=1,N),I=1,P)
 800  FORMAT(4(1X,'BUFF3=',E10.4))
      DO 11 I=1,P
      DO 11 J=1,N
 11   BUFF3(I,J)=BUFF2(I,J)
      DO 12 I=1,P
      DO 12 J=1,N
      II=I+L*P
 12   XMD(II,J)=BUFF2(I,J)
      IF(IECR.GT.8)WRITE(6,900)((BUFF2(I,J),J=1,N),I=1,P)
 900  FORMAT(4(1X,'BUFF2=',E10.4))
 13   CONTINUE
      NP=N*P
      IF(IECR.GT.3)WRITE(6,901)((XMD(I,J),J=1,N),I=1,NP)
 901  FORMAT(4(1X,'XMD=',E10.4))
C** ORTHOGONAL REDUCTION OF MD'=XMD
      IXMD=144
      EPSI1=1.E-3
      DO 14 I=1,N
      DO 14 J=1,N
 14   BUFF(I,J)=OMEGC(I,J)
      CALL ORTRED(XMD,BUFF,NP,N,IXMD,EPSI1,IECR)
      IF(IECR.GT.1)WRITE(6,902)((BUFF(I,J),J=1,N),I=1,N)
 902  FORMAT(4(1X,'OMEGI=',E10.4))
C** COMPUTATION OF RANKOMEGI AND LINEAR DEPENDENCY
       DO 16 I=1,N
       DO 16 J=1,N
 16    BUFF1(I,J)=BUFF(I,J)
      IOMEGI=12
      CALL MFGR(BUFF,IOMEGI,N,N,IRANK,IROW,ICOL,EPS,IER)
      IF(IECR.GT.1)WRITE(6,903)IER
 903  FORMAT(1X,'ERROR CODE IN COMPUTATION OF RANK OF OMEGI IS',I3)
      WRITE(6,904)IRANK
 904  FORMAT(1X,'RANK OF OMEGI IS',I3)
      IMU(JJ)=IRANK
      IF(IECR.GT.2)WRITE(6,905)(IROW(I),ICOL(I),I=1,N)
 905  FORMAT(4(1X,'IROW=',I3,'ICOL=',I3))
      IF(IECR.GT.2)WRITE(6,906)((BUFF(I,J),J=1,N),I=1,N)
 906  FORMAT(4(1X,'BUFF=',E10.4))
C**COMPUTATION OF THE GENERATOR
       WRITE(6,907)
 907   FORMAT(1X,'COMPUTATION OF THE GENERATOR GJJ')
      ICRI=4
       CALL ORTRED(XMD2,BUFF1,NP,N,IXMD,EPSI1,ICRI)
 15   CONTINUE
```

```
      WRITE(6,908)
  908 FORMAT(1X,'IF YOU WANT TO OVERCOME THE RANKS OF OMEGI',/,1X,
     1'GIVEN BY AUTOMATIC COMPUTATION,TYPE 00.OTHERWISE,TYPE11')
      READ(6,1000)ISIGN
 1000 FORMAT(I2)
      IF(ISIGN.EQ.11) GO TO 17
      WRITE(6,909)
  909 FORMAT(1X,'TYPE THE RANKS OF OMEGI,I=1,R')
      IR=R
      READ(5,*)(INU(I),I=1,IR)
      WRITE(6,*)(INU(I),I=1,IR)
      WRITE(6,910)
  910 FORMAT(1X,'IF SATISFIED,TYPE 11.IF NOT TYPE 00')
      READ(5,1000)ISIGN
      IF(ISIGN.EQ.0) GO TO 15
   17 CONTINUE
       CALL MAIN5A
      RETURN
      END
      SUBROUTINE MAIN5A
C** BUILDS THE MATRIX GG OF THE GENERATORS
      COMMON/MANI5A/GG
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
      DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12)
      DIMENSION BUFF(12,12),GG(12,12)
      INTEGER P,Q,R
      IF(IECR.GT.2) WRITE(6,100)
  100 FORMAT(1X,'SUBROUTINE MAIN5A')
    1 CONTINUE
      DO 3 II=1,N
    2 WRITE(6,101)II
  101 FORMAT(1X,'WRITE GG(I,J),J=1,R,I=',I3,'FREE FORMAT')
      IR=R
      READ(5,*)(GG(II,J),J=1,IR)
      WRITE(6,*)(GG(II,J),J=1,IR)
      WRITE(6,102)
  102 FORMAT(1X,'IF SATISFIED,TYPE 11.IF NOT,TYPE 00')
      READ(5,103)ISIGN
  103 FORMAT(I2)
      IF(ISIGN.EQ.0) GO TO 2
    3 CONTINUE
      WRITE(6,104)((GG(I,J),J=1,IR),I=1,N)
  104 FORMAT(4(1X,'GG=',E10.4))
      WRITE(6,102)
      READ(5,103)ISIGN
      IF(ISIGN.EQ.0) GO TO 1
      RETURN
      END
```

```
/*
/*EOJ */******
USERID GERARDJP CLASS A NAME MAIN5P      FORTRAN             06/27/78  20:33:39
:READ  MAIN5P   FORTRAN  A1 GERARD  5/17/78  11:23
      SUBROUTINE MAIN5P(XMD2)
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
      COMMON/TRASH2/BUFF1
      DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12)
      DIMENSION BUFF(12,12)
      DIMENSION XMD2(144,12),BUFF1(12,12)
      INTEGER P,Q,R
      IF(IECR.GT.2)WRITE(6,99)
 99   FORMAT(1X,'SUBROUTINE MAIN5P')
C**COMPUTATION OF M
      DO 1 I=1,P
      DO 1 J=1,N
      XMD2(I,J)=C(I,J)
 1    BUFF(I,J)=C(I,J)
      NN=N-1
      DO 5 L=1,NN
      NM1=L*P
      DO 2 I=1,P
      DO 2 J=1,N
      BUFF1(I,J)=0.E0
      DO 2 K=1,N
 2    BUFF1(I,J)=BUFF1(I,J)+BUFF(I,K)*A(K,J)
      IF(IECR.GT.8)WRITE(6,100)((BUFF1(I,J),J=1,N),I=1,P)
 100  FORMAT(4(1X,'BUFF1=',E10.4))
      DO 3 I=1,P
      DO 3 J=1,N
 3    BUFF(I,J)=BUFF1(I,J)
      DO 4 I=1,P
      DO 4 J=1,N
      II=I+L*P
 4    XMD2(II,J)=BUFF(I,J)
 5    CONTINUE
      NP=N*P
      IF(IECR.GT.3)WRITE(6,200)((XMD2(I,J),J=1,N),I=1,NP)
 200  FORMAT(4(1X,'XMD2=',E10.4))
      RETURN
      END
```

```
      SUBROUTINE MAIN6
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
      COMMON/MANI5/INU,INUS,INJO
      DIMENSION INU(12)
      DIMENSION A(12,12),B(12,12),C(12,12),CBI(12,12),BUFF(12,12)
      DIMENSION BI(12,12)
      INTEGER P,Q,R
C** CHECK WHETHER SUM NUI=NUS
C** COMPUTATION OF DETECTION SPACE DIMENSION
      DO 1 I=1,R
1     INU(I)=INU(I)+1
      IF(IECR.GT.1)WRITE(6,100)(I,INU(I),I=1,R)
100   FORMAT(1X,'DETECTION SPACE OF B',I3,'HAS A DIMENSION ',I3)
      INUS=INUS+R
      IF(IECR.GT.1)WRITE(6,200)INUS
200   FORMAT(1X,'DETECTION SPACE ASSOCIATED WITH THE SET OF BI ',/,
     11X,'SELECTED HAS A DIMENSION',I3)
      ISUM=0
      DO 2 I=1,R
2     ISUM=ISUM+INU(I)
      WRITE(6,201)ISUM
201   FORMAT(1X,'SUM OF NUI IS',I3)
      INUO=INUS-ISUM
3     WRITE(6,202)
202   FORMAT(1X,'IF THE SUM OF THE NUI IS EQUAL TO NUS-DIMENSION',/,1X,
     1'OF THE DETECTION SPACE ASSOCIATED WITH THE SET OF BI-THESE',/,1X,
     1'BIS ARE MUTUALLY DETECTABLE.A DETECTION FILTER CAM BE ',/,1X,
     1'DESIGNED FOR THIS SET WITH ASSIGNABLE EIGENVALUES.',/,1X,
     1'IF IT IS NOT.A TOTAL OF NUS-(SUM OF NUI) EIGENVALUES ARE',/,1X,
     1'UNASSIBNABLE.IF YOU WANT TO CHECK THEIR VALUE,TYPE11,',/,1X,
     1'OTHERWISE TYPE 00')
      READ(5,10)ISIGN
      WRITE(5,10)ISIGN
10    FORMAT(I2)
      IF(ISIGN.NE.11) GO TO 4
      WRITE(6,203)
203   FORMAT(1X,'THE PROGRAM WILL PROCEED ON COMPUTING THE UNASSIGN.',/,
     11X,'EIGENVALUES.IF YOU MADE A MISTAKE IN SELECTING THE OPTION',/,
     11X,'TYPE 00.OTHERWISE TYPE 11')
      READ(5,10)ISIGN1
      WRITE(6,10)ISIGN1
      IF(ISIGN1.EQ.0)GO TO 3
      CALL MAIN6A
      RETURN
4     WRITE(6,204)
204   FORMAT(1X,'THE PROGRAM WILL NOT COMPUTE THE UNASSIGN.',/,1X,
     1'EIGENVALUES IF YOU MADE A MISTAKE IN SELECTING THE OPTION TYPE00
     1'.',/,1X,'OTHERWISE TYPE 11')
      READ(5,10)ISIGN1
      WRITE(6,10)ISIGN1
      IF(ISIGN1.EQ.0) GO TO 3
      RETURN
      END
```

```
      SUBROUTINE MAIN6A
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
      COMMON/MANI4/OMEGC,DS,CS
      COMMON/MANI5/INU,INUS,INJO
      COMMON/MANI6/BUFF1
C** BUFF1=(CBIT*CBI)-1*CBIT
C** COMES FROM MAIN3
      COMMON/MANI6A/ADSC
      COMMON/MANI6B/XMO,IND
      COMMON/TRASH2/BUFF3
      DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12)
      DIMENSION OMEGC(12,12),DS(12,12),CS(12,12),BUFF1(12,12)
      DIMENSION BUFF(12,12),ADSC(12,12),XMO(12,12),BUFF3(12,12),INU(12)
      INTEGER P,Q,R
      IF(IECR.GT.2)WRITE(6,100)
 100   FORMAT(1X,'SUBROUTINE MAIN6A')
C** COMPUTATION OF (CBIT*CBI)-1*CBIT*C=CI
      DO 1 I=1,R
      DO 1 J=1,N
      BUFF(I,J)=0.E0
      DO 1 K=1,P
 1    BUFF(I,J)=BUFF(I,J)+BUFF1(I,K)*C(K,J)
      IF(IECR.GT.5)WRITE(6,101)((BUFF(I,J),J=1,N),I=1,R)
 101  FORMAT(4(1X,'CI=',E10.4))
      IF(IECR.GT.8)WRITE(6,200)((BUFF1(I,J),J=1,P),I=1,R)
 200   FORMAT(4(1X,'BUFF1=',E10.4))
C**  COMPUTATION OF M0=XMO
      IF(IECR.GT.8)WRITE(6,201)((ADSC(I,J),J=1,N),I=1,N)
 201   FORMAT(4(1X,'A-DS*C=',E10.4))
      IND=1
      DO 9 II=1,R
      INU1=INU(II)-2
      DO 2 J=1,N
 2    XMO(IND,J)=BUFF(II,J)
      DO 3 I=1,N
      DO 3 J=1,N
 3    BUFF1(I,J)=ADSC(I,J)
      IND=IND+1
      INU2=-INU1
       IF(INU2.EQ.1) GO TO 9
      DO 4 J=1,N
      XMO(IND,J)=0.E0
      DO 4 K=1,N
 4    XMO(IND,J)=XMO(IND,J)+BUFF(II,K)*BUFF1(K,J)
      IND=IND+1
       IF(INU2.EQ.0) GO TO 9
      DO 8 JJ=1,INU1
      DO 5 I=1,N
      DO 5 J=1,N
      BUFF3(I,J)=0.E0
      DO 5 K=1,N
 5    BUFF3(I,J)=BUFF3(I,J)+BUFF1(I,K)*ADSC(K,J)
      DO 6 J=1,N
      XMO(IND,J)=0.E0
      DO 6 K=1,N
 6    XMO(IND,J)=XMO(IND,J)+BUFF(I,K)*BUFF3(K,J)
      IND=IND+1
      DO 7 I=1,N
      DO 7 J=1,N
 7    BUFF1(I,J)=BUFF3(I,J)
 8    CONTINUE
```

```
 9    CONTINUE
      IF(IECR.GT.5)WRITE(6,300)((XMO(I,J),J=1,N),I=1,IND)
 300  FORMAT(4(1X,'MO=',E10.4))
      DO 10 I=1,N
      DO 10 J=1,N
 10   BUFF1(I,J)=BUFF(I,J)
C** BUFF1=CI IN COMMON MANI6
```

```
      CALL MAIN6B
      RETURN
      SUBROUTINE MAIN6A
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
      COMMON/MANI4/OMEGC,DS,CS
      COMMON/MANI5/INU,INUS,INJO
      COMMON/MANI6/BUFF1
C** BUFF1=(CBIT*CBI)-1*CBIT
C** COMES FROM MAIN3
      COMMON/MANI6A/ADSC
      COMMON/MANI6B/XMO,IND
      COMMON/TRASH2/BUFF3
      DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12)
      DIMENSION OMEGC(12,12),DS(12,12),CS(12,12),BUFF1(12,12)
      DIMENSION BUFF(12,12),ADSC(12,12),XMO(12,12),BUFF3(12,12),INU(12)
      INTEGER P,Q,R
      IF(IECR.GT.2)WRITE(6,100)
 100  FORMAT(1X,'SUBROUTINE MAIN6A')
C** COMPUTATION OF (CBIT*CBI)-1*CBIT*C=CI
      DO 1 I=1,R
      DO 1 J=1,N
      BUFF(I,J)=0.E0
      DO 1 K=1,P
 1    BUFF(I,J)=BUFF(I,J)+BUFF1(I,K)*C(K,J)
      IF(IECR.GT.5)WRITE(6,101)((BUFF(I,J),J=1,N),I=1,R)
 101  FORMAT(4(1X,'CI=',E10.4))
      IF(IECR.GT.8)WRITE(6,200)((BUFF1(I,J),J=1,P),I=1,R)
 200  FORMAT(4(1X,'BUFF1=',E10.4))
C**   COMPUTATION OF M0=XMO
      IF(IECR.GT.8)WRITE(6,201)((ADSC(I,J),J=1,N),I=1,N)
 201  FORMAT(4(1X,'A-DS*C=',E10.4))
      IND=1
      DO 9 II=1,R
      INU1=INU(II)-2
      DO 2 J=1,N
 2    XMO(IND,J)=BUFF(II,J)
      DO 3 I=1,N
      DO 3 J=1,N
 3    BUFF1(I,J)=ADSC(I,J)
      IND=IND+1
      INU2=-INU1
      IF(INU2.EQ.1) GO TO 9
      DO 4 J=1,N
      XMO(IND,J)=0.E0
      DO 4 K=1,N
 4    XMO(IND,J)=XMO(IND,J)+BUFF(II,K)*BUFF1(K,J)
      IND=IND+1
      IF(INU2.EQ.0) GO TO 9
      DO 8 JJ=1,INU1
      DO 5 I=1,N
      DO 5 J=1,N
      BUFF3(I,J)=0.E0
      DO 5 K=1,N
 5    BUFF3(I,J)=BUFF3(I,J)+BUFF1(I,K)*ADSC(K,J)
      DO 6 J=1,N
      XMO(IND,J)=0.E0
      DO 6 K=1,N
 6    XMO(IND,J)=XMO(IND,J)+BUFF(I,K)*BUFF3(K,J)
      IND=IND+1
      DO 7 I=1,N
      DO 7 J=1,N
```

```
 7      BUFF1(I,J)=BUFF3(I,J)
 8      CONTINUE
 9      CONTINUE
        IF(IECR.GT.5)WRITE(6,300)((XMO(I,J),J=1,N),I=1,IND)
 300    FORMAT(4(1X,'MO=',E10.4))
        DO 10 I=1,N
        DO 10 J=1,N
 10     BUFF1(I,J)=BUFF(I,J)
C** BUFF1=CI IN COMMON MANI6
        CALL MAIN63
        RETURN
        END
```

```
      SUBROUTINE MAIN6B
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
      COMMON/MANI3/IROW,ICOL
      COMMON/MANI4/OMEGC,DS,CS
      COMMON/MANI6B/XMO,IND
      COMMON/MANI6C/BUFF1
      DIMENSION XMO(12,12),OMEGC(12,12),DS(12,12),CS(12,12)
      DIMENSION BUFF(12,12),A(12,12),B(12,12),C(12,12),BI(12,12)
      DIMENSION CBI(12,12),BUFF1(12,12),ICOL(12),IROW(12)
      INTEGER P,Q,R
      IF(IECR.GT.2)WRITE(6,100)
 100  FORMAT(1X,'SUBROUTINE MAIN6B')
C** ORTHOGONAL REDUCTION OF XMO=MO
      DO 1 I=1,N
      DO 1 J=1,N
 1    BUFF(I,J)=OMEGC(I,J)
      IBUFF=12
      EPSI1=1.E-9
      CALL ORTRED(XMO,BUFF,IND,N,IBUFF,EPSI1,IECR)
      IF(IECR.GT.2)WRITE(6,101)((BUFF(I,J),J=1,N),I=1,N)
 101  FORMAT(4(1X,'OMEGOG=',E10.4))
C** COMPUTATION OF THE LINEAR DEPENDENCY OF THE COLUMNS OF OMEGOG
      DO 2 I=1,N
      DO 2 J=1,N
 2    BUFF1(I,J)=BUFF(I,J)
      CALL MFGR(BUFF,IBUFF,N,N,IRANK,IROW,ICOL,EPSI1,IER)
      IF(IECR.GT.2)WRITE(6,200)IER
 200  FORMAT(1X,'ERROR CODE IN COMPUTATION OF RANK OF OMEGOG IS',I3)
      IF(IECR.GT.2)WRITE(6,300)IRANK
 300  FORMAT(1X,'RANK OF OMEGOG IS',I3)
      IF(IECR.GT.2)WRITE(6,400)(IROW(I),ICOL(I),I=1,N)
 400  FORMAT(1X,'IROW=',I3,'ICOL=',I3)
      IF(IECR.GT.2)WRITE(6,500)((BUFF(I,J),J=1,N),I=1,N)
 500  FORMAT(4(1X,'DEP OF OMEGOG',E10.4))
      CALL MAIN6C
      RETURN
      END
      SUBROUTINE MAIN6C
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
      COMMON/MANI5/INU,INUS,INUO
      COMMON/MANI6C/BUFF1
      COMMON/MANI6D/ROG
      COMMON/TRASH1/IZ
C**IN COMMON MANI6C,BUFF1=OMEGOG,COMES FROM MAIN6B
      DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12)
      DIMENSION BUFF(12,12),BUFF1(12,12),IZ(12),ROG(12,12)
      DIMENSION INU(12)
      INTEGER P,Q,R
      IF(IECR.GT.2) WRITE(6,99)
 99   FORMAT(1X,'SUBROUTINE MAIN6C')
C** SELECTION OF ROG
 1    WRITE(6,100)
 100  FORMAT(1X,'IDENTIFY WHICH COLUMNS OF OMEGOG YOU WISH TO',/,1X,
     1'KEEP BY TYPING THEIR NUMBERS IN FORMAT I3.IF NUO=NUS-(SUM NI)',/,
     11X,'INDICATE WHAT ARE THE NUO FIRST LINEARLY INDEPENDENT',/,1X,
     1'COLUMNS OF OMEGOG')
      READ(5,1000)(IZ(I),I=1,INUO)
 1000 FORMAT(12I3)
      WRITE(6,1000)(IZ(I),I=1,INUO)
      WRITE(6,101)
 101  FORMAT(1X,'IF DATA ARE INCORRECTLY ENTERED,TYPE 00,OTHERWISE',/,1X
```

```
     1,,TYPE 11')
      READ(5,1001)ISIGN
 1001 FORMAT(I2)
      IF(ISIGN.EQ.0) GO TO 1
      DO 2 J=1,INUO
      JJ=IZ(I)
      DO 2 I=1,N
 2    ROG(I,J)=BUFF1(I,JJ)
      IF(IECR.GT.2)WRITE(6,200)((ROG(I,J),J=1,INUO),I=1,N)
 200  FORMAT(4(1X,'ROG=',E10.4))
C** ROG IS A (N*INUO) MATRIX
      CALL MAIN6D
      RETURN
      END
      SUBROUTINE MAIN6D
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
      COMMON/MANI6/CI
      COMMON/MANI5/INU,INUS,INJO
      COMMON/MANI6D/ROG
C** CI HAS BEEN COMPUTED IN MAIN6A(WAS BUFF1)
      COMMON/MANI6A/ADSC
      COMMON/MANI6E/TETA
      DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12)
      DIMENSION CI(12,12),BUFF(12,12),INU(12),ROG(12,12),TETA(12,12)
      DIMENSION ADSC(12,12)
      INTEGER P,Q,R
C** COMPUTATION OF TETA MATRIX
      IF(IECR.GT.2)WRITE(6,99)
 99   FORMAT(1X,'SUBROUTINE MAIN6D')
      IF(IECR.GT.8)WRITE(6,98)((CI(I,J),J=1,N),I=1,R)
 98   FORMAT(4(1X,'CI=',E10.4))
      IF(IECR.GT.8)WRITE(6,97)((ADSC(I,J),J=1,N),I=1,N)
 97   FORMAT(4(1X,'ADSC=',E10.4))
      DO 6 II=1,R
      NUI=INU(II)-1
      DO 1 J=1,N
      TETA(II,J)=0.E0
      DO 1 K=1,N
 1    TETA(II,J)=TETA(II,J)+CI(II,K)*ADSC(K,J)
      DO 2 J=1,N
 2    BUFF(II,J)=TETA(II,J)
      IF(NUI.EQ.0) GO TO 6
      DO 5 JJ=1,NUI
      DO 3 J=1,N
      TETA(II,J)=0.E0
      DO 3 K=1,N
 3    TETA(II,J)=TETA(II,J)+BUFF(II,K)*ADSC(K,J)
      DO 4 K=1,N
 4    BUFF(II,K)=TETA(II,K)
 5    CONTINUE
 6    CONTINUE
      IF(IECR.GT.8)WRITE(6,100)((TETA(I,J),J=1,N),I=1,R)
 100  FORMAT(4(1X,'TET=',E10.4))
      DO 7 I=1,R
      DO 7 J=1,INUO
      BUFF(I,J)=0.E0
      DO 7 K=1,N
 7    BUFF(I,J)=BUFF(I,J)+TETA(I,K)*ROG(K,J)
      DO 8 I=1,R
      DO 8 J=1,INUO
 8    TETA(I,J)=BUFF(I,J)
```

```
C**TETA IS A (R*INUO) MATRIX
      IF(IECR.GT.5)WRITE(6,200)((TETA(I,J),J=1,INUO),I=1,R)
 200  FORMAT(4(1X,'TETA=',E10.4))
      CALL MAIN6E
      RETURN
      END
```

```
      SUBROUTINE MAIN6E
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
      COMMON/MANI5/INU,INUS,INUO
      COMMON/MANI5A/GG
      COMMON/MANI6D/ROG
      COMMON/MANI6E/TETA
      COMMON/MANI6F/XPI
      COMMON/TRASH2/CBVEC,CBVIV,BUFF1,BUFF2,BUFF3
      DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12)
      DIMENSION BUFF(12,12),ROG(12,12),GG(12,12),TETA(12,12)
      DIMENSION XPI(12,12),INU(12),CBVEC(78),CBVIV(78),BUFF1(12,12)
      DIMENSION BUFF2(12,12),BUFF3(12,12)
      INTEGER P,Q,R
      IF(IECR.GT.2)WRITE(6,100)
  100 FORMAT(1X,'SUBROUTINE MAIN6E')
C** COMPUTATION OF MATRIX XPI=PI
C** COMPUTATION OF (ROGT*ROG)
      DO 1 I=1,INUO
      DO 1 J=1,INUO
      BUFF(I,J)=0.E0
      DO 1 K=1,N
    1 BUFF(I,J)=BUFF(I,J)+ROG(K,I)*ROG(K,J)
C** COMPUTATION OF (ROGT*ROG)-1
C** TRANSITION TO SYMMETRIC STORAGE
      IR=12
      IF(IECR.GT.5)WRITE(6,99)INUO
   99 FORMAT(1X,'INUO=',I3)
      IF(INUO.EQ.1) GO TO 11
      CALL VCVTFS(BUFF,INUO,IR,CBVEC)
      IF(IECR.GT.8)WRITE(6,101)((BUFF(I,J),J=1,INUO),I=1,INUO)
  101 FORMAT(4(1X,'BUFF=',E10.4))
      IF(IECR.GT.8)WRITE(6,102)(CBVEC(I),I=1,12)
  102 FORMAT(4(1X,'CBVEC=',F10.4))
C** COMPUTATION OF INVERSE OF CBVEC
      CALL LINV1P(CBVEC,INUO,CBVIV,IDGT,D1,D2,IER)
      IF(IECR.GT.8)WRITE(6,103)(CBVIV(I),I=1,12)
  103 FORMAT(4(1X,'CBVIV=',F10.4))
C** COMPUTATION OF (ROGT*ROG)-1
      CALL VCVTSF(CBVIV,INUO,BUFF,IR)
      GO TO 12
   11 CONTINUE
      BUFF(1,1)=1./BUFF(1,1)
   12 CONTINUE
      IF(IECR.GT.5)WRITE(6,104)((BUFF(I,J),J=1,INUO),I=1,INUO)
  104 FORMAT(4(1X,'ROGT*ROG-1 =',E10.4))
C** COMPUTATION OF (ROGT*ROG)-1*ROGT
      DO 2 I=1,INUO
      DO 2 J=1,N
      BUFF1(I,J)=0.E0
      DO 2 K=1,INUO
    2 BUFF1(I,J)=BUFF1(I,J)+BUFF(I,K)*ROG(J,K)
      IF(IECR.GT.8)WRITE(6,105)((BUFF1(I,J),J=1,N),I=1,INUO)
  105 FORMAT(4(1X,'BUFF1=',E10.4))
C** COMPUTATION OF A*ROG
      DO 3 I=1,N
      DO 3 J=1,INUO
      BUFF2(I,J)=0.E0
      DO 3 K=1,N
    3 BUFF2(I,J)=BUFF2(I,J)+A(I,K)*ROG(K,J)
      IF(IECR.GT.8)WRITE(6,106)((BUFF2(I,J),J=1,INUO),I=1,N)
  106 FORMAT(4(1X,'A*ROG=',E10.4))
```

```
C** COMPUTATION OF G*TETA
      DO 4 I=1,N
      DO 4 J=1,INUO
      BUFF(I,J)=0.E0
      DO 4 K=1,R
 4    BUFF(I,J)=BUFF(I,J)+GG(I,K)*TETA(K,J)
      IF(IECR.GT.8)WRITE(6,107)((BUFF(I,J),J=1,INUO),I=1,N)
 107   FORMAT(4(1X,'G*TETA=',E10.4))
C** COMPUTATION OF A*ROG-G*TETA
      DO 5 I=1,N
      DO 5 J=1,INUO
 5    BUFF3(I,J)=BUFF2(I,J)-BUFF(I,J)
C** COMPUTATION OF XPI
C** XPI IS A INUO*INUO MATRIX
      DO 6 I=1,INUO
      DO 6 J=1,INUO
      XPI(I,J)=0.E0
      DO 6 K=1,N
 6    XPI(I,J)=XPI(I,J)+BUFF1(I,K)*BUFF3(K,J)
      IF(IECR.GT.5)WRITE(6,108)((XPI(I,J),J=1,INUO),I=1,INUO)
 108  FORMAT(4(1X,'XPI=',E10.4))
C** COMPUTATION OF XPI EIGENVALUES
      CALL MAIN6F
      RETURN
      END
      SUBROUTINE MAIN6F
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
      COMMON/MANI5/INU,INUS,INJO
      COMMON/MANI6F/XPI
      COMMON/MANI66/W
      COMMON/TRASH2/BUFF1
      DIMENSION XPI(12,12),INU(12),BUFF(12,12),Z(1,1)
      DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12)
      DIMENSION BUFF1(12,12)
      COMPLEX W(12)
      INTEGER P,Q,R
      IF(IECR.GT.2)WRITE(6,99)
 99   FORMAT(1X,'SUBROUTINE MAIN6F')
C** COMPUTATION OF XPI EIGENVALUES
      DO 1 I=1,INUO
      DO 1 J=1,INUO
 1    BUFF1(I,J)=XPI(I,J)
      IJOB=0
      KWK=13
      IBUFF=12
      IZ=1
      CALL EIGRF(BUFF1,INUO,IBUFF,IJOB,W,Z,IZ,KWK,IER)
      WRITE(6,100)IER
 100  FORMAT(1X,'ERROR CODE IN COMPUTATION OF XPI EIGENVALUES IS',I3)
      WRITE(6,101)
 101  FORMAT(1X,'IF IER IS GREATER THAN 128,EIGENVALUES ARE NOT CORRECT'
     1)
      WRITE(6,102)(W(I),I=1,INJO)
 102  FORMAT(1X,'UNASS.EIGENV.',2(E10.4,2X,E10.4))
      RETURN
      END
```

```
      SUBROUTINE MAIN7
      INTEGER P,Q,R
C** STEP 5G
      WRITE(6,1000)
 1000 FORMAT(1X,'THREE POSSIBILITIES ARE OPEN NOW:',/,1X,
     1'ACCEPT THE UNASSIGNABLE EIGENVALUES OF THE DETECTION FILTER GOIN'
     1,/,1X,'WITH THIS SET OF EVENTS',/,1X,
     1'LOOK FOR A SUBSET OF THE ACTUAL BIS WHICH DOES NOT YIELD UNASSI.'
     1/,1X,'EIGENVALUES',/,1X,
     1'INCREASE THE DIMENSION OF THE REFERENCE MODEL')
      WRITE(6,1001)
 1001 FORMAT(1X,'THE LAST POSSIBILITY IS NOT YET OPERATIONAL')
 1    WRITE(6,1002)
 1002  FORMAT(1X,'IF YOU WANT TO FIND A SUBSET OF THE ACTUAL BIS',/,1X,
     1'WITH NO UNASSIGNABLE EIGENVALUES,TYPE 11.OTHERWISE,TYPE 00')
      READ(5,100)ISIGN
 100  FORMAT(I2)
      IF(ISIGN.NE.11) GO TO 3
      WRITE(6,1003)
 1003  FORMAT(1X,'THE PROGRAM WILL LOOK FOR A SUBSET OF THE BIS',/,1X,
     1'WITH NO UNASS. EIGENVALUES.IF YOU MADE A MISTAKE IN',/,1X,
     1'SELECTING THE OPTION,TYPE 00.OTHERWISE,TYPE 11')
      READ(5,100) ISIGN1
      WRITE(6,100)ISIGN1
      IF(ISIGN1.EQ.0) GO TO 1
      CALL MAIN7A
      RETURN
 3    WRITE(6,1006)
 1006 FORMAT(1X,'THE PROGRAM WILL DESIGN A FILTER WITH THE BIS',/,1X,
     1'AS EVENTS.IF YOU MADE A MISTAKE IN SELECTING THE OPTION,TYPE 00'
     1,/,1X,'OTHERWISE,TYPE 11')
      READ(5,100)ISIGN2
      WRITE(6,100)ISIGN2
      IF(ISIGN2.EQ.0) GO TO 1
      RETURN
      END
      SUBROUTINE MAIN7A
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
      COMMON/MANI3/IROW,ICOL
      COMMON/MANI5/INU,INUS,INUO
      COMMON/MANI6E/TETA
      COMMON/MANI6F/XPI
      COMMON/TRASH2/BUFF1,BUFF2,XMOI
      DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12)
      DIMENSION INU(12),BUFF(12,12),TETA(12,12),XPI(12,12),BUFF1(12,12)
      DIMENSION BUFF2(12,12),XMOI(12,12),IROW(12),ICOL(12)
      INTEGER P,Q,R
      IF(IECR.GT.2)WRITE(6,99)
 99   FORMAT(2X,'SUBROUTINE MAIN7A')
C** STEP 5H
C** COMPUTATION FOR ALL BIS
      DO 12 II=1,R
      WRITE(6,98)II
 98   FORMAT(1X,'COMPUTATION OF THE EIGENVALUES ASSOCIATED WITH B',I2)
      IND=1
C** COMPUTATION OF MOI
      NUI=INUO-1
      DO 1 J=1,INUO
      XMOI(IND,J)=TETA(II,J)
 1    BUFF(1,J)=TETA(II,J)
      IF(NUI.EQ.0) GO TO 4
```

```
      DO 4 JJ=1,NUI
      IND=IND+1
      DO 2 J=1,INUO
      BUFF1(IND,J)=0.E0
      DO 2 K=1,INUO
 2    BUFF1(IND,J)=BUFF1(IND,J)+BUFF(1,K)*XPI(K,J)
      DO 3 J=1,INUO
 3    BUFF(1,J)=BUFF1(IND,J)
      DO 33 J=1,INUO
 33   XMOI(IND,J)=BUFF1(IND,J)
 4    CONTINUE
      IF(IECR.GT.5)WRITE(6,100)((XMOI(I,J),J=1,INUO),I=1,INUO)
 100  FORMAT(4(1X,'MOI=',E10.4))
C** ORTHOGONAL REDUCTION OF MOI
      DO 6 I=1,INUO
      DO 5 J=1,INUO
 5    BUFF(I,J)=0.E0
 6    BUFF(I,I)=1.E0
      IF(IECR.GT.12)WRITE(6,101)((BUFF(I,J),J=1,INUO),I=1,INUO)
 101  FORMAT(4(1X,'BUFF=',E10.4))
      IXMOI=12
      EPSI1=0.0001 E0
      CALL ORTRED(XMOI,BUFF,INJO,INUO,IXMOI,EPSI1,IECR)
      IF(IECR.GT.5)WRITE(6,102)((BUFF(I,J),J=1,INUO),I=1,INUO)
 102   FORMAT(4(1X,'BUFF=',F10.4))
C** COMPUTATION OF BETA
      DO 7 I=1,INUO
      DO 7 J=1,INUO
 7    BUFF1(I,J)=BUFF(I,J)
      IF(IECR.GT.8)WRITE(6,97)INUO
      IF(INUO.EQ.1) GO TO 66
      IBUFF=12
      CALL MFGR(BUFF1,IXMOI,INJO,INUO,IRANK,IROW,ICOL,EPSI1,IER)
      IF(IECR.GT.2)WRITE(6,192)IER
 192  FORMAT(1X,'ERROR CODE IN COMPUTATION OF RANK OF BETAP IS',I3)
      IF(IECR.GT.2)WRITE(6,104)(IROW(I),ICOL(I),I=1,INUO)
 104  FORMAT(1X,'IROW=',I3,'ICOL=',I3)
      IF(IECR.GT.2)WRITE(6,105)((BUFF1(I,J),J=1,INUO),I=1,INUO)
 105  FORMAT(4(1X,'DEP OF BETA',E10.4))
      GO TO 666
 66   CONTINUE
      IRANK=1
      IF(ABS(BUFF1(1,1)).LT.EPSI1) IRANK=0
 666  CONTINUE
      IF(IECR.GT.2)WRITE(6,103)IRANK
 103  FORMAT(1X,'RANK OF BETAP IS',I3)
      IF(IRANK.EQ.0) GO TO 77
      CALL MAN7AA(IRANK)
 77   CONTINUE
C** BETA=BUFF
      IF(IECR.GT.12)WRITE(6,106)((BUFF(I,J),J=1,IRANK),I=1,INUO)
 106  FORMAT(4(1X,'BETA=',E10.4))
      DO 8 I=1,INUO
      DO 8 J=1,IRANK
 8    BUFF2(I,J)=BUFF(I,J)
C** BETA=BUFF2
C** COMPUTATION OF DELTA
      DO 9 I=1,INUO
      DO 9 J=1,INUO
      BUFF(I,J)=XMOI(I,J)
 9    BUFF1(I,J)=XMOI(I,J)
```

```
      IF(IECR.GT.8)WRITE(6,97)INUO
 97   FORMAT(1X,'INUO=',I3)
      IF(INUO.EQ.1) GO TO 88
      CALL MFGR(BUFF1,IXMOI,INUO,INUO,IRANK1,IROW,ICOL,EPSI1,IER)
      IF(IECR.GT.2)WRITE(6,107)IER
 107  FORMAT(1X,'ERROR CODE IN COMPUTATION OF RANK OF XMOI IS',I3)
      IF(IECR.GT.2)WRITE(6,109)(IROW(I),ICOL(I),I=1,INUO)
 109  FORMAT(1X,'IROW=',I3,'ICOL=',I3)
      IF(IECR.GT.2)WRITE(6,110)((BUFF1(I,J),J=1,INUO),I=1,INUO)
 110  FORMAT(4(1X,'DEP OF DELTA IS',E10.4))
      GO TO 888
 88   CONTINUE
      IRANK1=1
      IF(ABS(BUFF1(1,1)).LT.EPSI1) IRANK1=0
 888  CONTINUE
      IF(IECR.GT.2)WRITE(6,108)IRANK1
 108  FORMAT(1X,'RANK OF XMOI IS',I3)
      IF(IRANK1.EQ.0) GO TO 999
      CALL MAN7AB(IRANK1)
 999  CONTINUE
C** DELTA=BUFF
      IF(IECR.GT.12)WRITE(6,111)((BUFF(I,J),J=1,IRANK1),I=1,INUO)
 111  FORMAT(4(1X,'DELTA= ',E10.4))
      ISUM=IRANK+IRANK1
      IF(ISUM.EQ.INUO) GO TO 10
      WRITE(6,112)
 112  FORMAT(1X,'IRANK+IRANK1.NE.INUO ,THERE IS A MISTAKE')
      STOP
 10   CONTINUE
C** FORMATION OF DELTA:BETA
C** DELTA:BETA IS A INUO*INUO MATRIX
      DO 11 I=1,INUO
      IF(IRANK.EQ.0) GO TO 1110
      DO 11 J=1,IRANK
      JJ=IRANK1+J
 11   BUFF(I,JJ)=BUFF2(I,J)
      GO TO 1130
 1110 CONTINUE
 1130 CONTINUE
      IF(IECR.GT.5)WRITE(6,113)((BUFF(I,J),J=1,INUO),I=1,INUO)
 113  FORMAT(4(1X,'DELTA:BETA=',E10.4))
C** COMPUTATION OF THE SETS LAMBDAI
      CALL MAN7AC(IRANK)
 12   CONTINUE
      CALL MAIN7B
      RETURN
      END
```

```
      SUBROUTINE MAN7AA(IRANK)
C** SELECTION OF BETA FROM THE COLUMNS OF BUFF=BETAP
C** BUFF COMES FROM MAIN7A
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
      COMMON/MANI5/INU,INUS,INUO
      COMMON/TRASH1/IZ
      COMMON/TRASH2/BUFF1
      DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12)
      DIMENSION BUFF(12,12),BUFF1(12,12),IZ(12),INU(12)
      INTEGER P,Q,R
      DO 1 I=1,INUO
      DO 1 J=1,INUO
 1    BUFF1(I,J)=BUFF(I,J)
      IF(IECR.GT.12)WRITE(6,100)((BUFF(I,J),J=1,INUO),I=1,INUO)
 100  FORMAT(4(1X,'BUFF1=',E10.4))
      IF(IECR.GT.2)WRITE(6,101)
 101  FORMAT(1X,'SUBROUTINE MAIN7AA')
 2    WRITE(6,102)
 102  FORMAT(1X,'IDENTIFY WHICH COLUMNS OF BETAP YOU WISH TO KEEP',/,1X,
     1'TYPE THEIR NUMBERS IN FORMAT I3.YOU MUST KEEP RANK OF BETAP COL')
      READ(5,1000)(IZ(I),I=1,IRANK)
 1000 FORMAT(12I3)
      WRITE(6,1000)(IZ(I),I=1,IRANK)
      WRITE(6,103)
 103  FORMAT(1X,'IF DATA ARE INCORRECTLY ENTERED,TYPE 00',/,1X,
     1'OTHERWISE TYPE 11')
      READ(5,1001) ISIGN
 1001 FORMAT(I2)
      IF(ISIGN.EQ.0) GO TO 2
      DO 3 J=1,IRANK
      JJ=IZ(J)
      DO 3 I=1,INUO
 3    BUFF(I,J)=BUFF1(I,JJ)
      IF(IECR.GT.5)WRITE(6,104)((BUFF(I,J),J=1,IRANK),I=1,INUO)
 104  FORMAT(4(1X,'BETA=',E10.4))
      RETURN
      END
      SUBROUTINE MAN7AB(IRANK1)
C** SELECTION OF DELTA FROM THE COLUMNS OF BUFF
C** BUFF COMES FROM MAIN7A
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
      COMMON/MANI5/INU,INUS,INUO
      COMMON/TRASH1/IZ
      COMMON/TRASH2/BUFF1
      DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12)
      DIMENSION BUFF(12,12),BUFF1(12,12),IZ(12),INU(12)
      INTEGER P,Q,R
      DO 1 I=1,INUO
      DO 1 J=1,INUO
 1    BUFF1(I,J)=BUFF(I,J)
      IF(IECR.GT.12)WRITE(6,100)((BUFF(I,J),J=1,INUO),I=1,INUO)
 100  FORMAT(4(1X,'BUFF1=',F10.4))
      IF(IECR.GT.2)WRITE(6,101)
 101  FORMAT(2X,'SUBROUTINE MAIN7AB')
 2    WRITE(6,102)
 102  FORMAT(1X,'IDENTIFY WHICH COLUMNS OF XMOI YOU WISH TO KEEP',/,1X,
     1'TYPE THEIR NUMBERS IN FORMAT I3.YOU MUST KEEP RK OF XMOI COL')
      READ(5,1000)(IZ(I),I=1,IRANK1)
 1000 FORMAT(12I3)
      WRITE(6,1000)(IZ(I),I=1,IRANK1)
      WRITE(6,103)
```

```
 103  FORMAT(1X,'IF DATA ARE INCORRECTLY ENTERED,TYPE 00',/,1X,
     1'OTHERWISE TYPE 11')
      READ(5,1001) ISIGN
 1001 FORMAT(I2)
      IF(ISIGN.EQ.0) GO TO 2
      DO 3 J=1,IRANK1
      JJ=IZ(J)
      DO 3 I=1,INUO
 3    BUFF(I,J)=BUFF1(I,JJ)
      IF(IECR.GT.5)WRITE(6,104)((BUFF(I,J),J=1,IRANK1),I=1,INUO)
 104  FORMAT(4(1X,'BETA=',E10.4))
      RETURN
      END
      SUBROUTINE MAN7AC(IRANK)
C** COMPUTES THE SETS OF LAMBDAI
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
C** BUFF=DELTA:BETA ,COMES FROM MAIN7A
      COMMON/MANI5/INU,INUS,INUO
      COMMON/MANI6F/XPI
      COMMON/TRASH2/BUFF1,BUFF2
      COMPLEX WI(12)
      DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12)
      DIMENSION Z(1,1),BUFF(12,12),XPI(12,12),BUFF1(12,12),BUFF2(12,12)
      DIMENSION INU(12)
      INTEGER P,Q,R
C** COMPUTES THE INVERSE OF DELTA:BETA
      IF(IECR.GT.2)WRITE(6,100)
 100  FORMAT(1X,'SUBROUTINE MAIN7AC')
      IF(IECR.GT.9)WRITE(6,101)((BUFF(I,J),J=1,INUO),I=1,INUO)
 101  FORMAT(4(1X,'BUFF=',E10.4))
      DO 1 I=1,INUO
      DO 1 J=1,INUO
 1    BUFF1(I,J)=BUFF(I,J)
      IBUFF1=12
      IDGT=4
      WKAREA=12
      CALL LINV1F(BUFF1,INUO,IBUFF1,BUFF2,IDGT,WKAREA,IER)
      IF(IECR.GT.2)WRITE(6,99)IER
 99   FORMAT(1X,'ERROR CODE IN COMPUTATION OF D:B-1 IS ',I3)
      IF(IECR.GT.5)WRITE(6,102)((BUFF2(I,J),J=1,INUO),I=1,INUO)
 102  FORMAT(4(1X,'BUFF2=',E10.4))
C** (DELTA:BETA)-1 IS BUFF2
C** COMPUTATION OF BUFF2*XPI
      DO 2 I=1,INUO
      DO 2 J=1,INUO
      BUFF1(I,J)=0.E0
      DO 2 K=1,INUO
 2    BUFF1(I,J)=BUFF1(I,J)+BUFF2(I,K)*XPI(K,J)
      IF(IECR.GT.9)WRITE(6,103)((BUFF1(I,J),J=1,INUO),I=1,INUO)
 103  FORMAT(4(1X,'BUFF1=',E10.4))
C** COMPUTATION OF BUFF1*DELTA:BETA
      DO 3 I=1,INUO
      DO 3 J=1,INUO
      BUFF2(I,J)=0.E0
      DO 3 K=1,INUO
 3    BUFF2(I,J)=BUFF2(I,J)+BUFF1(I,K)*BUFF(K,J)
C** XPI* IS MADE OF THE LAST IRANK COLUMNS OF BUFF2
      IF(IECR.GT.5)WRITE(6,104)((BUFF2(I,J),J=1,INUO),I=1,INUO)
 104  FORMAT(4(1X,'PIO=',E10.4))
      NY=INUO-IRANK
      IF(IRANK.EQ.0) GO TO 5
```

```
      DO 4 I=1,IRANK                                                    MA701230
      DO 4 J=1,IRANK                                                    MA701240
      II=NN+I                                                           MA701250
      JJ=NN+J                                                           MA701260
 4    BUFF(I,J)=BUFF2(II,JJ)                                            MA701270
      IF(IECR.GT.5)WRITE(6,105)((BUFF(I,J),J=1,IRANK),I=1,IRANK)        MA701280
 105  FORMAT(4(1X,'XPI=',E10.4))                                        MA701290
C** COMPUTATION OF XPI EIGENVALUES                                      MA701300
      IJOB=0                                                            MA701310
      IBUFF=12                                                          MA701320
      IZ=1                                                              MA701330
      KWK=13                                                            MA701340
      CALL EIGRF(BUFF,IRANK,IBUFF,IJOB,WI,Z,IZ,KWK,IER)                 MA701350
      WRITE(6,106)IER                                                   MA701360
 106  FORMAT(1X,'ERROR CODE IN COMPUTATION OF XPII EIGENV. IS',I3)      MA701370
      WRITE(6,107)(WI(I),I=1,IRANK)                                     MA701380
 107  FORMAT(1X,'LAMBDAI ARE',2(E10.4,1X,E10.4))                        MA701390
      RETURN                                                            MA701400
 5    CONTINUE                                                          MA701410
      WRITE(6,108)                                                      MA701420
 108  FORMAT(1X,'THE SET OF LAMBDAI ASSOCIATED WITH THIS BI IS',/,1X,   MA701430
     1'THE NUL SET')                                                    MA701440
      RETURN                                                            MA701450
      END                                                               MA701460
      SUBROUTINE MAIN78                                                 MA700010
C** SELECTS THE BIS TO BE RETAINED                                      MA700020
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF                   MA700030
      COMMON/TRASH1/IZ                                                  MA700040
      COMMON/MANI0/IFLAG                                                MA700050
      DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12)         MA700060
      DIMENSION BUFF(12,12),IZ(12)                                      MA700070
      INTEGER P,Q,R                                                     MA700080
      IF(IECR.GT.2)WRITE(6,100)                                         MA700090
 100  FORMAT(1X,'SUBROUTINE MAIN78')                                    MA700100
 1    WRITE(6,101)                                                      MA700110
 101  FORMAT(1X,'IF YOU WANT TO CHANGE THE SET OF BI,TYPE 00',/,1X,     MA700120
     1'OTHERWISE,TYPE 11')                                              MA700130
      READ(5,1000) ISIGN                                                MA700140
 1000 FORMAT(I2)                                                        MA700150
      IF(ISIGN.NE.11) GO TO 2                                           MA700160
      WRITE(6,102)                                                      MA700170
 102  FORMAT(1X,'THE PROGRAM WILL NOT ALLOW YOU TO CHANGETHE SET',/,1X, MA700180
     1'OF BIS.IF YOU MADE A MISTAKE IN SELECTING THIS OPTION',/,1X,     MA700190
     1'TYPE 00.OTHERWISE,TYPE 11')                                      MA700200
      READ(5,1000)ISIGN1                                                MA700210
      IF(ISIGN1.EQ.0) GO TO 1                                           MA700220
      RETURN                                                            MA700230
 2    WRITE(6,103)                                                      MA700240
 103  FORMAT(1X,'THE PROGRAM WILL ASSUME YOU WANT TO SUBSTRACT',/,1X,   MA700250
     1'SOME BIS FROM THE SET OF EVENTS.IF YOU MADE A MISTAKE IN ',/,1X, MA700260
     1'SELECTING THIS OPTION,TYPE 00.OTHERWISE,TYPE 11')                MA700270
      READ(5,1000)ISIGN1                                                MA700280
      IF(ISIGN1.EQ.0) GO TO 1                                           MA700290
 3    WRITE(6,104)                                                      MA700300
 104  FORMAT(1X,'TYPE THE NUMBER OF EVENTS YOU WANT TO RETAIN')         MA700310
      READ(5,*)IRI                                                      MA700320
      WRITE(6,*)IRI                                                     MA700330
      WRITE(6,105)                                                      MA700340
 105  FORMAT(1X,'IF YOUMMADE A MISTAKE IN TYPING THE DATA TYPE00',/,1X, MA700350
     1'OTHERWISE,TYPE 11')                                              MA700360
      READ(5,1000)ISIGN                                                 MA700370
```

```
      IF(ISIGN.EQ.0)GO TO 3
 4    WRITE(6,106)
 106  FORMAT(1X,'TYPE THE INDEXES OF THE BIS YOU WANT TO RETAIN')
      READ(5,*)(IZ(I),I=1,IRI)
      WRITE(6,*)(IZ(I),I=1,IRI)
      WRITE(6,105)
      READ(5,1000)ISIGN
      IF(ISIGN.EQ.0) GO TO 4
C** FORMATION OF THE NEW MATRIX BI
      DO 5 I=1,N
      DO 5 J=1,IRI
      JJ=IZ(J)
 5    BUFF(I,J)=BI(I,JJ)
      DO 6 I=1,N
      DO 6 J=1,IRI
 6    BI(I,J)=BUFF(I,J)
      IF(IECR.GT.2)WRITE(6,107)((BI(I,J),J=1,IRI),I=1,N)
 107  FORMAT(4(1X,'BI=',E10.4))
      R=IRI
      IFLAG=1
      RETURN
      END
      SUBROUTINE MAIN8
C**  TEST FOR OUTPUT STATIONARITY
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
      DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12)
      DIMENSION BUFF(12,12)
      INTEGER P,Q,R
      IF(IECR.GT.2)WRITE(6,100)
 100  FORMAT(1X,'SUBROUTINE MAIN8')
 1    WRITE(6,101)
 101  FORMAT(1X,'WOULD YOU WISH TO TEST AN EVENT FOR OUTPUT STATIO',/,1X
     1'NARITY?IF YES,TYPE 11,IF NO,TYPE 00')
      READ(6,1000)ISIGN
 1000 FORMAT(I2)
      WRITE(6,1000)ISIGN
      IF(ISIGN.NE.11)GO TO 2
      WRITE(6,102)
 102  FORMAT(1X,'THE PROGRAM WILL PROCEED ON TESTING AN EVENT',/,1X,
     1'FOR OUTPUT STATIONARITY.IF YOU MADE A MISTAKE IN SELECTING',/,1X
     1,'THE OPTION,TYPE 00.OTHERWISE,TYPE 11')
      READ(6,1000)ISIGN
      WRITE(6,1000)ISIGN
      IF(ISIGN.EQ.0) GO TO 1
      CALL MAIN8A
      RETURN
 2    WRITE(6,103)
 103  FORMAT(1X,'THE PROGRAM WILL NOT TEST EVENTS FOR OUTPUT',/,1X,
     1'STATIONARITY.IF YOU MADE A MISTAKE IN SELECTING THIS',/,1X,
     1'OPTION,TYPE 00.OTHERWISE,TYPE 11')
      READ(5,1000) ISIGN
      WRITE(6,1000)ISIGN
      IF(ISIGN.EQ.0) GO TO 1
      RETURN
      END
      SUBROUTINE MAIN8A
C** READS THE EVENT BA FOR WHICH OUTPUT STATIONARITY IS TO BE TESTED
      COMMON/MANI8A/BA
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
      DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12)
      DIMENSION BUFF(12,12),BA(12)
```

```
      INTEGER P,Q,R
      IF(IECR.GT.2)WRITE(6,100)
100   FORMAT(1X,'SUBROUTINE MAIN8A')
1     WRITE(6,101)
101   FORMAT(1X,'WRITE BA(I),I=1,N,IN FREE FORMAT')
      READ(5,*)(BA(I),I=1,N)
      WRITE(6,*)(BA(I),I=1,N)
      WRITE(6,102)
102   FORMAT(1X,'IF YOU MADE A MISTAKE IN TYPING BA ,TYPE 00',/,1X,
     1'OTHERWISE ,TYPE 11')
      READ(5,1000)ISIGN
1000  FORMAT(I2)
      WRITE(6,1000)ISIGN
      IF(ISIGN.EQ.0) GO TO 1
      CALL MAIN8B
      RETURN
      END
```

```
      SUBROUTINE MAIN8B
C** COMPUTATION OF THE DETECTION SPACE OF BA
C** SUBROUTINE SIMILAR TO MAIN5
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
      COMMON/MANI8A/BA
      COMMON/TRASH2/BUFF1,BUFF2,BUFF3,XMD
      DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12)
      DIMENSION BUFF(12,12),BA(12),BUFF1(12,12),BUFF2(12,12)
      DIMENSION BUFF3(12,12),XMD(144,12)
      DIMENSION IROW(12),ICOL(12)
      INTEGER P,Q,R
C** COMPUTATION OF C*BA
      DO 1 I=1,P
      BUFF(I,1)=0.E0
      DO 1 J=1,N
 1    BUFF(I,1)=BUFF(I,1)+C(I,J)*BA(J)
      IF(IECR.GT.9)WRITE(6,100)(BUFF(I,1),I=1,P)
 100  FORMAT(4(1X,'CBA=',E10.4))
C** COMPUTATION OF DBA
      CC=0.
      DO 2 I=1,P
 2    CC=CC+BUFF(I,1)**2
      CC=1./CC
      IF(IECR.GT.6)WRITE(6,101)CC
 101  FORMAT(1X,'(CBAT*CBA)-1=',E10.4)
      DO 3 I=1,N
      BUFF(I,2)=0.E0
      DO 3 J=1,N
 3    BUFF(I,2)=BUFF(I,2)+A(I,J)*BA(J)
      IF(IECR.GT.8)WRITE(6,102)(BUFF(I,2),I=1,N)
 102  FORMAT(4(1X,'A*BA=',E10.4))
      DO 4 I=1,N
      DO 4 J=1,P
 4    BUFF1(I,J)=BUFF(I,2)*BUFF(J,1)*CC
      IF(IECR.GT.5)WRITE(6,103)((BUFF1(I,J),J=1,P),I=1,N)
 103  FORMAT(4(1X,'DBA=',E10.4))
C** COMPUTATION OF C'
      DO 5 I=1,P
      DO 5 J=1,P
 5    BUFF2(I,J)=-BUFF(I,1)*BUFF(J,1)*CC
      DO 6 J=1,P
 6    BUFF2(I,I)=1.+BUFF2(I,I)
      IF(IECR.GT.8)WRITE(6,104)((BUFF2(I,J),J=1,P),I=1,P)
 104  FORMAT(4(1X,'BUFF2=',E10.4))
      DO 7 I=1,P
      DO 7 J=1,N
      BUFF(I,J)=0.E0
      DO 7 K=1,P
 7    BUFF(I,J)=BUFF(I,J)+BUFF2(I,K)*C(K,J)
      IF(IECR.GT.5)WRITE(6,105)((BUFF(I,J),J=1,N),I=1,P)
 105  FORMAT(4(1X,'CPRIME=',E10.4))
C** COMPUTATION OF A-DBA*C
      DO 8 I=1,N
      DO 8 J=1,N
      BUFF2(I,J)=0.E0
      DO 8 K=1,P
 8    BUFF2(I,J)=BUFF2(I,J)+BUFF1(I,K)*C(K,J)
      IF(IECR.GT.8)WRITE(6,106)((BUFF2(I,J),J=1,N),I=1,N)
 106  FORMAT(4(1X,'DBA*C=',E10.4))
      DO 9 I=1,N
      DO 9 J=1,N
```

```
 9    BUFF1(I,J)=A(I,J)-BUFF2(I,J)
      IF(IECR.GT.5)WRITE(6,107)((BUFF1(I,J),J=1,N),I=1,N)
 107  FORMAT(4(1X,'A-DBA*C=',E10.4))
C** COMPUTATION OF XMD=MD'
      DO 10 I=1,P
      DO 10 J=1,N
      XMD(I,J)=BUFF(I,J)
 10   BUFF2(I,J)=BUFF(I,J)
      NN=N-1
      DO 14 L=1,NN
      NP1=L*P
      DO 11 I=1,P
      DO 11 J=1,N
      BUFF3(I,J)=0.E0
      DO 11 K=1,N
 11   BUFF3(I,J)=BUFF3(I,J)+BUFF2(I,K)*BUFF1(K,J)
      DO 12 I=1,P
      DO 12 J=1,N
 12   BUFF2(I,J)=BUFF3(I,J)
      DO 13 I=1,P
      DO 13 J=1,N
      II=I+L*P
 13   XMD(II,J)=BUFF3(I,J)
 14   CONTINUE
      NP=N*P
      IF(IECR.GT.3)WRITE(6,108)((XMD(I,J),J=1,N),I=1,NP)
 108  FORMAT(4(1X,'XMD=',E10.4))
C** ORTHOGONAL REDUCTION OF XMD
      IXMD=144
      EPSI1=0.0001
      DO 15 I=1,N
      DO 15 J=1,N
      BUFF(I,J)=0.E0
 15   BUFF(I,I)=1.E0
      CALL ORTRED(XMD,BUFF,NP,N,IXMD,EPSI1,IECR)
      IF(IECR.GT.1)WRITE(6,109)((BUFF(I,J),J=1,N),I=1,N)
 109  FORMAT(4(1X,'OMEGA=',E10.4))
C** COMPUTATION OF RK OMEGA AND LINEAR DEP
      DO 16 I=1,N
      DO 16 J=1,N
 16   BUFF1(I,J)=BUFF(I,J)
      IOMEG=12
      CALL MFGR(BUFF,IOMEG,N,N,IRANK,IROW,ICOL,EPS,IER)
      IF(IECR.GT.1)WRITE(6,110)IER
 110  FORMAT(1X,'IER=',I3)
      WRITE(6,111)IRANK
 111  FORMAT(1X,'RK OMEGA=',I3)
      WRITE(6,112)(IROW(I),ICOL(I),I=1,N)
 112  FORMAT(1X,'IROW=',I3,'ICOL=',I3)
      WRITE(6,113)((BUFF(I,J),J=1,N),I=1,N)
 113  FORMAT(4(1X,'DEP OF SA IS',E10.4))
      CALL MAIN8C(IRANK)
      RETURN
      END
      SUBROUTINE MAIN8C(IRANK)
C** COMPUTATION OF THE SUBSET OF EVENTS BI FROM WHICH BI IS
C** OUTPUT STATIONARY
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
      COMMON/MANI8A/BA
      COMMON/TRASH2/BUFF1
      DIMENSION A(12,12),B(12,12),BI(12,12),CBI(12,12),C(12,12)
```

```
      DIMENSION BUFF(12,12),BA(12),BUFF1(12,12)
      DIMENSION IROW(12),ICOL(12)
      INTEGER P,Q,R
      IF(IECR.GT.2)WRITE(6,100)
  100 FORMAT(1X,'SIBROUTINE MAIN8C')
C** COMPUTATION OF C*BA
      DO 2 I=1,P
      BUFF(I,1)=0.E0
      DO 2 J=1,N
 2    BUFF(I,1)=BUFF(I,1)+C(I,J)*BA(J)
C** COMPUTATION OF LINEAR DEPENDENCY OF CBA..CBI
C** IF R=11,THIS SUBROUTINE WONT WORK
      IF(R.NE.11)GO TO 3
      WRITE(6,101)
  101 FORMAT(1X,'R=11,YOU CANNOT TEST OUTPUT STATIONARITY WITH THIS',/,
     11X,'SUBROUTINE.CHECK SOURCE')
      STOP
 3    CONTINUE
      IRI=R+1
      DO 4 I=1,P
 4    CBI(I,IRI)=BUFF(I,1)
      DO 5 I=1,P
      DO 5 J=1,IRI
 5    BUFF(I,J)=CBI(I,J)
      IBUFF=12
      EPSI1=0.0001 E0
      CALL MFGR(BUFF,IBUFF,P,IRI,IRANK1,IROW,ICOL,EPSI1,IER)
      IF(IECR.GT.2)WRITE(6,102)IER
  102 FORMAT(1X,'ERROR CODE IN RANK OF BUFF IS',I3)
      IF(IECR.GT.2)WRITE(L,103)IRANK1
  103 FORMAT(1X,'RANK OF CBA ..CBI IS ',I3)
      IF(IECR.GT.2) WRITE(6,104)(IROW(I),ICOL(I),I=1,P)
  104 FORMAT(1X,'IROW=',E10.4,'ICOL=',E10.4)
      IF(IECR.GT.2)WRITE(6,105)((BUFF(I,J),J=1,IRI),I=1,P)
  105 FORMAT(4(1X,'BUFF=',E10.4))
      IF(IECR.GT.2) WRITE(6,106)
  106 FORMAT(1X,'SOLVING FOR LINEAR DEPENDENCY IN PRECEDING',/,1X,
     1'MATRIX,YOU KNOW THE RELATION BETWEEN CBA AND OTHER CBI',/,1X,
     1'CBA CAN BE MADE OUTPUT SEPERABLE WITH THE CBI FROM WHICH',/,1X,
     1'IT IS DEPENDENT.SEE RULES OF THUMB TO SEE IF IT IS ',/,1X,
     1'ADVISABLE TO DO SO')
      RETURN
      END
```

```
      SUBROUTINE MAIN9
C** COMPUTATION OF THE MATRIX TO (STEP 6 A)
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
      COMMON/MANI5/INU,INUS,INJO
      COMMON/MANI6A/ADSC
      COMMON/MANI6C/TO,IRANK,II
      COMMON/MANI6D/ROG
      COMMON/MANI3/IROW,ICOL
      COMMON/TRASH2/W,BUFF2
C** WHAT WAS BEFORE IN MANI6C,OMEGOG,IS OF NO INTEREST FROM NOW ON
      DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12)
      DIMENSION BUFF(12,12),W(12,12),IZ(12),TO(12,12),ADSC(12,12)
      DIMENSION INU(12),ROG(12,12),BUFF2(12,12),IROW(12),ICOL(12)
      INTEGER P,Q,R
      IF(IECR.GT.2)WRITE(6,100)
 100  FORMAT(1X,'SUBROUTINE MAIN9')
 1    CONTINUE
 2    WRITE(6,101)
 101  FORMAT(1X,'TYPE THE NUMBER OF RANKS OF CS WHICH HAVE A',/,1X,
     1'NON-ZERO AUXILIARY VECTOR IN THE ORTHOG REDUCTION OF XMD',/,
     11X,'STARTING WITH THE IDENTITY MATRIX')
C** ENTERING THE NUMBER OF WR
      READ(5,*)II
      WRITE(6,102)II
 102  FORMAT(I3)
      WRITE(6,103)
 103  FORMAT(1X,'IF DATA ARE CORRECTLY ENTERED,TYPE11.OTHERWISE,TYPE 0')
      READ(5,102)ISIGN
      IF(ISIGN.EQ.0) GO TO 2
      IF(II.EQ.0) GO TO 9
 3    CONTINUE
C** ENTERING THE WR
      DO 5 I=1,N
 4    WRITE(6,104)I,II
 104  FORMAT(1X,'TYPE WI(',I3,'),FOR I=1,',I3)
      READ(5,*)(W(I,J),J=1,II)
      WRITE(6,*)(W(I,J),J=1,II)
      WRITE(6,103)
      READ(5,102)ISIGN
      IF(ISIGN.EQ.0) GOTO4
 5    CONTINUE
      WRITE(6,105)((W(I,J),J=1,II),I=1,N)
 105  FORMAT(4(1X,'W=',E10.4))
      WRITE(6,103)
      READ(5,102)ISIGN
      IF(ISIGN.EQ.0) GO TO 3
C** ENTERING THE EXPONENT ASSOCIATED WITH THE WR
 6    WRITE(6,106)
 106  FORMAT(1X,'TYPE THE EXPONENTS QI ASSOCIATED WITH THE WI')
      READ(5,*)(IZ(I),I=1,II)
      WRITE(6,107)(IZ(I),I=1,II)
 107  FORMAT(12I3)
      WRITE(6,103)
      READ(5,102)ISIGN
      IF(ISIGN.EQ.0) GO TO 6
C** COMPUTATION OF ((A-DS*C)**IZ(I-1))*W(,I)
      CALL MAIN9A(ADSC,IZ,W,N,BUFF,II)
C** COMPUTATION OF TO
      ISUM=0
      DO 7 I=1,II
 7    ISUM=ISUM+IZ(I)
```

```
C** ISUM IS THE COLUMN DIMENSION OF W
      DO 8 I=1,N
      DO 8 J=1,ISUM
 8    BUFF2(I,J)=W(I,J)
      IW=12
      EPS=0.0001E0
      CALL MFGR(W,IW,N,ISUM,IRANK,IROW,ICOL,EPS,IER)
      WRITE(6,112) IER
 112  FORMAT(1X,'ERROR CODE IN COMPUTATION OF RK OF W IS',I3)
      IF(IECR.GT.1)WRITE(6,113)IRANK
 113  FORMAT(1X,'RK OF ADSCW IS',I3)
      IF(IECR.GT.1)WRITE(6,114)(IROW(I),ICOL(I),I=1,N)
 114  FORMAT(1X,'IROW=',I3,'ICOL=',I3)
      IF(IECR.GT.1)WRITE(6,111)((W(I,J),J=1,ISUM),I=1,N)
 111  FORMAT(4(1X,'DEP OF W IS',E10.4))
C** SELECTION OF IND COLUMNS OF W
      CALL MAIN9B(BUFF2,N,IRANK,IECR)
 9    CONTINUE
C** TEST TO CHECK THAT BUFF2 HAS N-NUS COLUMNS
      IF(IECR.GT.12)WRITE(6,108)INUO
 108  FORMAT(1X,'INUO=',I3)
      IF(IECR.GT.12)WRITE(6,109)INUS
 109  FORMAT(1X,'INUS=',I3)
      IF(II.EQ.0)IRANK=0
      IDIM=IRANK+INUS
      IF(IDIM.EQ.N) GO TO 14
      WRITE(6,110)
 110  FORMAT(1X,'TO DOES NOT HAVE N-INUS COLUMNS.YOU MADE',/,1X,
     1'A  MISTAKE,CHECK THE USER GUIDE')
      GO TO 1
 14   CONTINUE
      IF(INUO.EQ.0) GO TO 11
      DO 10 I=1,N
      DO 10 J=1,INUO
 10   TO(I,J)=ROG(I,J)
 11   CONTINUE
      IF(II.EQ.0) GO TO 13
      DO 12 I=1,N
      DO 12 J=1,IRANK
      JJ=INUO+J
 12   TO(I,JJ)=BUFF2(I,J)
 13   CONTINUE
      L=IRANK+INUO
      IF(IECR.GT.9)WRITE(6,115)((TO(I,J),J=1,L),I=1,N)
 115  FORMAT(4(1X,'TO=',E10.4))
      RETURN
      END
      SUBROUTINE MAIN9A(ADSC,IZ,W,N,BUFF,II)
C** COMPUTES ((A-DS*C)**IZ(I-1)*W(I)
      COMMON/MANI6E/BUFFI
C** IN MANI6E WAS TETA,OF NO INTEREST FROM NOW ON
C**WE SHALL USE BUFFI AS A WORKING AREA
      DIMENSION ADSC(12,12),IZ(12),W(12,12),BUFF(12,12),BUFFI(12,12)
      INTEGER P,Q,R
      IF(IECR.GT.2) WRITE(6,100)
 100  FORMAT(1X,'SUBROUTINE MAIN9A')
      IF(IECR.GT.12)WRITE(6,101)((ADSC(I,J),J=1,N),I=1,N)
 101  FORMAT(4(1X,'ADSC=',E10.4))
      INC=1
      DO 4 IND=1,II
      DO 1 I=1,N
```

```
      BUFF1(I,INC)=W(I,IND)
 1    BUFF(I,IND)=W(I,IND)
      INC=INC+1
      KK=IZ(IND)-1
      IF(KK.EQ.0) GO TO 4
      DO 4 K=1,KK
      DO 2 I=1,N
      W(I,IND)=0.E0
      DO 2 J=1,N
 2    W(I,IND)=W(I,IND)+ADSC(I,J)*BUFF(J,IND)
      DO 3 I=1,N
      BUFF1(I,INC)=W(I,IND)
 3    BUFF(I,IND)=W(I,IND)
      INC=INC+1
 4    CONTINUE
      DO 5 I=1,N
      DO 5 J=1,II
 5    BUFF(I,J)=W(I,J)
      DO 6 I=1,N
      DO 6 J=1,INC
 6    W(I,J)=BUFF1(I,J)
      IF(IECR.GT.5)WRITE(6,102)((W(I,J),J=1,INC),I=1,N)
 102  FORMAT(4(1X,'ADSC**IZ(I)*W=',E10.4))
C** BUF2=((A-DSC)**(IZ(1)-1)*W1,(A-DSC)**(IZ(2)-1)*W2...
C** W=(W1,(A-DSC)*W1,..,(A-DSC)**(IZ(U)-1)*W1,W2,..
      RETURN
      END
      SUBROUTINE MAIN9B(BUFF2,N,IRANK,IECR)
C** SELECTION OF INDEPENDENT COLUMNS OF W
      COMMON/MANI3/ICOL
      COMMON/MANI6E/BUFF
      DIMENSION BUFF2(12,12),BUFF(12,12),ICOL(12)
      INTEGER P,Q,R
      IF(IECR.GT.2)WRITE(6,100)
 100  FORMAT(1X,'SUBROUTINE MAIN9B')
 1    WRITE(6,101)
 101  FORMAT(1X,'TYPE THE INDEXES OF THE COLUMNS OF W YOU WANT',/,1X,
     1'TO KEEP.YOU MUST KEEP IRANK COLUMNS')
      READ(5,*)(ICOL(I),I=1,IRANK)
      WRITE(6,102)(ICOL(I),I=1,IRANK)
 102  FORMAT(15I3)
      WRITE(6,103)
 103  FORMAT(1X,'IF DATA ARE INCORRECTLY ENTERED,TYPE 00.OTHERWISE',/,1X
     1,'TYPE 11')
      READ(5,104)ISIGN
 104  FORMAT(I2)
      IF(ISIGN.EQ.0) GO TO 1
      DO 2 I=1,N
      DO 2 J=1,IRANK
      JJ=ICOL(J)
 2    BUFF(I,J)=BUFF2(I,JJ)
      DO 3 I=1,N
      DO 3 J=1,IRANK
 3    BUFF2(I,J)=BUFF(I,J)
      IF(IECR.GT.5)WRITE(6,105)((BUFF2(I,J),J=1,IRANK),I=1,N)
 105  FORMAT(4(1X,'BUFF2=',E10.4))
      RETURN
      END
```

```
      SUBROUTINE MAIN9C
C** COMPUTES T AND T-1 (STEP 63)
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
      COMMON/MANI6C/TO,IRANK,II
      COMMON/MANI5A/GG
      COMMON/MANI5/INU,INUS,INUO
      COMMON/MANI9C/T,TT,TM,TTM
      COMMON/TRASH2/BUFF1,BUFF2,BUFF3,BUFF4
      COMMON/MANI4/OMEGC,DS,CS
      DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12)
      DIMENSION BUFF(12,12),TO(12,12),GG(12,12),BUFF1(12,12)
      DIMENSION BUFF2(12,12),INU(12),OMEGC(12,12),CS(12,12)
      DIMENSION DS(12,12),TM(12,12),TTM(12,12)
      DIMENSION T(12,12),TT(12,12),BUFF4(12,12)
      INTEGER P,Q,R
      IF(IECR.GT.2)WRITE(6,100)
 100  FORMAT(1X,'SUBROUTINE MAIN9C')
C** COMPUTES GI,..,A**(INU(I)-1) FOR ALL GI
      CALL MA9CA(A,INU,N,IECR,GG,BUFF1,BUFF2,BUFF3,R)
C** BUFF1=(G1,A*G1,..A**(INU(1)-1)*G1,G2,A*G2
C** BUFF2=(A**(INU(1)-1)*G1,A**(INU(2)-1)*G2,..
      ISUM=0
      DO 1 I=1,R
 1    ISUM=ISUM+INU(I)
C** ISUM NUMBER OF COLUMNS OF BUFF1
      IF(IECR.GT.9)WRITE(6,101)ISUM
 101  FORMAT(1X,'ISUM=',I3)
      DO 2 J=1,ISUM
      DO 2 I=1,N
 2    T(I,J)=BUFF1(I,J)
      L=IRANK+INUO
      IF(L.EQ.0) GO TO 4
      DO 3 J=1,L
      DO 3 I=1,N
      JJ=ISUM+J
 3    T(I,JJ)=TO(I,J)
 4    CONTINUE
C** TEST TO CHECK THAT T HAS N COLUMNS
      INUA=INUS-INUO
      IF(ISUM.EQ.INUA) GO TO 5
      WRITE(6,102)
 102  FORMAT(1X,'BUFF1 DOES NOT HAVE INUA COLUMNS',/,1X,
     1'THERE IS A MISTAKE.CHECK THE USER GUIDE')
      STOP
 5    CONTINUE
      ITOT=ISUM+IRANK
      ITOT=ITOT+INUO
      IF(ITOT.EQ.N) GO TO 6
      WRITE(6,103)
 103  FORMAT(1X,'T DOES NOT HAVE N COLUMNS',/,1X,
     1'THERE IS A MISTAKE.CHECK THE USER GUIDE')
      STOP
 6    CONTINUE
      IF(IECR.GT.5)WRITE(6,104)((T(I,J),J=1,N),I=1,N)
 104  FORMAT(4(1X,'T=',E10.4))
C** COMPUTATION OF T-1=TT
      IT=12
      WKAREA=12
      DO 66 I=1,N
      DO 66 J=1,N
 66   BUFF4(I,J)=T(I,J)
```

```
      CALL LINV2F(T,N,IT,TT,IDGT,WKAREA,IER)
      DO 67 I=1,N
      DO 67 J=1,N
 67   T(I,J)=BUFF4(I,J)
      IF(IECR.GT.1)WRITE(6,105)IER,IDGT
 105  FORMAT(1X,'ERROR CODE IN COMPUTATION OF T-1',I3,'IDGT=',I3)
      IF(IECR.GT.5)WRITE(6,106)((TT(I,J),J=1,N),I=1,N)
 106  FORMAT(4(1X,'T-1=',E10.4))
      ITOT=II+R
      IF(ITOT.EQ.P) GO TO 7
      WRITE(6,109)
 109  FORMAT(1X,'TTM IS NOT P*P.CHECK THE USER GUIDE')
      STOP
 7    CONTINUE
C** COMPUTATION OF TM
      CALL MA9CB(C,P,N,BUFF2,R,CS,BUFF,II,BUFF1,BUFF3,TM,IECR)
C** COMPUTATION OF TM-1
      DO 77 I=1,P
      DO 77 J=1,P
 77   BUFF4(I,J)=TM(I,J)
      CALL LINV2F(TM,P,IT,TTM,IDGT,WKAREA,IER)
      DO 78 I=1,P
      DO 78 J=1,P
 78   TM(I,J)=BUFF4(I,J)
      IF(IECR.GT.1)WRITE(6,107)IER,IDGT
 107  FORMAT(1X,'ERROR CODE FOR TM-1 COMPUTATION',I3,'IDGT=',I3)
      IF(IECR.GT.5)WRITE(6,108)((TTM(I,J),J=1,P),I=1,P)
 108  FORMAT(4(1X,'TTM=',E10.4))
      RETURN
      END
      SUBROUTINE MA9CA(A,INU,N,IECR,GG,BUFF1,BUFF2,BUFF3,R)
C** COMPUTES GI,..,A**(INU(I)-1) FOR ALL GI
      DIMENSION A(12,12),INU(12),GG(12,12),BUFF1(12,12),BUFF2(12,12)
      DIMENSION BUFF3(12,12)
      INTEGER P,Q,R
      IF(IECR.GT.2)WRITE(6,100)
 100  FORMAT(1X,'SUBROUTINE MA9CA')
      IF(IECR.GT.12)WRITE(6,101)(INU(I),I=1,R)
      IND=1
      DO 6 II=1,R
      DO 1 I=1,N
      BUFF1(I,IND)=GG(I,II)
 1    BUFF2(I,II)=GG(I,II)
      IND=IND+1
      KK=INU(II)-1
      IF(KK.EQ.0) GO TO 5
      DO 5 K=1,KK
      DO 2 I=1,N
 2    BUFF3(I,II)=BUFF2(I,II)
      DO 3 I=1,N
      BUFF2(I,II)=0.E0
      DO 3 J=1,N
 3    BUFF2(I,II)=BUFF2(I,II)+A(I,J)*BUFF3(J,II)
      DO 4 I=1,N
 4    BUFF1(I,IND)=BUFF2(I,II)
      IND=IND+1
 5    CONTINUE
 6    CONTINUE
 101  FORMAT(8(1X,'INU=',I3))
      IF(IECR.GT.5)WRITE(6,102)((BUFF2(I,J),J=1,R),I=1,N)
 102  FORMAT(4(1X,'BUFF2=',E10.4))
```

```
      IND=IND-1
      IF(IECR.GT.5)WRITE(6,103)((BUFF1(I,J),J=1,IND),I=1,N)
 103  FORMAT(4(1X,'BUFF1=',E10.4))
C** BUFF1=(G1,A*G1,..,A**(INU(J)-1)*G1,G2,A*G2
C** BUFF2=(A**(INU(U)-1)*G1,A**(INU(I)-1)*G2,
      RETURN
      END
      SUBROUTINE MA9CB(C,P,N,BUFF2,R,CS,BUFF,II,BUFF1,BUFF3,TM,IECR)
C** COMPUTES TM
      DIMENSION C(12,12),BUFF2(12,12),CS(12,12),BUFF(12,12),BUFF1(12,12)
      DIMENSION BUFF3(12,12),TM(12,12)
      INTEGER P,Q,R
      IF(IECR.GT.2)WRITE(6,100)
 100  FORMAT(1X,'SUBROUTINE MA9CB')
C** BUFF2=(A**(INU(U)-1)*G1,A**(INU(I)-1)*G2,..
C** BUFF=(A-DSC)**(IZ(U)-1)*W1,(A-DSC)**(IZ(I)-1)*W2,..
      IF(IECR.GT.12)WRITE(6,101)((BUFF2(I,J),J=1,R),I=1,N)
 101  FORMAT(4(1X,'BUFF2=',F10.4))
      IF(IECR.GT.12)WRITE(6,102)((BUFF(I,J),J=1,II),I=1,N)
 102  FORMAT(4(1X,'BUFF=',E10.4))
      IF(IECR.GT.12)WRITE(6,103)((CS(I,J),J=1,N),I=1,P)
  103 FORMAT(4(1X,'CS=',E10.4))
C** T=C*BUFF2:CS*BUFF
      DO 1 I=1,P
      DO 1 J=1,R
      TM(I,J)=0.E0
      DO 1 K=1,N
 1    TM(I,J)=TM(I,J)+C(I,K)*BUFF2(K,J)
      IF(II.EQ.0) GO TO 4
      DO 2 I=1,P
      DO 2 J=1,II
      BUFF1(I,J)=0.E0
      DO 2 K=1,N
 2    BUFF1(I,J)=BUFF1(I,J)+CS(I,K)*BUFF(K,J)
      IF(IECR.GT.9)WRITE(6,104)((TM(I,J),J=1,R),I=1,P)
 104  FORMAT(4(1X,'TM1=',E10.4))
      IF(IECR.GT.9)WRITE(6,105)((BUFF1(I,J),J=1,II),I=1,P)
 105  FORMAT(4(1X,'TM2=',E10.4))
      DO 3 I=1,P
      DO 3 J=1,II
      JJ=II+J
 3    TM(I,JJ)=BUFF1(I,J)
 4    CONTINUE
      JJ=II+R
      IF(IECR.GT.5)WRITE(6,106)((TM(I,J),J=1,JJ),I=1,P)
 106  FORMAT(4(1X,'TM=',E10.4))
```

```
      RETURN
      END
      SUBROUTINE MAIN9D
C** COMPUTES AH,BH,CH,DH,(STEP 6D)
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
      COMMON/MANI9C/T,TT,TM,TTM
      COMMON/MANI4/AH,BH,CH
C** IN MANI4 WERE OMEGC,DS,CS,OF NO INTEREST FROM NOW ON
      DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12)
      DIMENSION BUFF(12,12),T(12,12),TT(12,12),TM(12,12),TTM(12,12)
      DIMENSION AH(12,12),BH(12,12),CH(12,12)
      INTEGER P,Q,R
C** COMPUTATION OF AH
      IF(IECR.GT.2)WRITE(6,100)
  100 FORMAT(1X,'SUBROUTINE MAIN9D')
      IF(IECR.GT.12)WRITE(6,101)((T(I,J),J=1,N),I=1,N)
  101 FORMAT(4(1X,'T=',E10.4))
      IF(IECR.GT.12)WRITE(6,102)((TT(I,J),J=1,N),I=1,N)
  102 FORMAT(4(1X,'TT=',E10.4))
      IF(IECR.GT.12)WRITE(6,103)((A(I,J),J=1,N),I=1,N)
  103 FORMAT(4(1X,'A=',E10.4))
      DO 1 I=1,N
      DO 1 J=1,N
      BUFF(I,J)=0.E0
      DO 1 K=1,N
  1   BUFF(I,J)=BUFF(I,J)+A(I,K)*T(K,J)
      DO 2 I=1,N
      DO 2 J=1,N
      AH(I,J)=0.E0
      DO 2 K=1,N
  2   AH(I,J)=AH(I,J)+TT(I,K)*BUFF(K,J)
      IF(IECR.GT.0)WRITE(6,104)
  104 FORMAT(1X,'A HAT=')
      IF(IECR.GT.0)WRITE(6,*)((AH(I,J),J=1,N),I=1,N)
C** COMPUTATION OF BH
      IF(IECR.GT.12)WRITE(6,105)((B(I,J),J=1,Q),I=1,N)
  105 FORMAT(4(1X,'B=',E10.4))
      DO 3 I=1,N
      DO 3 J=1,Q
      BH(I,J)=0.E0
      DO 3 K=1,N
  3   BH(I,J)=BH(I,J)+TT(I,K)*B(K,J)
      IQ=Q
      IF(IECR.GT.0)WRITE(6,106)
  106 FORMAT(1X,'B HAT=')
      IF(IECR.GT.0)WRITE(6,*)((BH(I,J),J=1,IQ),I=1,N)
C** COMPUTATION OF CH
      IF(IECR.GT.12)WRITE(6,107)((TTM(I,J),J=1,P),I=1,N)
  107 FORMAT(4(1X,'TTM=',E10.4))
      IF(IECR.GT.12)WRITE(6,108)((C(I,J),J=1,N),I=1,P)
  108 FORMAT(4(1X,'C=',E10.4))
      DO 4 I=1,P
      DO 4 J=1,N
      BUFF(I,J)=0.E0
      DO 4 K=1,N
  4   BUFF(I,J)=BUFF(I,J)+C(I,K)*T(K,J)
      DO 5 I=1,P
      DO 5 J=1,N
      CH(I,J)=0.E0
      DO 5 K=1,N
  5   CH(I,J)=CH(I,J)+TTM(I,K)*BUFF(K,J)
```

```
      IF(IECR.GT.0)WRITE(6,109)
 109  FORMAT(1X,'C HAT=')
      IP=P
      IF(IECR.GT.0)WRITE(6,*)((CH(I,J),J=1,N),I=1,IP)
      CALL MAIN9E
      RETURN
      END
      SUBROUTINE MAIN9E
C** COMPUTES DFR
      COMMON/MANI4/AH,BH,CH
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
      COMMON/MANI5/INU,INUS,INUO
      COMMON/MANI5A/DFR
C** IN MANI5A WAS GG,OF NO INTEREST FROM NOW ON
      DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12)
      DIMENSION BUFF(12,12),INU(12),DFR(12,12),AH(12,12)
      DIMENSION BH(12,12),CH(12,12)
      INTEGER P,Q,R
      IF(IECR.GT.2)WRITE(6,100)
 100  FORMAT(1X,'SUBROUTINE MAIN9E')
      IF(IECR.GT.12)WRITE(6,101)((AH(I,J),J=1,N),I=1,N)
 101  FORMAT(4(1X,'AH=',E10.4))
      IF(IECR.GT.12)WRITE(6,102)(INU(I),I=1,R)
 102  FORMAT(1X,8('INU=',I3))
      DO 5 JJ=1,R
      J=0
      DO 1 K=1,JJ
 1    J=J+INU(K)
      DO 2 I=1,N
 2    DFR(I,JJ)=AH(I,J)
      JJJ=JJ-1
      J=0
      DO 3 K=1,JJJ
 3    J=J+INU(K)
      IF(JJJ.EQ.0) J=0
      JK=J+INU(JJ)
      J=J+1
      DO 4 I=J,JK
 4    DFR(I,JJ)=0.E0
 5    CONTINUE
      ISUM=0
      DO 6 J=1,R
 6    ISUM=ISUM+INU(J)
      ISUM=ISUM+1
      DO 7 J=ISUM,P
      DO 7 I=1,N
 7    DFR(I,J)=0.E0
      IF(IECR.GT.9)WRITE(6,103)
 103  FORMAT(1X,'DFR=')
      IP=P
      IF(IECR.GT.9)WRITE(6,*)((DFR(I,J),J=1,IP),I=1,N)
      CALL MAIN9F
      RETURN
      END
      SUBROUTINE MAIN9F
C** COMPUTES THE POL COEFF
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
      COMMON/MANI5/INU,INUS,INUO
      COMMON/MANI6A/XP
      COMMON/TRASH2/BUFF1
C** IN MANI6A WAS ADSC OF NO INTEREST FROM NOW ON
```

```
      DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12)
      DIMENSION BUFF(12,12),INJ(12),XP(12,12),BUFF1(12)
      INTEGER P,Q,R
C** READS THE EIGENVALUES DESIRED FOR EACH DETECTION SPACE
      IF(IECR.GT.2)WRITE(6,100)
  100 FORMAT(1X,'SUBROUTINE MAIN9F')
      DO 8 II=1,R
      NI=INU(II)
 1    WRITE(6,101)NI,II
  101 FORMAT(1X,'TYPE THE DESIRED',I3,'EIGENVALUES FOR THE',/,1X,
     1I3,'DETECTION SPACE.FREE FORMAT')
      READ(6,*)(BUFF1(I),I=1,NI)
      WRITE(6,*)(BUFF1(I),I=1,NI)
      WRITE(6,102)
  102 FORMAT(1X,'IF DATA ARE CORRECTLY ENTERED,TYPE 11.OTHERWIDE,TYPE0')
      READ(5,103)ISIGN
  103 FORMAT(I2)
      IF(ISIGN.EQ.0) GO TO 1
C** COMPUTES THE POL COEFF
      IF(NI.LE.4) GO TO 2
      WRITE(6,104)
  104 FORMAT(1X,'THE PROGRAM DOES NOT ALLOW MULTIPLICITY GREATER THAN 4'
     1,/,1X,'SEE SOURCE,SUBROUTINE MAIN9F')
 2    CONTINUE
      INI=NI+1
      DO 3 I=INI,4
 3    BUFF1(I)=0.E0
      XP(1,II)=0.E0
      DO 4 I=1,INI
 4    XP(1,II)=XP(1,II)+BUFF1(I)
      XP(1,II)=-XP(1,II)
      XP(2,II)=0.E0
      DO 5 I=2,4
 5    XP(2,II)=XP(2,II)+BUFF1(1)*BUFF1(I)
      DO 6 I=3,4
 6    XP(2,II)=XP(2,II)+BUFF1(2)*BUFF1(I)
      XP(2,II)=XP(2,II)+BUFF1(3)*BUFF1(4)
      X=BUFF1(1)*BUFF1(2)
      XXA=X*BUFF1(3)
      XXB=X*BUFF1(4)
      Y=BUFF1(3)*BUFF1(4)
      YYA=Y*BUFF1(1)
      YYB=Y*BUFF1(2)
      XP(3,II)=XXA+XXB
      XP(3,II)=XP(3,II)+YYA
      XP(3,II)=XP(3,II)+YYB
      XP(3,II)=-XP(3,II)
      XP(4,II)=1.E0
      DO 7 I=1,4
 7    XP(4,II)=XP(4,II)*BUFF1(I)
      DO 9 I=1,NI
      JJ=2*NI+1-I
      XP(JJ,II)=XP(I,II)
 9     CONTINUE
      DO 10 I=1,NI
      JJ=I+NI
      XP(I,II)=XP(JJ,II)
 10   CONTINUE
 8     CONTINUE
      IF(IECR.GT.5)WRITE(6,105)
  105 FORMAT(1X,'POL COEF ARE')
```

```
      IR=R
      IF(IECR.GT.5)WRITE(6,*)((XP(I,J),J=1,IR),I=1,N)
      CALL MAIN9G
      RETURN
      END
```

```
      SUBROUTINE MAIN9G
C** COMPUTES DPSI
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
      COMMON/MANI4/AH,BH,CH
      COMMON/MANI6A/XP
      COMMON/MANI6B/DPSI
      COMMON/MANI5/INU,INUS,INJO
C** IN MANI6B WERE XMO,TETA,OF NO INTEREST FROM NOW ON
      DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12)
      DIMENSION BUFF(12,12),AH(12,12),BH(12,12),CH(12,12)
      DIMENSION XP(12,12),DPSI(12,12),INU(12)
      INTEGER P,Q,R
      IF(IECR.GT.2)WRITE(6,100)
  100 FORMAT(1X,'SUBROUTINE MAIN9G')
      DO 4 JJ=1,R
      DO 1 I=1,N
    1 DPSI(I,JJ)=0.E0
      IA=0
      JJJ=JJ-1
      DO 2 I=1,JJJ
    2 IA=IA+INU(I)
      IA=IA+1
      IF(JJJ.EQ.0)IA=1
C** IA IS THE ROW WHERE PI BEGINS(STEP 6E)
      IB=IA+INU(JJ)
      IB=IB-1
      DO 3 I=IA,IB
      II=I-IA
      II=II+1
    3 DPSI(I,JJ)=XP(II,JJ)+AH(I,IB)
    4 CONTINUE
      JJ=R+1
      DO 5 J=JJ,P
      DO 5 I=1,N
    5 DPSI(I,J)=0.E0
      IF(IECR.GT.5)WRITE(6,101)
  101 FORMAT(1X,'DPSI=')
      IP=P
      IF(IECR.GT.5)WRITE(6,*)((DPSI(I,J),J=1,IP),I=1,N)
      CALL MAIN9H
      RETURN
      END
      SUBROUTINE MAIN9H
C** COMPUTES DH=DPSI+DFR
C** COMPUTES D=T*DH*TTM
      COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,BUFF
      COMMON/MANI9C/T,TT,TM,TTM
      COMMON/MANI5A/DFR
      COMMON/MANI4/AH,BH,CH
      COMMON/MANI6B/DPSI
      COMMON/TRASH2/D,DH,BUFF1
      DIMENSION A(12,12),B(12,12),BI(12,12),C(12,12),CBI(12,12)
      DIMENSION BUFF(12,12),T(12,12),TT(12,12),TM(12,12),TTM(12,12)
      DIMENSION DFR(12,12),DPSI(12,12),D(12,12),DH(12,12),BUFF1(12,12)
      DIMENSION AH(12,12),BH(12,12),CH(12,12)
      INTEGER P,Q,R
      IF(IECR.GT.2)WRITE(6,100)
  100 FORMAT(1X,'SUBROUTINE MAIN9H')
      DO 1 I=1,N
      DO 1 J=1,P
    1 DH(I,J)=DFR(I,J)+DPSI(I,J)
```

```
      IF(IECR.GT.0)WRITE(6,101)                                         MAI00620
 101  FORMAT(1X,'D HAT=')                                               MAI00630
      IP=P                                                              MAI00640
      IF(IECR.GT.0)WRITE(6,*)((DH(I,J),J=1,IP),I=1,N)                   MAI00650
C** CHECK :COMPUTES AH-DH*CH                                            MAI00660
      IF(IECR.LE.3) GO TO 4                                             MAI00670
      DO 2 I=1,N                                                        MAI00680
      DO 2 J=1,N                                                        MAI00690
      BUFF(I,J)=0.E0                                                    MAI00700
      DO 2 K=1,P                                                        MAI00710
 2    BUFF(I,J)=BUFF(I,J)+DH(I,K)*CH(K,J)                               MAI00720
      DO 3 I=1,N                                                        MAI00730
      DO 3 J=1,N                                                        MAI00740
    3 BUFF1(I,J)=AH(I,J)-BUFF(I,J)                                      MAI00750
      WRITE(6,102)                                                      MAI00760
 102  FORMAT(1X,'AH-DH*CH')                                             MAI00770
      WRITE(6,*)((BUFF1(I,J),J=1,N),I=1,N)                              MAI00780
 4    CONTINUE                                                          MAI00790
C** COMPUTES D=T*DH*TTM                                                 MAI00800
      DO 5 I=1,N                                                        MAI00810
      DO 5 J=1,P                                                        MAI00820
      BUFF(I,J)=0.E0                                                    MAI00830
      DO 5 K=1,P                                                        MAI00840
 5    BUFF(I,J)=BUFF(I,J)+DH(I,K)*TTM(K,J)                              MAI00850
      DO 6 I=1,N                                                        MAI00860
      DO 6 J=1,P                                                        MAI00870
      D(I,J)=0.E0                                                       MAI00880
      DO 6 K=1,N                                                        MAI00890
  6   D(I,J)=D(I,J)+T(I,K)*BUFF(K,J)                                    MAI00900
      WRITE(6,103)                                                      MAI00910
 103  FORMAT(1X,'D=')                                                   MAI00920
      WRITE(6,*)((D(I,J),J=1,IP),I=1,N)                                 MAI00930
C**CHECK:COMPUTES A-D*C                                                 MAI00940
      IF(IECR.LE.3)RETURN                                               MAI00950
      DO 7 I=1,N                                                        MAI00960
      DO 7 J=1,N                                                        MAI00970
      BUFF(I,J)=0.E0                                                    MAI00980
      DO 7 K=1,P                                                        MAI00990
 7    BUFF(I,J)=BUFF(I,J)+D(I,K)*C(K,J)                                 MAI01000
      DO 8 I=1,N                                                        MAI01010
      DO 8 J=1,N                                                        MAI01020
 8    BUFF1(I,J)=A(I,J)-BUFF(I,J)                                       MAI01030
      WRITE(6,104)                                                      MAI01040
 104  FORMAT(1X,'A-D*C')                                                MAI01050
      WRITE(6,*)((BUFF1(I,J),J=1,N),I=1,N)                              MAI01060
      RETURN                                                            MAI01070
      END                                                               MAI01080
      SUBROUTINE MFGR(A,IA,M,N,IRANK,IROW,ICOL,EPS,IER)                 RAN00010
      DIMENSION A(1),IROW(1),ICOL(1)                                    RAN00020
      DOUBLE PRECISION SAVE,HOLD,WORK,F1,F2                             RAN00030
      INDER=IER                                                         RAN00040
      IER=0                                                             RAN00050
      IF(M) 2,2,1                                                       RAN00060
 1    IF(N) 2,2,3                                                       RAN00070
 2    IER=1000                                                          RAN00080
      GO TO 44                                                          RAN00090
 3    IF(IA)4,2,5                                                       RAN00100
 4    IRA=-IA                                                           RAN00110
      ICA=1                                                             RAN00120
      GO TO 6                                                           RAN00130
 5    IRA=1                                                             RAN00140
```

```
      ICA=IA                                                            RAN00150
6     PIV=0.E0                                                          RAN00160
      IC=1                                                              RAN00170
      DO 9 J=1,N                                                        RAN00180
      ICOL(J)=J                                                         RAN00190
      IR=IC                                                             RAN00200
      DO 8 I=1,M                                                        RAN00210
      SEEK=A(IR)                                                        RAN00220
      IF(ABS(SEEK)-ABS(PIV))8,8,7                                       RAN00230
7     PIV=SEEK                                                          RAN00240
      NC=J                                                              RAN00250
      NR=I                                                              RAN00260
8     IR=IR+IRA                                                         RAN00270
9     IC=IC+ICA                                                         RAN00280
      DO 10 I=1,M                                                       RAN00290
10    IROW(I)=I                                                         RAN00300
      TOL=ABS(EPS*PIV)                                                  RAN00310
      IRANK=0                                                           RAN00320
      IDA=IRA+ICA                                                       RAN00330
      JD=1                                                              RAN00340
     JR=1                                                               RAN00350
     JC=1                                                               RAN00360
      LIM=M                                                             RAN00370
      IF(M-N) 12,12,11                                                  RAN00380
11    LIM=N                                                             RAN00390
12    DO 27 J=1,LIM                                                     RAN00400
      IF(ABS(PIV)-TOL) 28,28,13                                         RAN00410
13    IRANK=J                                                           RAN00420
      IF(NR-IRANK)16,16,14                                              RAN00430
14    INR=(NR-1)*IRA+1                                                  RAN00440
      IM=JR                                                             RAN00450
      MR=INR                                                            RAN00460
      DO 15 I=1,N                                                       RAN00470
      SEEK=A(IM)                                                        RAN00480
      A(IM)=A(MR)                                                       RAN00490
      A(MR)=SEEK                                                        RAN00500
      MR=MR+ICA                                                         RAN00510
15    IM=IM+ICA                                                         RAN00520
      IN=IROW(NR)                                                       RAN00530
      IROW(NR)=IROW(IRANK)                                              RAN00540
      IROW(IRANK)=IN                                                    RAN00550
16    IF(NC-IRANK)19,19,17                                              RAN00560
17    IM=JC                                                             RAN00570
      INC=ICA*(NC-1)+1                                                  RAN00580
      MC=INC                                                            RAN00590
      DO 18 I=1,M                                                       RAN00600
      SEEK=A(IM)                                                        RAN00610
      A(IM)=A(MC)                                                       RAN00620
      A(MC)=SEEK                                                        RAN00630
      IM=IM+IRA                                                         RAN00640
18    MC=MC+IRA                                                         RAN00650
      IN=ICOL(NC)                                                       RAN00660
      ICOL(NC)=ICOL(IRANK)                                              RAN00670
      ICOL(IRANK)=IN                                                    RAN00680
19    SAVE=PIV                                                          RAN00690
      PIV=0.E0                                                          RAN00700
      J1=J+1                                                            RAN00710
      IR=JD                                                             RAN00720
      IC=JD+ICA                                                         RAN00730
      I=J                                                               RAN00740
20    IF(I-M) 21,26,26                                                  RAN00750
```

```
21     I=I+1                                    RAN00760
       IR=IR+IRA                                RAN00770
       HOLD=A(IR)                               RAN00780
       HOLD=HOLD/SAVE                           RAN00790
       A(IR)=HOLD                               RAN00800
       KRC=IR+ICA                               RAN00810
       KC=IC                                    RAN00820
       K=J                                      RAN00830
22     IF(K-N)23,20,20                          RAN00840
23     K=K+1                                    RAN00850
       F1=A(KRC)                                RAN00860
       F2=A(KC)                                 RAN00870
       WORK=F1-F2*HOLD                          RAN00880
       A(KRC)=WORK                              RAN00890
       IF(DABS(WORK)-ABS(PIV))25,25,24          RAN00900
24     PIV=WORK                                 RAN00910
       NR=I                                     RAN00920
       NC=K                                     RAN00930
25     KRC=KRC+ICA                              RAN00940
       KC=KC+ICA                                RAN00950
       GO TO 22                                 RAN00960
26     JC=JC+ICA                                RAN00970
       JR=JR+IRA                                RAN00980
27     JD=JD+IDA                                RAN00990
28     IF(IRANK-M)29,34,34                      RAN01000
29     J=IRANK-1                                RAN01010
       IR1=IRANK+1                              RAN01020
       JC=JD-IDA                                RAN01030
       JR=JD-ICA                                RAN01040
30     IF(J)34,34,31                            RAN01050
31     IR=JR                                    RAN01060
       IC=JC-ICA                                RAN01070
       IJ=JC+IRA                                RAN01080
       J1=J+1                                   RAN01090
       DO 33 I=IR1,M                            RAN01100
       KR=IR                                    RAN01110
       KC=JC                                    RAN01120
       WORK=0.E0                                RAN01130
       DO 32 K=J1,IRANK                         RAN01140
       WORK=WORK+A(KR)*A(KC)                    RAN01150
       KR=KR-ICA                                RAN01160
32     KC=KC-IRA                                RAN01170
       IR=IR+IRA                                RAN01180
       A(IJ)=A(IJ)-WORK                         RAN01190
33     IJ=IJ+IRA                                RAN01200
       J=J-1                                    RAN01210
       GO TO 30                                 RAN01220
  34   IF(IRANK-N) 35,43,43                     RAN01230
35     J=IRANK                                  RAN01240
       IR1=IRANK+1                              RAN01250
       JR=JD-IDA                                RAN01260
       JC=JD-IRA                                RAN01270
  36   IF(J)43,43,37                            RAN01280
37     IC=JC                                    RAN01290
       JI=JR+ICA                                RAN01300
       JD=JD-IDA                                RAN01310
       DO 42 I=IR1,N                            RAN01320
       KR=JR                                    RAN01330
       KC=IC                                    RAN01340
       WORK=0.E0                                RAN01350
       K=J                                      RAN01360
```

```
39    IF(K-IRANK)40,41,41                                    RAN01370
40    F1=A(KR)                                               RAN01380
      F2=A(KC)                                               RAN01390
      WORK=WORK+F1*F2                                        RAN01400
      KC=KC-ICA                                              RAN01410
      KR=KR-IRA                                              RAN01420
      K=K+1                                                  RAN01430
      GO TO 39                                               RAN01440
41    IC=IC+ICA                                              RAN01450
      A(JI)=-(A(JI)+WORK)/A(JJ)                              RAN01460
42    JI=JI+ICA                                              RAN01470
      JR=JR-IRA                                              RAN01480
      J=J-1                                                  RAN01490
      GO TO 36                                               RAN01500
43    CONTINUE                                               RAN01510
44    CONTINUE                                               RAN01520
45    CONTINUE                                               RAN01530
46    RETURN                                                 RAN01540
      END                                                    RAN01550
```

```
      SUBROUTINE ORTRED(DEP1,ARR,NPRIME,NREAL,IDEP1,EPSI1,IECR)
      DIMENSION DEP1(1),DEP(144,12),ARR(12,12),WRED(12)
      DIMENSION WW(12,12)
      COMMON/TRASH2/WW
      N=IDEP1*NREAL
      IF(IECR.GT.12)WRITE(6,1004)(DEP1(I),I=1,N)
 1004 FORMAT(5(1X,'DEP1=',E10.4))
      DO 10 K=1,N
      II=(K-1)/IDEP1
      I=II+1
      J=K-II*IDEP1
      IF(IECR.GT.12)WRITE(6,1005)I,J
 1005 FORMAT(1X,'I=',I3,'J=',I3)
 10   DEP(I,J)=DEP1(K)
      IF(IECR.GT.8)WRITE(6,1003)((DEP(I,J),J=1,NREAL),I=1,NPRIME)
 1003 FORMAT(4(1X,'DEP=',E10.4))
      IF(IECR.GT.0)WRITE(6,100)
 100  FORMAT(1X,'COMPUTATION OF ORTHOGONAL REDUCTION,WITH A
     1PRECISION OF EPSI1')
      INDEX=0
      DO 9 J=1,NPRIME
      ISIGN=0
      DO 1 K=1,NREAL
 1    WRED(K)=0.E0
      DO 3 K=1,NREAL
      DO 2 L=1,NREAL
 2    WRED(K)=WRED(K)+ARR(K,L)*DEP(J,L)
      IF(ABS(WRED(K)).GT.EPSI1)ISIGN=1
 3    CONTINUE
      IF(ISIGN.EQ.1) GO TO 4
      IF(IECR.GT.0)WRITE(6,101)J
 101  FORMAT(1X,'VECTOR DEP(',I3,',.) IS ALREADY ORTHOGONAL TO
     1 ARR AT THIS POINT')
      GO TO 8
 4    WVJ=0.E0
      IF(IECR.GT.3)WRITE(6,1000)(WRED(K),K=1,NREAL)
 1000 FORMAT(4(1X,'WRED=',E10.4))
      DO 5 I=1,NREAL
 5    WVJ=WVJ+WRED(I)*DEP(J,I)
      IF(IECR.GT.3) WRITE(6,1001)WVJ
      DO 6 I=1,NREAL
      DO 6 L=1,NREAL
 6    WW(I,L)=WRED(I)*WRED(L)/WVJ
      DO 12 I=1,NREAL
      DO 12 L=1,NREAL
      IF(ABS(WW(I,L)).LE.EPSI1)WW(I,L)=0.
 12   CONTINUE
      IF(IECR.GT.5)WRITE(6.1002)((WW(I,L),L=1,NREAL),I=1,NREAL)
      INDEX=INDEX+1
      DO 7 I=1,NREAL
      DO 7 L=1,NREAL
 7    ARR(I,L)=ARR(I,L)-WW(I,L)
 8    CONTINUE
      IF(IECR.GT.0)WRITE(6,102)INDEX
 102  FORMAT(1X,
     1I3,'ALTERATIONS TO ARR HAVE BEEN PERFORMED')
 9    CONTINUE
 1001 FORMAT(1X,'WVJ=',E10.4)
 1002 FORMAT(1X,5 E10.4)
      DO 11 I=1,NREAL
      DO 11 J=1,NREAL
```

```
      IF(ABS(ARR(I,J)).LE.EPSI1) ARR(I,J)=0.
 11   CONTINUE
      RETURN
      END
```

APPENDIX B

LISTING OF LONGITUDINAL GUIDEWAY VEHICLE SIMULATION

```
:READ  TEST      FORTRAN  A1 DISCA   6/22/78  17:39
C
C
C          SIMULATION PROGRAM - LCV DETECTION FILTER  STUDY
C
C
        IMPLICIT REAL*4(A-H,O-Z)
      DIMENSION C(3,7),D(3,5)
      DIMENSION CT(9,1),VCM(9,1),CHT(9,1),VWIND(9,1)
      DIMENSION AD1(7,1),AD2(7,1)
     >,X(7,1),Y(3,1),XD(7)
     >,XMES(6),YMES(2),XINI(6),EPS1(2)
     >,CX(3,1),DJ(3,1)
      COMMON/A1/ A(7,7),B(7,5),U(5,1)
      REAL*8 DT,TM
      DIMENSION WW(7,9),CC(24)
C
C          ALL UNITS MUST BE INPUT IN FPS STANDARD SYSTEM
C                   FT - LB - SEC
C
C          SPECIFY SYSTEM CONSTANTS
C
      EXTERNAL FCT
      NN=7
      TOL=0.1
      IND=1
      WRITE(6,102)
      READ(5,201)IPI
      AMVEST=350.0E0
      GN=32.2E0
      TFAIL=10000.
      ICOMP=0
      AMV=373.0E0
      AK1=1500.0E0
      AK2=1000.0E0
      AK3=12150.0E0
C     AK4=1.0E0
      AK4=1.2E0
      A1=12150.0E0
      A2=-55.5E0
      A3=-1466.5E0
      A4=-12150.0E0
      CV=100.0E0
      AJMAX=6.44E0
      AMAX=8.05E0
      IFL=0
      IFLAG3=0
      VW=0.0E0
      IDUMMY=0
C     WT=7.333333E0
C
C  SEED FOR NOISE
C
      ISEED=314100
      AEROC=0.03E0
      BW=AEROC
      DN=1.0E0
C
C          INITIAL CONDITIONS
C
      WRITE(6,103)
```

```
      READ (5,202)XX
      WRITE(6,104)
      READ(5,202)VC
      WRITE(6,105)
      READ(5,201)ICHV
      IF(ICHV.EQ.0)GO TO 12
      WRITE(6,106)
      DO 12 J=1,ICHV
      READ(5,*)CT(J,1),VCM(J,1)
12    CONTINUE
      WRITE(6,107)
      READ(5,*)DT
      IF(DT.EQ.0.)DT=0.1D0
      WRITE(6,108)
      READ(5,201)ICHW
      IF(ICHW.EQ.0)GO TO 13
      WRITE(6,109)
      DO 13 J=1,ICHW
      READ(5,*)CHT(J,1),VWIND(J,1)
13    CONTINUE
      WRITE(6,216)
      READ(5,*)ICOMP
      IF(ICOMP.EQ.0)GO TO 16
      WRITE(6,217)ICOMP
      READ(5,*)TFAIL,FAILVL
16    CONTINUE
      WRITE(6,110)
      READ(5,*)IDMP
C           PRINT BACK THIS INFORMATION
C
      WRITE(6,210)XX
      WRITE(6,211)VC
      IF(ICHV.EQ.0)GO TO 15
      DO 15 J=1,ICHV
      WRITE(6,212)J,CT(J,1),VCM(J,1)
15    CONTINUE
      WRITE(6,213)DT
      IF(ICHW.EQ.0)GO TO 14
      DO 14 J=1,ICHW
      WRITE(6,214)J,CHT(J,1),VWIND(J,1)
14    CONTINUE
      X(1,1)=XX
      X(2,1)=VC/AK4
      X(7,1)=VC
      VOLD=0.
      V2COMM=100.E0
      TM=0.0D0
      TTM=0.E0
      TTTM=TTM+DT
C
C           INITIALIZE DETECTION STATE XMES(6,1)
C
      XINI(1)=VC/AK4
      XINI(2)=0.E0
      XINI(3)=0.E0
      XINI(4)=0.E0
      XINI(5)=0.E0
      XINI(6)=VC
C
C           LEAD VEHICLE DYNAMICS
C
```

```
      XVH=1000.0
      VVH=20.0
      VVHEST=10.0
      DX=XVH-X(1,1)
C
C           FORM DETECTION MODEL DESIGN MATRICES
C
      CALL SOLUT(6)
      CALL MODEL(AMV,AMVEST,A2,A3,A4,AK1,AK2,AK3,AK4)
C
C           COMPUTE A MATRIX
C
      DO 1 J=1,7
      DO 1 K=1,7
      A(J,K)=0.0E0
1     CONTINUE
17    CONTINUE
      A(1,2)=1.0E0
      A(2,3)=1.0E0/AMV
      A(3,4)=1.0E0
      A(4,5)=1.0E0
      A(5,2)=-AK1*AK4*AK3
      A(5,3)=A4
      A(5,4)=A3
      A(5,5)=A2
      A(5,6)=AK2*AK3
      A(5,7)=AK1*AK3
      A(6,2)=-AK4
      A(6,7)=1.0E0
C
C           COMPUTE B MATRIX
C
      DO 2 J=1,7
      DO 2 K=1,5
      B(J,K)=0.0E0
2     CONTINUE
      B(2,1)=-CV/AMV
      B(2,3)=-BW/AMV
      B(2,4)=-1.0E0
      B(5,2)=AMVEST*AK3
      B(5,5)=-AK1*AK3*DN*AK4
      B(6,5)=-AK4*DN
      B(7,2)=DN
C
C           COMPUTE C MATRIX
C
      DO 3 J=1,3
      DO 3 K=1,7
      C(J,K)=0.0E0
3     CONTINUE
       C(1,2)=-AK1*AK4
      C(1,6)=AK2
      C(1,7)=AK1
       C(2,2)=AK4
      C(3,7)=1.E0
C
C           COMPUTE D MATRIX
C
      DO 4 J=1,3
      DO 4 K=1,5
      D(J,K)=0.0E0
```

```
      Y(J,1)=0.0E0
4     CONTINUE
      D(1,2)=AMVEST
      D(1,5)=-AK1*AK4
      D(2,5)=AK4
      IF(IP1.EQ.1)WRITE(6,218)
      IF(IP1.EQ.1)CALL MDUMP(A,7,7)
      IF(IP1.EQ.1)WRITE(6,219)
      IF(IP1.EQ.1)CALL MDUMP(B,7,5)
      IF(IP1.EQ.1)WRITE(6,220)
      IF(IP1.EQ.1)CALL MDUMP(C,3,7)
      IF(IP1.EQ.1)WRITE(6,221)
      IF(IP1.EQ.1)CALL MDUMP(D,3,5)
C
C           NAVIGATION LOOP
C
100   CONTINUE
C
C           COMPUTE NEW INPUT TO NON-LINEAR SYSTEM
C
C
C           FOLLOWER MODE SENSOR COMPUTATIONS
C
      DXP=DX
      XVH=XVH+VVH*DT
      DX=XVH-X(1,1)
      DX=1000.
C
C           VARIABLE GAIN G1 AND FOLLOWER COMMAND VELOCITY
C
      G1=0.2
      V2COMM=G1*DX
C
C           VELOCITY PROFILER
C
      IF(ICHV.EQ.0.AND.TM.LT.DT*.5)V1COMM=VC
      IF(ICHV.EQ.0.AND.TM.LT.DT*.5)GO TO 22
      DO 22 J=1,ICHV
      IF(DABS(TM-CT(J,1)).LE.DT*.5)V1COMM=VCM(J,1)
22    CONTINUE
C
C           TAKE SMALLER VELOCITY
C
      IF(V1COMM.LT.V2COMM)VCOMM=V1COMM
      IF(V1COMM.GE.V2COMM)VCOMM=V2COMM
      IF(ABS(VCOMM-VOLD).GT.0.01)IFL=0
      IF(V2COMM.LE.V1COMM)IFL=2
      IF(((ITM/IDMP)*IDMP).EQ.ITM)WRITE(6,222)(XMES(J),J=1,6),
     >YMES(1),YMES(2),EPS1(1),EPS1(2)
      CALL PROFLL(VCOMM,AMAX,AJMAX,DT,AC,VC,XX,IFL,IMD)
      VOLD=VCOMM
C
C           WIND GUST MODEL
C
C
      IF(ICHW.EQ.0)VW=0.0E0
      IF(ICHW.EQ.0)GO TO 20
      IF(ICHW.EQ.1)VW=VWIND(1,1)
      IF(ICHW.EQ.1)GO TO 20
      K=ICHW-1
      DO 20 J=1,K
```

```
      IF(TM.GT.CHT(J,1).AND.TM.LT.CHT(J+1,1))VW=VWIND(J,1)
20    CONTINUE
C             TRACK SLOPE MODEL
C
      CALL TSTTRK(X(1,1),GN,SLOPE,G)
C
C RANDOM NUMBER GENERATOR-GAUSSIAN DISTRIBUTION
C  ZERO MEAN -VARIANCE 1./100.
C
      WT=GGNOF(ISEED)/100.
C
C             INPUT VECTOR U
C
      IF(X(2,1).EQ.0.0)U(1,1)=0.0E0
      IF(X(2,1).NE.0.0)U(1,1)=X(2,1)/ABS(X(2,1))
      U(2,1)=AC
      U(3,1)=(X(2,1)+VW)**2
      U(4,1)=GN*SLOPE
C  NOISE INPUT
      U(5,1)=WT
C      WT=0.
      VIND=AK4*(X(2,1)+WT)
C      U(5,1)=0.E0
C   U(4,1) IS GRAVITY INPUT
C     U(4,1)=0.E0
C  U(3,1) IS AERO INPUT
C     U(3,1)=0.E0
C U(1,1) IS COULOMB INPUT
C     U(1,1)=0.E0
      IF(((ITM/IDMP)*IDMP).EQ.ITM)WRITE(6,213)TM,X(1,1),X(2,1),VC
     1,U(4,1),U(3,1),AC,V1COMM,VIND,U(1,1),IFL,IMD,IND
C
C             SOLUTION TO STATE EQUATION
C
      VERR=X(2,1)-VC
      CALL FINDIF(A,7,B,5,X,U,DT,AD1,AD2)
      DO 25 I=1,7
      XD(I)=X(I,1)
25    CONTINUE
C     CALL DVERK(NN,FCT,TTM,XD,TTTM,TOL,IND,CC,NN,WW,IER)
      DO 23 I=1,7
C     X(I,1)=XD(I)
23    CONTINUE
C
C        COMPUTE SYSTEM OUTPUT Y
C
      CALL MMULD(C,X,CX,3,7,1)
      CALL MMULD(D,U,DU,3,5,1)
      CALL MADDD(CX,DU,Y,3,1)
C
C        DETECTION INTERFACE
C
      CALL FILTER(6,XMES,YMES,IFLAG3,XINI,TTM,TTTM,TOL,Y,U
     1,EPS1)
      TTM=TTTM
      TTTM=TTTM+DT
      TM=TM+DT
      ITM=ITM+1
      TSTOP=20.0E0
C
C             FAILURE IMPLEMENTATION
```

```
C
      IF(ICOMP.EQ.0)GO TO 30
      TTEMP=TM
      IF(ABS(TFAIL-TTEMP).GT.DT*.5)GO TO 30
      IF(ICOMP.EQ.1)AMVEST=FAILVL
      IF(ICOMP.EQ.2)AMV=FAILVL
      IF(ICOMP.EQ.3)AK1=FAILVL
      IF(ICOMP.EQ.4)AK2=FAILVL
      IF(ICOMP.EQ.5)AK3=FAILVL
      IF(ICOMP.EQ.6)AK4=FAILVL
      IF(ICOMP.EQ.7)AAK1=FAILVL
      IF(ICOMP.EQ.8)AAK2=FAILVL
      IF(ICOMP.EQ.9)AAK3=FAILVL
      IF(ICOMP.EQ.10)AAK4=FAILVL
      IF(ICOMP.EQ.11)C1=FAILVL
      IF(ICOMP.EQ.12)C2=FAILVL
      IF(ICOMP.EQ.13)SPD1=FAILVL
      IF(ICOMP.EQ.14)SPD2=FAILVL
      WRITE(6,215)ICOMP,FAILVL,TM
      GO TO 17
30    CONTINUE
      IF(TM.LT.TSTOP)GO TO 100
      IF(IP1.EQ.1)CALL MDUMP(A,7,7)
      IF(IP1.EQ.1)CALL MDUMP(B,7,5)
      IF(IP1.EQ.1)CALL MDUMP(C,3,7)
      IF(IP1.EQ.1)CALL MDUMP(D,3,5)
      IF(IP1.EQ.1)CALL MDUMP(X,7,1)
      IF(IP1.EQ.1)CALL MDUMP(U,5,1)
      IF(IP1.EQ.1)CALL MDUMP(Y,3,1)
C
C            FORMAT STATEMENTS
C
102   FORMAT(' DESIGN MATRIX DUMP (1=YES)')
103   FORMAT(' INITIAL TRACK POSITION ')
104   FORMAT(' INITIAL VELOCITY (FT/SEC)')
105   FORMAT(' WAYSIDE COMMAND VELOCITY CHANGES (I1)')
106   FORMAT(' ENTER CHANGE TIME AND VCOMM')
107   FORMAT(' SAMPLING INCREMENT (DT=.1 DEFAULT)')
108   FORMAT(' WIND GUSTS (I1)')
109   FORMAT(' ENTER CHANGE TIME AND VWIND (+=HEADWIND)')
110   FORMAT(' INTEGER X,Y,U DUMP (NO. OF DTS)')
210   FORMAT(' INITIAL POSITION',F12.3,/)
211   FORMAT(' INITIAL VELOCITY',F12.3,/)
212   FORMAT(' VELOCITY CHANGE TIME, NEW VCOMMAND ',I3,2F8.2)
213   FORMAT(' TIME ',10(F8.2,1X),3I4)
214   FORMAT(' WIND LEVEL CHANGE TIME, NEW VELOCITY',I3,2F8.2)
202   FORMAT(F12.5)
201   FORMAT(I1)
215   FORMAT(/,' FAILURE OF DEVICE ',I3,2X,'NEW VALUE =',
     >F10.4,2X,'TM = ',F10.4,//)
216   FORMAT(' COMPONENT FAILURE NO. (ZERO IF NO FAIL)')
217   FORMAT(' TIME OF FAILURE AND NEW COMPONENT GAIN FOR NO. '
     .,I5)
218   FORMAT(/,' A MATRIX',/)
219   FORMAT(/,' B MATRIX',/)
220   FORMAT(/,' C MATRIX',/)
221   FORMAT(/,' D MATRIX',/)
222   FORMAT(' FILT OUT',10G10.2)
      STOP
      END
```

```
:READ  FILE1    FORTRAN  A1 DISCA   5/26/78  20:23
      SUBROUTINE FCT(NN,TTM,X,XPRIME)
      COMMON/A1/ A(7,7),B(7,5),U(5,1)
      DIMENSION XPRIME(7),X(7),X1(7,1),X2(7,1),XP(7,1)
      CALL MMULD(B,U,X1,NN,5,1)
      DO 1 I=1,7
      X2(I,1)=X(I)
1     CONTINUE
      CALL MMULD(A,X2,XP,NN,NN,1)
      DO 2 I=1,7
      XPRIME(I)=XP(I,1)+X1(I,1)
2     CONTINUE
      RETURN
      END
      FUNCTION SGN(X)
      IF(X.LT.0.E0)SGN=-1.E0
      IF(X.GE.0.E0)SGN=1.E0
      RETURN
      END
      SUBROUTINE FINDIF(A,N,B,M,X,U,DT,AD1,AD2)
      IMPLICIT REAL*4(A-H,O-Z)
      DIMENSION A(N,N),B(N,M),X(N,1),U(M,1),AD1(N,1),AD2(N,1)
      REAL*8 DT,TCOUNT
      CALL MMULD(A,X,AD1,N,N,1)
      CALL MMULD(B,U,AD2,N,M,1)
      DO 1 J=1,N
      X(J,1)=X(J,1)+(AD1(J,1)+AD2(J,1))*DT
1     CONTINUE
      RETURN
      END
      SUBROUTINE TSTTRK(X,GN,SLOPE,G)
C
C    TSTTRK - LONGITUDINAL PROFILE OF AGRT TEST TRACK, USED TO COMPUTE
C         SLOPE AND GRAVITY INFORMATION
C
C    INPUT - X - LONGITUDINAL POSITION (FT)
C         GN - NOMINAL GRAVITY (FT/SEC**2)
C
C    OUTPUT - SLOPE - TEST TRACK SLOPE AT X (RAD)
C         G - GRAVITY COMPONENT PARALLEL TO TRACK (FT/SEC**2)
C
C         LRV DETECTION FILTER SIMULATION - USDOT
C         MARCH 10, 1978 ... MICHAEL J. DYMENT
C
      IMPLICIT REAL*4(A-H,O-Z)
C
C         TRACK CONSTANTS
C         TRACK LENGTH = TL
C
      XL=6000.E0
      X1=1550.E0
      X2=1850.E0
      X3=2150.E0
      X4=2350.E0
      X5=2650.E0
      X6=2950.E0
C
C         RESET POSITION IF VEHICLE HAS LAPPED TRACK
C
      IF(X.GT.XL)X=X-XL
C
```

```
C           VERTICAL CURVE 1
C
      IF(X.LT.X1)GO TO 6
      IF(X.GE.X2)GO TO 2
      XT=X-X1
      DY=0.0002E0*XT
      GO TO 1
2     CONTINUE
C
C           VERTICAL CURVE 2
C
      IF(X.GE.X3)GO TO 3
      XT=X-X3
      DY=-0.0002E0*XT
      GO TO 1
3     CONTINUE
C
C           PLATEAU
C
      IF(X.GE.X4) GO TO 4
      DY=0.0E0
      GO TO 1
4     CONTINUE
C
C           VERTICAL CURVE 3
C
      IF(X.GE.X5) GO TO 5
      XT=X-X4
      DY=-0.0002E0*XT
      GO TO 1
5     CONTINUE
C
C           VERTICAL CURVE 4
C
      IF(X.GE.X6)GO TO 6
      XT=X-X6
      DY=0.0002E0*XT
      GO TO 1
6     CONTINUE
C
C           HORIZONTAL TRACK
C
      DY=0.0E0
1     CONTINUE
C
C           COMPUTE GRAVITY COMPONENT
C
      G=GN*DY
      SLOPE=DY
      RETURN
      END
      SUBROUTINE MMULD(A,B,C,L,M,N)
      REAL A(L,M),B(M,N),C(L,N)
      DO 1 I=1,L
      DO 1 J=1,N
      C(I,J)=0.0E0
      DO 2 K=1,M
      C(I,J)=C(I,J)+A(I,K)*B(K,J)
2     CONTINUE
1     CONTINUE
      RETURN
```

```
      END
      SUBROUTINE MADDD(A,B,C,M,N)
      REAL A(M,N),B(M,N),C(M,N)
      DO 1 I=1,M
      DO 1 J=1,N
      C(I,J)=A(I,J)+B(I,J)
1     CONTINUE
      RETURN
      END
      SUBROUTINE MSUBD(A,B,C,M,N)
      REAL A(M,N),B(M,N),C(M,N)
      DO 1 I=1,M
      DO 1 J=1,N
      C(I,J)=A(I,J)-B(I,J)
1     CONTINUE
      RETURN
      END
      SUBROUTINE PROFLL(VCOMM,AMAX,JMAX,DT,AA,VC,AX,IFL,IMD)
C
C
C     PROFLE:
C           CREATES A PROFILED VELOCITY COMMAND SUBJECT TO MAXIMUM
C           ACCELERATION AND JERK CRITERIA
C
C     INPUT
C           VCOMM - EXTERNAL COMMANDED VELOCITY (FT/SEC)
C           AMAX - ACCELERATION CRITERIA (FT/SEC**2)
C           JMAX - JERK CRITERIA (FT/SEC**3)
C            DT - INTEGRATION INTERVAL
C           IFL - FLAG
C                     0 - NEW VELOCITY COMMAND - SELECT NEW MODE
C                     1 - RETAIN PRESENT MODE
C           IMD - FLAG
C                     1 - STANDARD PROFILE
C                     2 - MODIFIED AMAX PROFILE
C                     3 - ZERO ACCELERATION PROFILE (PREP FOR IMD=1,2)
C
C     OUTPUT
C           AA - COMMANDED ACCELERATION
C           VC - COMMANDED VELOCITY
C           AX - COMMANDED POSITION
C
C
      IMPLICIT REAL*4(A-H,O-Z)
      REAL*8 DT
      REAL JMAX,JMX
C
C     TEST FOR MODE SELECT
C
      DV=VCOMM-VC
      IF(IFL.EQ.2)IMD=3
      IF(IFL.EQ.2)GO TO 10
      IF(IFL.NE.0)GO TO 10
      VI=AMAX**2/(2.E0*JMAX)
      IF(ABS(AA).GE.1.E-6)IMD=3
      IF(ABS(AA).GE.1.E-6)GO TO 10
      IF(ABS(DV).GE.2.E0*VI)IMD=1
      IF(ABS(DV).GE.2.E0*VI)GO TO 10
      IMD=2
10    CONTINUE
C
```

```
C       MODE SELECT LOCATION
C
        IF(IMD.EQ.1)GO TO 100
        IF(IMD.EQ.2)GO TO 200
        IF(IMD.EQ.3)GO TO 300
        IF(IMD.EQ.4)GO TO 400
C
C       MODE 1 - STANDARD VELOCITY PROFILE
C
100     CONTINUE
        IF(IFL.NE.0)GO TO 110
        TCOUNT=0.000D0
        AA=0.E0
        T1=AMAX/JMAX
        T2=(ABS(DV)-T1*AMAX)/AMAX
        IT1=T1/DT+1
        T1=FLOAT(IT1)*DT
        IT2=T2/DT+1
        T2=FLOAT(IT2)*DT
        JMX=ABS(DV)/(T1**2+T1*T2)
        AMX=JMX*T1
        JMX=JMX*SGN(DV)
        AMX=AMX*SGN(DV)
        T2=T2+T1
        T3=T1+T2
        IFL=1
110      CONTINUE
        TCOUNT=TCOUNT+DT
        IF(TCOUNT.GE.0.E0.AND.TCOUNT.LE.T1)AA=AA+JMX*DT
        IF(TCOUNT.GT.T1.AND.TCOUNT.LT.T2)AA=AMX
        IF(TCOUNT.GE.T2.AND.TCOUNT.LT.T3)AA=AA-JMX*DT
        IF(TCOUNT.GE.T3)AA=0.E0
        IF(ABS(AA).GT.AMAX)AA=AMX
        VC=VC+AA*DT
        AX=AX+VC*DT
        RETURN
C
C       MODE 2 - MODIFIED VELOCITY PROFILE
C
200     CONTINUE
        IF(IFL.NE.0)GO TO 210
        TCOUNT=0.0000D0
        DDV=ABS(DV)/2.E0
        AAMAX=SQRT(2.E0*DDV*JMAX)
        T1=AAMAX/JMAX
        IT=T1/DT+1
        T1=FLOAT(IT)*DT
        AAMAX=2.E0*DDV/T1
        JMX=AAMAX/T1
        AMX=ABS(AAMAX)*SGN(DV)
        JMX=ABS(JMX)*SGN(DV)
        T2=2.E0*T1
        AA=0.E0
        IFL=1
210     CONTINUE
        TCOUNT=TCOUNT+DT
        IF(TCOUNT.GE.0.E0.AND.TCOUNT.LE.T1)AA=AA+JMX*DT
        IF(TCOUNT.GT.T1.AND.TCOUNT.LE.T2)AA=AA-JMX*DT
        IF(TCOUNT.GT.T2)AA=0.E0
        VC=VC+AA*DT
        AX=AX+VC*DT
```

```
      RETURN
C
C     MODE 3 - PREPARE PROFILE FOR MODES 1 OR 2
C
300   CONTINUE
      IF(IFL.EQ.2)GO TO 330
      IF(IFL.NE.0)GO TO 310
      TCOUNT=0.D0
      T1=ABS(AA/JMAX)
      IT1=T1/DT+1
      T1=FLOAT(IT1)*DT
      JMX=AA/T1
      JMX=-SGN(AA)*ABS(JMX)
      IFL=1
310   CONTINUE
      TCOUNT=TCOUNT+DT
      IF(TCOUNT.GT.T1)GO TO 320
      AA=AA+JMX*DT
      VC=VC+AA*DT
      AX=AX+VC*DT
      RETURN
320   CONTINUE
      AA=0.E0
      AX=AX+VC*DT
      TCOUNT=TCOUNT+DT
      IFL=0
      RETURN
330   CONTINUE
      AAA=DV/DT
      AJT=AAA-AA
      AJT=AJT/DT
      IF(ABS(AJT).GT.JMAX)AA=AA+JMAX*SGN(AJT)*DT
      IF(ABS(AA).GT.AMAX)AA=AMAX*SGN(AA)
      VC=VC+AA*DT
      AX=AX+VC*DT
      RETURN
C
C     MODE 4 - UNSPECIFIED
C
400   CONTINUE
      RETURN
      END
      SUBROUTINE MDUMP(A,M,N)
      DIMENSION A(M,N)
      WRITE(6,100)
100   FORMAT('         ')
      DO 1 I=1,M
      PRINT10,I,(A(I,K),K=1,N)
1     CONTINUE
10    FORMAT (I2,1X,10(E9.3,1X))
      RETURN
      END
```

```
:READ  FILE2     FORTRAN  A1 DISCA    6/22/78  17:40
      SUBROUTINE FILTER(N,XMES,YMES,IFLAG,XINI,T,TEND,TOL,Y,UU,EPS1)
      EXTERNAL DETEC
      REAL*8 TINT
      COMMON/DET/ADET,BDET,CDET,DDET,EDET,IP,IQ,IS,U,EPS,BP
      DIMENSION ADET(6,6),BDET(6,2),CDET(2,6),DDET(6,2)
      DIMENSION EDET(2,2),XMES(6),YMES(2),XINI(6)
      DIMENSION EPS1(2),V(4),AD1(5,1),AD2(5,1)
     .,UU(5),U(2),EPS(2),BUF(3)
      DIMENSION BP(6,4),Y(3)
      TINT=TEND-T
      IF(IFLAG.NE.0) GO TO 3
      DO 1 I=1,N
 1    XMES(I)=XINI(I)
      DO 2 I=1,IP
  2   EPS(I)=0.
C2    EPS(I)=2.
 3    CONTINUE
      IFLAG=IFLAG+1
      NW=NW+1
      U(1)=UU(2)
      U(2)=1.
      V(1)=UU(2)
C
C       COULOMB FRICTION COMPONENT
C
      V(2)=1.
C     V(2)=0.
      V(3)=EPS(1)
      V(4)=EPS(2)
      IND=1
      CALL FINDIF(ADET,6,BP,4,XMES,V,TINT,AD1,AD2)
C     CALL DVERK(N,DETEC,T,XMES,TEND,IND,C,NW,W,IER)
C** COMPUTATION OF YMES,EPS
      DO 4 I=1,IP
      BUF(I)=0.
      DO 4 J=1,N
 4    BUF(I)=BUF(I)+CDET(I,J)*XMES(J)
      DO 5 I=1,IS
      YMES(I)=0.
      DO 5 J=1,IQ
 5    YMES(I)=YMES(I)+EDET(I,J)*U(J)
      DO 6 I=1,IP
 6    YMES(I)=YMES(I)+BUF(I)
      DO 7 I=1,IP
 7    EPS(I)=Y(I)-YMES(I)
C     DO 8 I=1,IP
C8    EPS1(I)=EPS(I)
      EPS1(1)=EPS(1)+1500.*EPS(2)
C     EPS1(2)=373.*EPS(2)
      EPS1(2)=310.83*EPS(2)
      RETURN
      END
      SUBROUTINE DETEC(N,T,XMES,XMESP)
      COMMON/DET/ADET,BDET,CDET,DDET,EDET,IP,IQ,IS,U,EPS,BP
      DIMENSION ADET(6,6),BDET(6,2),CDET(2,6),DDET(6,2)
      DIMENSION EDET(2,2),U(2),EPS(2),BUF1(6),BUF2(6)
      DIMENSION XMES(6),XMESP(6),BP(6,4)
C** COMPUTES XMESP=ADET*XMES+BDET*U+DDET*EPS
      DO 1 I=1,N
      BUF1(I)=0.
```

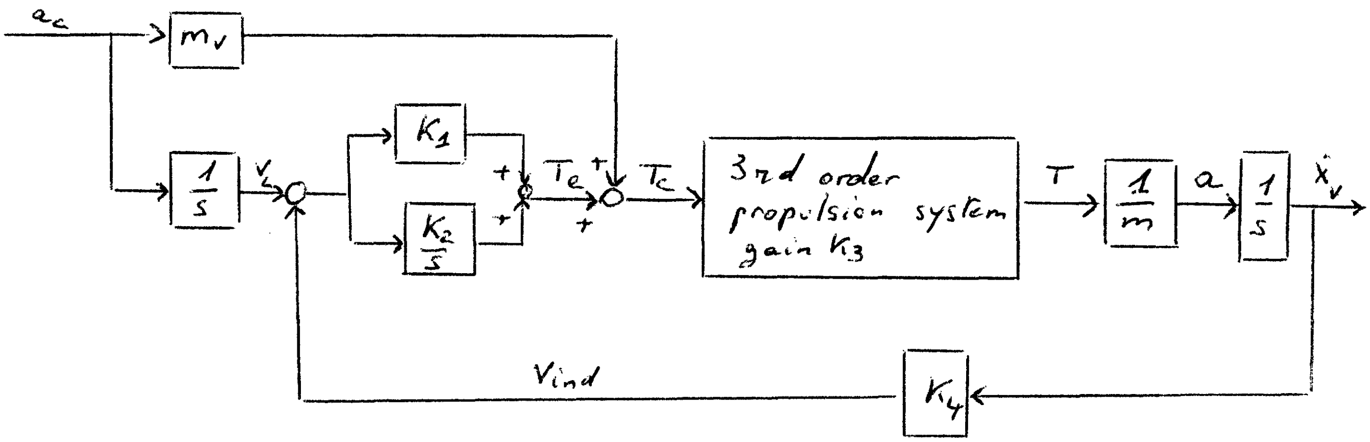
```
      DO 1 J=1,N
    1 BUF1(I)=BUF1(I)+ADET(I,J)*XMES(J)
      DO 2 I=1,N
      BUF2(I)=0.
      DO 2 J=1,IQ
    2 BUF2(I)=BUF2(I)+BDET(I,J)*U(J)
      DO 3 I=1,N
      XMESP(I)=0.
      DO 3 J=1,IP
    3 XMESP(I)=XMESP(I)+DDET(I,J)*EPS(J)
      DO 4 I=1,N
    4 XMESP(I)=XMESP(I)+BUF1(I)+BUF2(I)
      RETURN
      END
      SUBROUTINE MODEL(AMV,AMVEST,A2,A3,A4,AK1,AK2,AK3,AK4)
C
C COMPUTES ADET,BDET,CDET,DDET,EDET,BP
C
      COMMON/DET/ADET,BDET,CDET,DDET,EDET,IP,IQ,IS,U,EPS,BP
      DIMENSION ADET(6,6),BDET(6,2),CDET(2,6),DDET(6,2)
      DIMENSION EDET(2,2)
      DIMENSION BP(6,4),U(2),EPS(2)
      IP=2
      IQ=2
      IS=2
      N=6
      DO 1 I=1,N
      DO 1 J=1,N
    1 ADET(I,J)=0.
      ADET(1,2)=1./AMV
      ADET(2,3)=1.
      ADET(3,4)=1.
      ADET(4,2)=A4
      ADET(4,3)=A3
      ADET(4,4)=A2
      ADET(4,5)=AK3
      ADET(5,1)=-AK2*AK4
      ADET(5,2)=-AK1*AK4/AMV
      ADET(5,6)=AK2
      DO 2 I=1,N
      DO 2 J=1,IQ
    2 BDET(I,J)=0.
      BDET(4,1)=AK3*AMVEST
      BDET(5,1)=AK1
      BDET(6,1)=1.
C
C   COULOMB FRICTION PARAMETERS
C
      BDET(1,2)=-100./AMV
      BDET(5,2)=AK1*AK4*100./AMV
      DO 3 I=1,IP
      DO 3 J=1,N
    3 CDET(I,J)=0.
      CDET(1,5)=1.
      CDET(2,1)=AK4
      DO 4 I=1,IP
      DO 4 J=1,IS
    4 EDET(I,J)=0.
      EDET(1,1)=AMVEST
      DO 5 I=1,N
      DO 5 J=1,IQ
```

```
 5    BP(I,J)=BDET(I,J)
      DO 6 I=1,N
      DO 6 J=1,IP
      JJ=J+IQ
 6    BP(I,JJ)=DDET(I,J)
      WRITE(6,100)((ADET(I,J),J=1,N),I=1,N)
 100  FORMAT(6(1X,'ADET=',E10.4))
      WRITE(6,101)((BDET(I,J),J=1,IQ),I=1,N)
 101  FORMAT(2(1X,'BDET=',E10.4))
      WRITE(6,102)((CDET(I,J),J=1,N),I=1,IP)
 102  FORMAT(6(1X,'CDET=',E10.4))
      WRITE(6,103)((EDET(I,J),J=1,IS),I=1,IP)
      WRITE(6,104)((BP(I,J),J=1,4),I=1,6)
 104  FORMAT(4(1X,'BP=',E10.4))
 103  FORMAT(2(1X,'EDET=',E10.4))
      RETURN
      END
      SUBROUTINE SOLUT(N)
C
C  READS DDET
C
      COMMON/DET/ADET,BDET,CDET,DDET,EDET,IP,IQ,IS,U,EPS,BP
      DIMENSION ADET(6,6),BDET(6,2),CDET(2,6),DDET(6,2)
      DIMENSION EDET(2,2),BP(6,4),U(1),EPS(2)
      IP=2
      WRITE(6,100)
 100  FORMAT(1X,'TYPE DDET,DETECTION FILTER')
      READ(6,*)((DDET(I,J),J=1,IP),I=1,N)
      WRITE(6,101)((DDET(I,J),J=1,IP),I=1,N)
 101  FORMAT(2(1X,'DDET=',E10.4))
      RETURN
      END
```

APPENDIX C

A PROBLEM MET IN THE FILTER DESIGN

The reference model for the filter is the following one:



Initially, K_4 nominal value was selected to be 1. The first choice of states was V_{ind}, T, $\dot{T}$, $\ddot{T}$, T_e, V_c. The outputs are T_c and V_{ind}, the input is a_c. We have the equations

$$\dot{V}_{ind} = \frac{K_4 T}{m}$$

$$(T)^{\bullet} = \dot{T}$$

$$(\dot{T})^{\bullet} = \ddot{T}$$

$$(\ddot{T})^{\bullet} = a_4 T + a_3 \dot{T} + a_2 \ddot{T} + K_3 T_c$$

$$= a_4 T + a_3 \dot{T} + a_2 \ddot{T} + K_3 T_e + K_3\, m_v a_c$$

$$T_e = K_1 (\dot{V}_c - \dot{V}_{ind}) + K_2\, (V_c - V_{ind})$$

$$= K_1\, a_c - K_1 K_4 \frac{T}{m} + K_2 V_c - K_2 V_{ind}$$

$$\dot{V}_c = a_c$$

Furthermore

$$\begin{cases} T_c = T_e + m_v a_c \\ V_{ind} = V_{ind} \end{cases}$$

In matrix form

$$(A\text{-}1) \quad \begin{pmatrix} \dot{V}_{ind} \\ \dot{T} \\ \ddot{T} \\ \dddot{T} \\ \dot{T}_e \\ \dot{V}_c \end{pmatrix} = \begin{pmatrix} 0 & \frac{K_4}{m} & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & a_4 & a_3 & a_2 & K_3 & 0 \\ -K_2 & -\frac{K_1 K_4}{m} & 0 & 0 & 0 & K_2 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} V_{ind} \\ T \\ \dot{T} \\ \ddot{T} \\ T_e \\ V_c \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \\ K_3 m_v \\ K_1 \\ 1 \end{pmatrix} a_c$$

$$(A\text{-}2) \quad \begin{pmatrix} T_e \\ V_{ind} \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} V_{ind} \\ T \\ \dot{T} \\ \ddot{T} \\ T_e \\ V_c \end{pmatrix} + \begin{pmatrix} m_v \\ 0 \end{pmatrix} a_c$$

Event associated with K_4: $\begin{pmatrix} 1/m \\ 0 \\ 0 \\ 0 \\ -K_1/m \\ 0 \end{pmatrix} = \underline{b}_4$

$$C \underline{b}_4 = \begin{pmatrix} -\frac{K_1}{m} \\ \frac{1}{m} \end{pmatrix}$$

Event associated with K_1: $\begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \underline{b}_1 \qquad C\underline{b}_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$

Event associated with K_2: $\begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \underline{b}_2 \qquad C\underline{b}_2 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$

($\underline{b}_1 = \underline{b}_2$)

Event associated with K_3: $\begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \underline{b}_3$

But $C\underline{b}_3 = 0$. We compute $A\underline{b}_3$, $CA\underline{b}_3 = 0$, we compute $A^2\underline{b}_3$, $CA^2\underline{b}_3 = 0$. We compute

$$A^3\underline{b}_3 = \begin{pmatrix} \frac{K_4}{m} \\ a_2 \\ a_3 + a_2^2 \\ a_4 + 2a_3a_2 + a_3^2 \\ -\frac{K_4K_1}{m} \\ 0 \end{pmatrix} \qquad CA^3\underline{b}_3 = \begin{pmatrix} -\frac{K_4K_1}{m} \\ \frac{K_4}{m} \end{pmatrix}$$

It appears that failures in K_1 and K_2 are detection equivalent, which can be accepted. However, with this choice of states, failures in K_3 and failures in K_4 are not output separable (in fact $\underline{b}_3 \in \overline{R}_4$, $\underline{b}_3$ and $\underline{b}_4$ are detection equivalent). This cannot be tolerated.

As C is of rank 2, the easiest way to distinguish between 3 kinds of failures is to have 2 of them generate a unidirectional output, and the 3rd one generate a planar output. To do this, a set of states was selected such that one failure was a sensor failure. The new set selected was $\dot{X}_v$, T, $\dot{T}$, $\ddot{T}$, T_e, V_c. We have the equations

$$T_c = T_e + m_v a_c$$

$$\dot{T}_e = K_1 a_c - K_1 K_4 \frac{T}{m} + K_2 V_c - K_2 K_4 \dot{X}_v$$

$$(\dot{X}_v)^{\boldsymbol{\cdot}} = T/m$$

$$(T)^{\boldsymbol{\cdot}} = \dot{T}$$

$$(\dot{T})^{\boldsymbol{\cdot}} = \ddot{T}$$

$$(\ddot{T})^{\boldsymbol{\cdot}} = K_3 T_e + K_3 m_v a_c + a_2 \ddot{T} + a_3 \dot{T} + a_4 T$$

$$\dot{V}_c = a_c$$

In matrix form

$$(A\text{-}3)\quad \begin{pmatrix} \ddot{X}_v \\ \dot{T} \\ \ddot{T} \\ \dddot{T} \\ \dot{T}_e \\ \dot{V}_c \end{pmatrix} = \begin{pmatrix} 0 & \frac{1}{m} & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & a_4 & a_3 & a_2 & K_3 & 0 \\ -K_2 K_4 & -\frac{K_1 K_4}{m} & 0 & 0 & 0 & K_2 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \dot{X}_v \\ T \\ \dot{T} \\ \ddot{T} \\ T_e \\ V_c \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \\ K_3 m_v \\ K_1 \\ 1 \end{pmatrix} a_c$$

$$(A\text{-}4)\quad \begin{pmatrix} T_c \\ V_{ind} \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 \\ K_4 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \dot{X}_v \\ T \\ \dot{T} \\ \ddot{T} \\ T_e \\ V_c \end{pmatrix} + \begin{pmatrix} m_v \\ 0 \end{pmatrix} a_c$$

Event associated with K_1: $\begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \underline{b}_1 \qquad C\underline{b}_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$

Event associated with K_2: $\begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \underline{b}_2 = \underline{b}_1$

Event associated with K_3 $\begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \underline{b}_3$

$C\underline{b}_3 = 0$ we compute

$A\underline{b}_3$, $CA\underline{b}_3 = 0$, $A^2\underline{b}_3$, $CA^2\underline{b}_3 = 0$

$$A^3\underline{b}_3 = \begin{pmatrix} 1/m \\ a_2 \\ a_3 + a_2^2 \\ a_4 + 2a_3 a_2 + a_2^3 \\ -k_1 k_4/m \\ 0 \end{pmatrix} \qquad CA^3\underline{b}_3 = \begin{pmatrix} -k_1 k_4/m \\ k_4/m \end{pmatrix}$$

Event associated with K_4:

$$\Delta C = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \qquad \underline{b}_4 = \underline{b}_1 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

As the vector $\underline{e}_{61} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$ is such that $C\,\underline{e}_{61} \parallel \begin{pmatrix} 0 \\ 1 \end{pmatrix}$

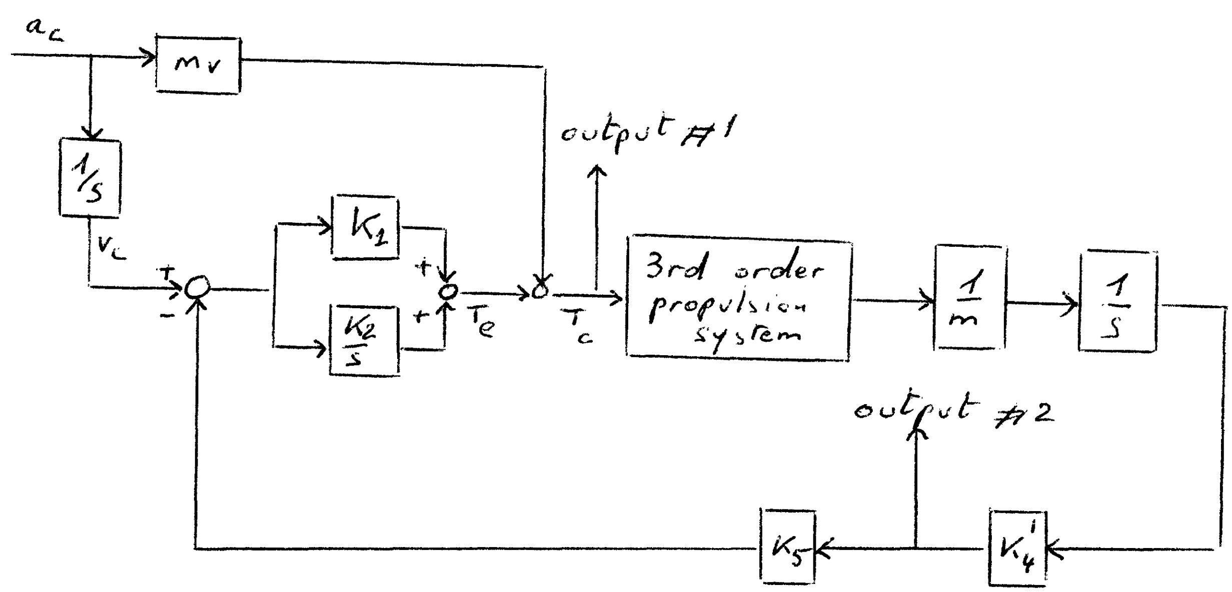
and $A\underline{e}_{61} \parallel \underline{b}_1$. The events associated with a failure in K_4 are $\underline{e}_{61}$ and $\underline{b}_1$. As $C\underline{e}_{61}$ and $C\underline{b}_1$ are linearly independent, a failure in K_4 generates an output constrained to a plane.

With $K_4 = 1$, a detection filter was designed for the events $\underline{b}_1$ and $A^3\underline{b}_3$. Its numerical value was

$$(A\text{-}5) \qquad D = \begin{pmatrix} 0 & -15.5 \\ 0 & -2331.75 \\ 0 & 5\,567\,951 \\ 12150 & -231\,628\,128 \\ 20 & 52\,243.35 \\ 0.1 & 149.98 \end{pmatrix} \qquad C = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Running the simulation with this filter, it was discovered that a failure in K_4 generated a unidirectional output along $\underline{\xi}_2$. The physical reason is obvious: the A matrices in (A-1) and A-3) differ only in two places—the terms A_{12} and A_{51} are different in each case, by a factor of K_4. Furthermore, the C matrices in (A-2) and (A-4) differ only in one place: C_{21} has a different value, the ratio between the two different values being K_4. If K_4 is equal to 1, the matrix equations in (A-1), (A-3) and in (A-2), (A-4) are numerically equal. A detection filter designed for $\underline{b}_1$ and $A^3\underline{b}_3$ will have the same numerical value in the two cases. In other words, more intuitively, if $K_4 = 1$, an outside observer would not know in looking only at the numerical equations whether the states V_{ind}, T, $\dot{T}$, $\ddot{T}$, T_e, V_c or the states $\dot{X}_v$, T, $\dot{T}$, $\ddot{T}$, T_e, V_c are selected. It does then make sense that a failure in K_4 generates unidirectional output along $\mathcal{E}_2$ because in the first case (to which it is numerically equal), this is what happens.

The solution is obvious: to give to K_4 a value different from 1. It was first attempted to perform this without changing the physical value of the tachometer gain, but just the system model. The new reference model was:



If $K_5K_4' = 1$, nothing is changed in the real system.

This could be done physically by multiplying the real output V_{ind} by K_4' before the comparison of the system output # 2 and the reference output # 2. If this multiplication is digitally made by a computer, it could be considered exact.

With this reference model, the equations are

$$T_c = T_e + m_v a_c$$

$$\dot{T}_e = K_1 a_c - K_1 K_5 K_4' \frac{T}{m} + K_2 V_c - K_2 K_5 K_4' \dot{X}_v$$

$$(\dot{X}_v)^{\cdot} = T/m$$

$$(T)^{\cdot} = \dot{T}$$

$$(\dot{T})^{\cdot} = \ddot{T}$$

$$(\ddot{T})^{\cdot} = K_3 T_e + K_3 m_v a_c + a_2 \ddot{T} + a_3 \dot{T} + a_4 T$$

In matrix form

$$(A\text{-}6) \quad \begin{pmatrix} \ddot{X}_v \\ \dot{T} \\ \ddot{T} \\ \dddot{T} \\ \dot{T}_e \\ \dot{V}_c \end{pmatrix} = \begin{pmatrix} 0 & \frac{1}{m} & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & a_4 & a_3 & a_2 & K_3 & 0 \\ -K_1 K_5 K_4' & -\frac{K_1 K_5 K_4'}{m} & 0 & 0 & 0 & K_2 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \dot{X}_v \\ T \\ \dot{T} \\ \ddot{T} \\ T_e \\ V_c \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \\ K_3 m_v \\ K_1 \\ 1 \end{pmatrix} a_c$$

$$\begin{pmatrix} T_c \\ V_{ind} \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 \\ K_4' & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} \dot{X}_v \\ T \\ \dot{T} \\ \ddot{T} \\ T_e \\ V_c \end{pmatrix} + \begin{pmatrix} m_v \\ 0 \end{pmatrix} a_c$$

Event associated with K_1: $\underline{b}_1 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$ $\quad C\underline{b}_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$

Event associated with K_2: $\underline{b}_1$

Event associated with K_3: $\underline{b}_3 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix}$ $\quad C\underline{b}_3 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$

$A^3\underline{b}_3$ is such that $CA^3\underline{b}_3 = 0$, as $CA^2\underline{b}_3 = 0$, the event associated with K_3 will be $A^3\underline{b}_3$

$$A^3\underline{b}_3 = \begin{pmatrix} 1/m \\ a_2 \\ a_3 + a_2^2 \\ a_4 + 2a_3a_2 + a_2^3 \\ -\frac{K_1 K_5 K'_4}{m} \\ 0 \end{pmatrix} \qquad CA^3\underline{b}_3 = \begin{pmatrix} -\frac{K_1 K_5}{m} K'_4 \\ \frac{K'_4}{m} \end{pmatrix}$$

Events associated with K'_4:

a - variation in A along $\underline{b}_1 = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$

b - variation in C along $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$ As $\underline{F} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$ is such that

$C\underline{F} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$ and $A\underline{F} // \underline{b}_1$

The events associated with K_4' are $\underline{b}_1$ and $\underline{F}$.

A detection filter was designed for failures in K_1 and K_3, with the same eigenvalues as those selected for (A-5). Numerically, it was found, with $K_5 = 2$, $K_4' = .5$.

$$(A\text{-}7) \quad D = \begin{pmatrix} 0 & -31 \\ 0 & -4662.75 \\ 0 & 11\,135\,904. \\ 12\,150. & -463\,225\,657.6 \\ 20. & 104\,486.687 \\ 0.1 & 299.96 \end{pmatrix} \quad C = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 \\ .5 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

It appears that, except for numerical roundoff, the products DC of the matrices given in (A-5) and in (A-7) are equal. The behavior of the error $\underline{\varepsilon}$, whose differential equation is $\dot{\underline{\varepsilon}} = (A\_DC)\,\underline{\varepsilon} + \underline{b}_i n_i(t)$ will be the same. This explains why the filter D in (A-7) cannot distinguish between a failure in K_4 and a failure in K_3, as was discovered in running a test.

It was then decided to give to K_4 a value of 1.20, a failure in K_4 then generated outputs along both channels $\underline{\varepsilon}_1$ and $\underline{\varepsilon}_2$. The value 1.20 is, of course, arbitrary. An actual velocity indicator would have a scale factor relating input velocity to output signal which is determined by the instrument. Its value would almost certainly not be 1.0. However, if K_4 is not equal to 1, in steady state, $\dot{X}_V$ will not be equal to V_c, but to V_c/K_4, with the actual system configuration, when the integrator of a_c has a gain 1. In other words, physically, to reach a desired velocity V, the velocity command must be equal to K_4V. This can be done by inserting a gain K_4 at the output of the profiler, just before the beginning of the velocity control loop.

It must be emphasized that these filters were not designed to detect velocity sensor failures; therefore they do not prescribe by design the behavior of the errors in response to a velocity sensor failure. These filters were designed only to constrain the error due to controller failures to $\mathcal{E}_1$ and the error due to propulsion failures to $\mathcal{E}_2$. The error response to a velocity sensor failure is then a matter of chance, and it just happens that with $K_4 = 1.0$ the error is contained along $\mathcal{E}_2$.

APPENDIX D

ORTHOGONAL REDUCTION PROCEDURE

Orthogonal reduction is a procedure which determines the null space of a matrix V; i.e., all independent solutions of $V\underline{w} = 0$. Suppose V is $n' \cdot n$

$$V = \begin{bmatrix} \underline{v}_1^T \\ \vdots \\ \underline{v}_{n'}^T \end{bmatrix}$$

The orthogonal reduction procedure is an iterative process which generates an $n \cdot n$ symmetric positive semi-definite matrix whose range space coincides with the null space of V. In each iteration, a row of V is tested to determine if it is orthogonal to the range space of the symmetric matrix. If not, the range space of the matrix is reduced so that this is the case. The procedure begins with any symmetric positive definite $n \cdot n$ matrix $\Omega^{(1)}$. An auxiliary n-vector is defined by $\underline{w}_1 = \Omega^{(1)}\underline{v}_1$. If $\underline{v}_1$ is nonzero $\underline{w}_1$ will be nonzero, since $\Omega^{(1)}$ is positive definite. Furthermore $\underline{w}_1^T\underline{v}_1$ will be nonzero. A new symmetric positive semi-definite matrix is defined by

$$\Omega^{(2)} = \Omega^{(1)} - \frac{\underline{w}_1\underline{w}_1^T}{\underline{w}_1^T\underline{v}_1}$$

This matrix has the property that $\Omega^{(2)}\underline{v}_1 = 0$.

The procedure continues according to the following general iteration

(1) with $\Omega^{(i)}$ from the previous iteration, form the auxiliary vector $\underline{w}_i = \Omega^{(i)}\underline{v}_i$

(2) if $\underline{w}_i \neq 0$ set $\Omega^{(i+1)} = \Omega^{(i)} - \dfrac{\underline{w}_i \underline{w}_i^T}{\underline{w}_i^T \underline{v}_i}$

if $\underline{w}_i = 0$ set $\Omega^{(i+1)} = \Omega^{(i)}$ and return to (1).

The algorithm has the following important properties:

(1) If $\Omega^{(i)}$ is positive semidefinite, $\underline{w}_i^T \underline{v}_i = 0$ if and only if $\underline{w}_i = 0$. This follows from the definition of $\underline{w}_i$.

(2) If $\Omega^{(i)}$ is positive semidefinite so is $\Omega^{(i+1)}$. This is obviously true if $\underline{w}_i = 0$. Assume $\underline{w}_i \neq 0$. For any arbitrary n-vector $\underline{z}$ and any scalar α

$$(\underline{z} - \alpha \underline{v}_i)^T \Omega^{(i)} (\underline{z} - \alpha \underline{v}_i) \geq 0 \qquad \text{(A-8)}$$

In particular this must be true for

$$\alpha = \frac{\underline{w}_i^T \underline{z}}{\underline{w}_i^T \underline{v}_i}$$

Substituting this value of α in (A8) and expanding it, we get

$$(\underline{z} - \alpha \underline{v}_i)^T \Omega^{(i)} (\underline{z} - \alpha \underline{v}_i) = \underline{z}^T \Omega^{(i)} \underline{z} - 2\alpha \underline{v}_i^T \Omega^{(i)} \underline{z} + \alpha^2 \underline{v}_i^T \Omega^{(i)} \underline{v}_i$$

$$= \underline{z}^T \Omega^{(i)} \underline{z} - 2\alpha \underline{w}_i^T \underline{z} + \alpha^2 \underline{w}_i^T \underline{v}_i$$

$$= \underline{z}^T \Omega^{(i)} \underline{z} - \frac{(\underline{w}_i^T \underline{z})^2}{\underline{w}_i^T \underline{v}_i}$$

$$= \underline{z}^T \Omega^{(i+1)} \underline{z} \geq 0 \qquad \text{(A-9)}$$

By induction, this shows that all $\Omega^{(i)}$ are positive semidefinite if the starting matrix $\Omega^{(1)}$ is at least positive semidefinite.

(3) If $\underline{w}_i \neq 0$ then $\text{rk}\, \Omega^{(i+1)} = \text{rk}\, \Omega^{(i)} - 1$

and the null space of $\Omega^{(i+1)}$ is the subspace formed by $\underline{v}_i$ and the null space of $\Omega^{(i)}$.

In equation (A.9) equality holds (and thus $\Omega^{(i+1)} \underline{z} = 0$) if and only if $(\underline{z} - \alpha \underline{v}_i)$ Iies in the null space of $\Omega^{(i)}$. But this implies $\underline{z}$ must be in the subspace formed by $\underline{v}_i$ and the null space of $\Omega^{(i)}$.

(4) At any point in the process the range space of $\Omega^{(i)}$ is made up of all vectors orthogonal to $\{\underline{v}_1,\ldots,\underline{v}_{i-1}\}$ (if the starting matrix is positive definite only. In step 5b and 5c of the detection filter design algorithm, the range space of $\Omega^{(i)}$ contains all vectors orthogonal to $\{\underline{v}_1,\ldots,\underline{v}_{i-1}\}$, but may have additional ones as well). If $\Omega^{(i)}$ is positive definite, when all the rows of V have been processed, the final matrix $\Omega^{(n'+1)}$ has a range space which coincides with the null space of V. The number of reductions made is equal to the rank of V.

(5) If $\Omega^{(1)}$ is positive definite and $\underline{w}_i = 0$, then $\underline{v}_i$ is linearly dependent on the preceding vectors $\{\underline{v}_1,\ldots,\underline{v}_{i-1}\}$. By virtue of property (4), the vectors $\{\underline{v}_1,\ldots,\underline{v}_{i-1}\}$ span the null space of $\Omega^{(i)}$. Since $\underline{w}_i = 0$ implies $\underline{v}_i$ is in the null space of $\Omega^{(i)}$, it must be expressible as a linear combination of the vectors $\{\underline{v}_1,\ldots,\underline{v}_{i-1}\}$.

In step 5c of the detection filter design algorithm, a matrix Ω_i was found where columns span the space R_i. In step 5d the orthogonal reduction procedure is applied to

$$M = \begin{bmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{bmatrix} \qquad \text{starting with } \Omega_i \qquad (\text{for } i = 1,\ldots,r)$$

As, by definition, $R_i \subset \eta(C)$ this orthogonal reduction will end on a zero matrix. The detection generator $\underline{g}_i$ of R_i is a multiple of the last nonzero auxiliary vector before termination, if rk $(\bar{R}_i) \neq 1$.

$$\underline{w}_i = \Omega^{(i)} (\underline{c}_j A^{\nu_i - 1})^T \neq 0$$

By construction $\underline{w}_i$ lies in the null space of M and satisfies

$$\begin{bmatrix} C \\ CA \\ \vdots \\ CA^{\nu_i - 2} \end{bmatrix} \underline{w}_i = 0 \qquad \text{and} \qquad CA^{\nu_i - 1} \underline{w}_i \neq 0$$

These are all the requirements for a detection generator, except for the magnitude. $\underline{w}_i$ is a multiple of the detection generator $\underline{g}_i$.

If rk $(\bar{R}_i) = 1$, rk$(R_i) = 0$ as $\bar{R}_i = \underline{b}_i \oplus R_i$ (if $C\underline{b}_i \neq 0$, otherwise, use $A\underline{b}_i$ etc). Then $\Omega_i = 0$, and there is no nonzero auxiliary vector before termination in the orthogonal reduction of M. It is trivial in this case to find the detection generator: it is $\underline{b}_i$.

Intermediate turning points

In step 5b of the detection filter design algorithm, orthogonal reduction is applied to a matrix

$$M_D = \begin{bmatrix} C_S' \\ C_S'(A - D_S C) \\ \vdots \\ C_S'(A - D_S C)^{n-1} \end{bmatrix}$$

starting with a positive definite matrix (on option). The rows of M_D correspond to the $\underline{v}_i^T$ defined earlier. Because of the cyclic manner in which the rows of M_D are generated it is not necessary to process all the rows. A row can be skipped if it is known that it is linearly dependent on preceding rows, because the auxiliary vector in that case will be zero. When a particular auxiliary vector is found to be zero, for example $\underline{w}_i = \Omega^{(i)} (C_j (A-D_s C)^{\ell})^T = 0$, where C_j is the jth row of C_s', it is then known that $C_j (A-D_s C)^{\ell}$ is linearly dependent on the preceding rows in M_D. But if this is so, then all the remaining rows of M_D generated by C_j (i.e., $C_j (A-D_s C)^k$, $k > \ell$) will also be dependent on preceding rows of M_D. The auxiliary vectors associated with these rows will all be zero, so there is no need to consider them in the reduction procedure. The appearance of the first zero auxiliary vector will be referred to as the intermediate turning point for C_j.

(Note: this intermediate turning point notion is not valid when the starting matrix is not positive definite. In that case an auxiliary vector could be zero even if the row processed were not linearly dependent on the preceding rows.)