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
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 uni-
directional 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 applica-
tion 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

<table>
<thead>
<tr>
<th>Chapter No.</th>
<th>Title</th>
<th>Page No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>Theoretical Background</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>Detection Filter Design Algorithm</td>
<td>22</td>
</tr>
<tr>
<td>4</td>
<td>Guideway Vehicle Simulation and Detection Filter Design</td>
<td>43</td>
</tr>
<tr>
<td>5</td>
<td>Experimental Results</td>
<td>68</td>
</tr>
<tr>
<td>6</td>
<td>Conclusions</td>
<td>85</td>
</tr>
</tbody>
</table>

**Appendices**

<table>
<thead>
<tr>
<th></th>
<th>Title</th>
<th>Page No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Listing of Detection Filter Design Program</td>
<td>A-1</td>
</tr>
<tr>
<td>B</td>
<td>Listing of Longitudinal Guideway Vehicle Simulation</td>
<td>B-1</td>
</tr>
<tr>
<td>C</td>
<td>A Problem Met in Detection Filter Design</td>
<td>C-1</td>
</tr>
<tr>
<td>D</td>
<td>Orthogonal Reduction Procedure</td>
<td>D-1</td>
</tr>
</tbody>
</table>

**References** 86
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 nonlinear 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 $x$ be the state vector ($n$ dimensioned)
$u$ be the input vector ($q$ dimensioned)
$y$ be the output vector ($m$ dimensioned)

The system equations are

\[
\dot{x} = Ax + Bu
\]
\[
y = Cx
\]

where

$A$ is an $n \times n$ matrix
$B$ is an $n \times q$ matrix
$C$ is an $m \times n$ matrix

From the comparison between $y$, the output of the system measured by means of sensors, and $y_m$, 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:

![Diagram of General Reference Model]

Equations are

\[
\begin{align*}
\dot{x} &= A \dot{x} + B u \\
\dot{y} &= C \dot{x}
\end{align*}
\]

\[
\begin{align*}
\dot{x}_m &= A_m x_m + B_m u + D(y - y_m) \\
y_m &= C_m x_m
\end{align*}
\]

\[
\begin{align*}
(\dot{x}_m(0) \text{ is necessary to start the simulation, but as } (A - DC) \text{ is designed to have no positive real part eigenvalues, } x_m(0) \text{ does not need to be equal to } x(0) \text{ for } x(t) \text{ and } x_m(t) \text{ to be equal in steady state}).
\end{align*}
\]

In the absence of failure \( A_m \equiv A \), \( B_m \equiv B \), \( C_m \equiv C \).

Introducing the error vector \( \xi = x - x_m \) we have

\[
\begin{align*}
\dot{x} - \dot{x}_m &= A(x - x_m) - D(y - y_m) \\
&= A(x - x_m) - DC(x - x_m) \\
\dot{\xi} &= (A - DC) \xi \\
\xi(0) &= x(0) - x_m(0)
\end{align*}
\]
The output error $\xi' = y - y_m$ then follows the equation

$$\xi' = C \xi.$$

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 modeled according to one of the two following ways:

- **controller failure model**

  \[
  \begin{align*}
  \dot{x} &= A x + B u + b_i n_i(t) \\
  y &= C x
  \end{align*}
  \]

- **sensor failure model**

  \[
  \begin{align*}
  \dot{x} &= Ax + Bu \\
  y &= C x + e_{m_i} n_c(t)
  \end{align*}
  \]

where $b_i$ and $e_{m_i}$ are two time-invarying vectors, even if the failures introduce time variations in the matrices.

**Example**

Consider the simple case below

\[
\begin{array}{c}
\mu \rightarrow \frac{k_1}{s^2 + 2\zeta\omega_n s + \omega_n^2} \rightarrow \ddot{z} \rightarrow k_2 \rightarrow y
\end{array}
\]

Equation

$$\frac{\ddot{z}}{\mu} = \frac{k_1}{s^2 + 2\zeta\omega_n s + \omega_n^2} \iff \ddot{z} + 2\zeta\omega_n \dot{z} + \omega_n^2 z = k_1 \mu$$

Using the phase variable $x_1 = z$, $x_2 = \dot{z}$, we have the state equations
\[
\begin{pmatrix}
\dot{x}_1 \\
\dot{x}_2
\end{pmatrix} =
\begin{pmatrix}
0 & 1 \\
-\omega_n^2 & -2\xi_\omega_n
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2
\end{pmatrix} +
\begin{pmatrix}
0 \\
1
\end{pmatrix} u
\]

\[
y = \begin{pmatrix}
k_2 \\
0
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2
\end{pmatrix}
\]

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

\[
\begin{align*}
\dot{x} &= A x + B u + \begin{pmatrix} 0 \\ 1 \end{pmatrix} (k_1(t) - 1) K_1 u \\
y &= C x
\end{align*}
\]

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

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

\[
\begin{align*}
\dot{x} &= A x + B u \\
y &= C x + (1) (K_2) [k_2(t) - 1] x_1
\end{align*}
\]

In this case \( e_{mi} = 1 \) \( n_{ci}(t) = K_2 [k_2(t) - 1] x_1 \)

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

- **controller failure model** \( \dot{\xi} = (A - DC) \xi + b_i n_i(t) \)

\[
\xi_i' = C \xi_i
\]

- **sensor failure model** \( \dot{\xi} = (A - DC) \xi - d_i n_{ci}(t) \)

\[
\xi_i' = C \xi_i + e_{mi} n_{ci}(t)
\]

where \( d_i \) is the \( i \)th 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 $b_i$ (controller failure model) can be constrained to be unidirectional
c - outputs associated with a $n_{ci}$ (sensor failure model) can be constrained to a plane.

More precisely: definition (Beard)

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

(1) $C \xi$ maintains a fixed direction in the output space, where $\xi(t)$ 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 $\dot{\xi} = (A - DC) \xi + b \, n(t)$ with $A - DC$ matrix negative definite is, after the vanishing of the transient terms

$$\xi(t) = \int_{\zeta_0}^{\xi} \exp \left[ - (A - DC)(t - \tau) \right] b \, n(\tau) \, d\tau$$

and this solution lies in the space spanned by the columns of

$W_b = [b, (A - DC) b, \ldots, (A - DC)^{n-1} b]$.

That is to say, $\xi$ may be expressed in the form

$$\xi(t) = W_b q(t)$$

Then

$$C \xi(t) = C W_b q(t)$$

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 \( \mathcal{N}(A) \).

- **detection equivalent events** (Jones)

  Two events \( b_1 \) and \( b_2 \) associated with failures in a system are said to be detection equivalent if
  
  (a) every detection filter for \( b_1 \) is a detection filter for \( b_2 \)
  
  (b) the unidirectional output error generated by the failure associated with \( b_2 \) is in the same output direction as that associated with \( b_1 \).

- **detection space of an event**

  The detection space of \( b_1 \) contains all events which are detection equivalent to \( b_1 \).

  (a) Vector space definition (Jones).

Let \( b_1 \) be an event vector associated with a failure. Assume that \( C b_1 \neq 0 \). Detection space for \( b_1 \) is denoted by \( \mathcal{R}_1 \) and is the direct sum
\[ \bar{R}_1 = b_1 \oplus R_1 \] where \( R_1 \subseteq \mathbb{R}^n \) is the largest subspace satisfying the three conditions

1. \( \mathcal{N}(M) \cap R_1 = \emptyset \)
2. \( R_1 \subseteq \mathcal{N}(C) \)
3. \( AR_1 \subseteq \bar{R}_1 \)

As \( b_1 \notin \mathcal{N}(M) \), condition (1) ensures that \( \bar{R}_1 = b_1 \oplus R_1 \) is an observable subspace for \((A,B,C)\). Condition (2) ensures that for every vector of \( \bar{R}_1 \),

\[ \bar{\xi} = \alpha b_1 + \xi \quad \text{where} \quad \xi \in R_1 \]

Then every vector of \( \bar{R}_1 \) has the same output direction as \( b_1 \).

(b) Matrix notation (Beard)

\[ \bar{R}_1 \text{ can be shown to be the null space of } \]

\[ M' = \begin{bmatrix}
E_m - \xi b_1 [c_{b_1}^T c_{b_1}] (c_{b_1})^T \\
E_m - \xi b_1 [c_{b_1}^T c_{b_1}] (c_{b_1})^T [A - A b_1 [c_{b_1}^T c_{b_1}] (c_{b_1})^T C]
\end{bmatrix} \]

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

Note: If \( C b_1 = 0 \), just replace \( b_1 \) by \( A^\nu b_1 \) where \( \nu \) is the smallest integer such that \( CA^\nu b_1 \) is not zero, in the above definition.

* detection generator

It is possible to show that the detection space \( \bar{R} b_1 \) is
cyclic invariant, and that there exists a unique vector \( g \) in \( \mathbb{R}_{b_1} \) such that the vectors 
\[
g, A g, \ldots, A^{\nu_{b_1}-1} g
\]
are a basis for \( \mathbb{R}_{b_1} \).

Vector definition (Jones)

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

1. \( A^k g \leq R_{b_1} \) \( \nu < \nu_{b_1} \cdot g \)
2. \( C A^{\nu_{b_1}-1} g = c b_1 \)

Beard showed that if the \( \nu_{b_1} \) eigenvalues of \( A - DC \) associated with the controllable subspace of \( b_1 \) are given by the roots of

\[
\nu_{b_1} + p_{b_1} \nu_{b_1-1} + \ldots + p_2 \nu + p_1 = 0
\]

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

\[
DCA^{\nu_{b_1-1}} g = p_1 g + p_2 A g + \ldots + p_{\nu_{b_2}} A^{\nu_{b_1-1}} g + A^{\nu_{b_1}} g
\]

Mutually detectable set of events (Jones)

Given the inhomogeneous error equations

\[
\dot{\xi} = (A - DC) \xi + b_i n_i(t) \quad i = 1, \ldots, r
\]

\[
\xi = c \xi
\]

The failures associated with the events \( b_1, \ldots, b_r \) are mutually
detectable by a single failure detection system if

1. the output generated by each of \(\vec{b}_1 \cdot n_1(t), \ldots, \vec{b}_r \cdot n_r(t)\)
   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 \(\vec{C}_{b_1}\). From a practical point of view, however, one case can be immediately examined: if two events \(b_1\) and \(b_2\) are such that \(\vec{C}_{b_1}\) is parallel to \(\vec{C}_{b_2}\), there are 2 possibilities:

1. If \(b_2 \in \mathcal{F}_1\), then \(b_2\) and \(b_1\) are detection equivalent, and a detection filter cannot distinguish between failures associated with these two events.
2. If \(b_2 \not\in \mathcal{F}_1\), it can be shown that if \(\vec{C}_{b_1} \parallel \vec{C}_{b_2}\) a failure detection system which detects failures associated with \(b_1\) cannot simultaneously detect failures associated with \(b_2\).

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 \(\vec{C}_{b_1}\) 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 \(\vec{b}_1, \ldots, \vec{b}_r\) are **output separable** if the rank of \([\vec{C}_{b_1}, \ldots, \vec{C}_{b_r}] = r\).

It can be shown that output separability is sufficient to
guarantee that a \( D \) can be found for which failures associated with \( b_1, \ldots, b_r \) produce unidirectional output errors. If the dimension of \( R_i \) is denoted by \( \gamma_i \), \( \sum_{i=1}^{r} \gamma_i \) 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 \( R_i \) can be almost arbitrarily assigned, when the \( b_i \) are output separable, but that some eigenvalues of \( A - DC \) are determined, and cannot be changed. More precisely:

1. Let \( S \) be the space spanned by \( \{b_1, \ldots, b_r\} \). We can define a detection space for \( S \), \( \overline{R}_s \), such that (Jones)

\[
\overline{R}_s = R_s \oplus S
\]

where \( R_s \) is the largest subspace which satisfies

1. \( \mathcal{M}(M) \cap R_s = 0 \)
2. \( R_s \subset \mathcal{N}(C) \)
3. \( AR_s \subset \overline{R}_s \)

\( \overline{R}_s \) can be shown to be the null space of

\[
D_s = AB[(CB)^T(CB)]^{-1}(CB)^T
\]

\[
C_s' = [E_m - CB[(CB)^T(CB)]^{-1}(CB)^T]C
\]

and \( \overline{E} \) is the matrix \( \{b_1, \ldots, b_r\} \).

\( \overline{R}_s \) is a direct extension of \( R_i \) defined formerly. In a sense, it is the set of events which are detection equivalent to the set of \( b_i \), \( i = 1, \ldots, r \). If the \( b_i \)'s are output separable, we have
Let us define $\mathcal{V}_s' = \mathcal{V}_1' + \ldots + \mathcal{V}_r'$ where $\mathcal{V}_i'$ is the dimension of $\overline{R}_i$.

If $D$ is chosen such that it makes outputs associated with $b_1,\ldots,b_r$ unidirectional, only $\mathcal{V}_s'$ of the eigenvalues associated with $\overline{R}_s$ can be almost arbitrarily assignable, the remaining $\mathcal{V}_s - \mathcal{V}_s'$ are unassignable. Therefore

$$\overline{R}_1,\ldots,\overline{R}_r \text{ are mutually detectable } \iff \sum_{i=1}^{\infty} \mathcal{V}_i = \mathcal{V}_s$$

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 $b_i$'s are not mutually detectable, and a detectable subset if these eigenvalues are not acceptable.

- **Excess subspace.** If $b_1,\ldots,b_r$ are not mutually detectable, the **excess subspace** of $\overline{R}_s$ is any subspace $R_o \subset \mathfrak{N}(C)$ which satisfies $\overline{R}_s = \overline{R}_1 \oplus \ldots \oplus \overline{R}_r \oplus R_o$.

In general, it is not unique. However, it can be shown that the $\mathcal{V}_o$ eigenvalues of $A-DC$ associated with $R_o$ (which are the $\mathcal{V}_o$ unassignable eigenvalues of $A-DC$ for a given set of $b_i$'s) are independent of the choice of $D$. The algorithm used will determine a basis of the unique subspace $R_{oq}$ defined by:
\[ \text{Rog excess subspace of } \tilde{R}_k \]
\[ \text{AR}_\text{log} \subset \text{Rog} \oplus g_1 \oplus \ldots \oplus g_r \]

It can be shown that \( \text{Rog} \) is the null space of the matrix

\[
M_c = \begin{bmatrix}
\bar{c}_1' \\
\vdots \\
\bar{c}_1'(A - D_\delta C) \\
\bar{c}_2' \\
\vdots \\
\bar{c}_2'(A - D_\delta C) \\
\vdots \\
\bar{c}_r' \\
\vdots \\
\bar{c}_r'(A - D_\delta C)
\end{bmatrix}
\]

- where the \( \bar{c}_i \)'s are the rows of the matrix \( ((CB)^T(CB))^{-1}(CB)^T \)
- \( D_\delta = A\bar{B}[(CB)^T(CB)]^{-1}(CB)^T \)
- \( \bar{B} \) matrix \([b_1, \ldots, b_r]\)

Let us call \( \text{Rog} \) this basis. Once it is determined, as \( \text{AR}_\text{log} \subset \text{Rog} \oplus g_1 \oplus \ldots \oplus g_r \), we have with \( G = [g_1, \ldots, g_r] \)

\[ \text{AR}_\text{og} = \text{Rog} \ \bar{T} + G \ \Theta \]

it can be shown that the rows of \( \Theta \) are equal to

\[ \Theta_i = \bar{c}_i'(A - D_\delta C) \text{Rog for } i = 1, \ldots, r \]  

(2-1)

As \( A, \text{Rog}, \Theta \) are known, \( \bar{T} \) can be computed, and the unassignable eigenvalues of \( A - DC \) associated with \( \bar{B} \) are the eigenvalues of \( \bar{T} \).

- Property of \( \text{Rog}_k \): suppose we have a set of events \( b_1, \ldots, b_r \) which are not mutually detectable for a system \( (A, C) \). If we define \( \text{Rog}_k = \text{excess subspace associated with} \)

\[ (b_1, \ldots, b_{k-1}, b_{k+1}, \ldots, b_r) \]

for \( k = 1, \ldots, r \), Jones showed that
(a) $R_{og_k} \subset R_{og}$ for all $k$

(b) The excess subspace associated with the set of events $b_i$ where $b_j$ and $b_k$ 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 $b_i$'s with acceptable unassignable eigenvalues:

- for each event $b_i$ it computes the set $\bigwedge_i$ of unassignable eigenvalues associated with $b_1, \ldots, b_{i-1}, b_{i+1}, \ldots, b_n$
- the designer knows that if he eliminates $b_i$, for $i \in J$ of the set of events, the unassignable eigenvalues of the set of events, $b_j, j \subset [1, \ldots, r] - J$ will be

$$\bigcap_{i \in J} \bigwedge_i$$

In particular, if $\bigcap_{i \in J} \bigwedge_i = \emptyset$, the remaining events once the events $b_i, i \notin J$, have been extracted are mutually detectable.

- Output stationarity (Reference 2)

Assume $b_1, \ldots, b_k$ are output separable, and let $D^*$ be the class of operators such that for every $D \in D^*$ the output generated by each of $b_1, \ldots, b_k$ with respect to $(A - DC, C)$ is unidirectional. The subspace $\mathcal{F}$ can be made output stationary with $b_1, \ldots, b_k$ if there exists a $D' \in D^*$ such that the output from every element $\sum_{i} e_i \in \mathcal{F}$ for $(A - D'C, C)$ is unidirectional along $C \left\{ e_i \right\}$ (or $CA \left\{ e_i \right\}$ if $C \left\{ e_i \right\} = 0$, etc).

The cost of using output stationarity to increase the number
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.

Suppose we want to make $h_i$ output stationary with $b_1, \ldots, b_r$ where $b_1, \ldots, b_r$ is a set of output separable vectors, and $b_1, \ldots, b_r, h_i$ are not output separable. This implies that $C h_i$ is a linear combination of $C b_1, \ldots, C b_r$. Suppose that $b_1, \ldots, b_\ell$ is the smallest subset of $b_1, \ldots, b_r$ such that

$$ C F_L^C \alpha_L^C = \beta_i C h_i $$

has a solution where

$$ \alpha_L^C = [\alpha_1', \ldots, \alpha_\ell']^T $$

$$ F_L^C = [b_1, \ldots, b_\ell] $$

and $\alpha_1', \ldots, \alpha_\ell'$ are a set of nonzero coefficients. If $b_1, \ldots, b_\ell$ are mutually detectable, it can be demonstrated that $h_i$ can be made output stationary with $b_1, \ldots, b_r$ if there exists a solution to $R_L^C = \overline{S}_i$ where

$$ R_L^C = [R_1', \ldots, R_\ell'] $$

and where $\overline{S}_i = [h_i : S_i]$, detection space of $h_i$. 


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 $\mathbb{R}_f$ associated with the whole set of events, and the detection space $\mathbb{R}_i$ associated with each $b_i$. In the process, it computes the detection generator $g_i$ associated with each $\mathbb{R}_i$.

If $\sum_i k(\mathbb{R}_i) = \leq k(\mathbb{R}_f)$, 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 $\sum_i k(\mathbb{R}_i) > \sum_i k(\mathbb{R}_f)$, there are $k(\mathbb{R}_f) - \sum_i k(\mathbb{R}_i)$ 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 $b_i$'s with no unassignable eigenvalues.

To do this, the program computes for each $b_i$ the set $\mathbb{R}_i$ of unassignable eigenvalues associated with the events $(b_1, \ldots, b_{i-1}, b_{i+1}, \ldots, b_r)$. The designer then takes out the events $b_j$, $j \in I$, $I$ a set of indices, $(I \subset [1, \ldots, r])$ such that
\( \bigwedge_{j \in I} \bigwedge_j^j \) contains only acceptable unassignable eigenvalues. If, for example, one of the \( \bigwedge_i \)'s, assume \( \bigwedge_1 \), is the null set, the designers know that the events \( b_2, \ldots, b_x \) are mutually detectable.

\( \circ \) 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 base normal canonical form (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

\[
T^{-1} = \begin{bmatrix}
q_1, \cdots, q_{\gamma_i}, q_{\gamma_i+1}, \cdots, q_{\gamma_i+\gamma_i-1}, \cdots, q_{\gamma_i+\gamma_i-1}, T_0
\end{bmatrix}
\tag{3-1}
\]

where the \( q_i \)'s are the detection generators of the \( \bar{R}_i \)'s

\( \gamma_i \) is the dimension of \( \bar{R}_i \)

\( T_0 \) is a matrix such that \( T \) is nonsingular. The choice made for \( T_0 \) is:

\[
T_0 = \begin{bmatrix}
\bar{z}_1, \cdots, \bar{z}_{\gamma_0}, (A-D C)^{q_{\gamma_0+1}-1} W_{\gamma_0+1}, \cdots, (A-D C)^{q_m-1} W_m
\end{bmatrix}
\]

where \( \bar{z}_1, \ldots, \bar{z}_{\gamma_0} \) is a basis for \( R_{0\gamma} \) (If the events are mutually detectable, \( R_{0\gamma} \) has dimension 0).
The \( w_{r+1}, \ldots, w_m \) are chosen so that the vectors
\[
\{ w_{r+1}, \ldots, (A - D_s \beta) \}^{q_{r+1} - 1} \cup \{ (A - D_s \beta) \}^{q_{r+1} - 1} \cup \{ w_1, \ldots, w_m \}
\]
complete the set \( \{ q_{r+1}, \ldots, A q_{r+1}, \ldots, A q_{r+1}, \ldots, q_{r+1} \} \cup \{ q_{r+1}, \ldots, q_{r+1} \} \) to form a basis in \( \mathbb{R}^n \). If \( \mathbb{R}_s \) is already of dimension \( n \), where \( n \) is the state space dimension, there is no need for the vectors \( w_{r+1}, \ldots, w_m \). The set
\[
\{ q_{r+1}, \ldots, A q_{r+1}, \ldots, A q_{r+1}, \ldots, q_{r+1} \}
\]
a basis of \( \mathbb{R}_s \), is also a basis for \( \mathbb{R}^n \).

More precisely, the \( w_i \) are the auxiliary vectors associated with \( C'_i(A - D \beta C) \) in the orthogonal reduction of

\[
M_c = \begin{bmatrix}
C'_s \\
C'_s(A - D_s \beta) \\
\vdots \\
C'_s(A - D_s \beta)^{n-1}
\end{bmatrix}
\]

starting with the identity matrix

where \( C_i' \) is a row of \( C'_s \)

\[
C'_s = \left[ \mathcal{E}_m - \mathcal{B} \left( \mathcal{B}^T \mathcal{B} \right)^{-1} \mathcal{B}^T \right] \mathcal{C}
\]

\[
D_s = \mathcal{B} \left( \mathcal{B}^T \mathcal{B} \right)^{-1} \mathcal{B}^T
\]

\( q_i \) is the largest integer such that all of \( C'_i, \ldots, C'_i(A - D \beta C) \) 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 \( T^{-1} \).

\[
T^{-1} = \begin{bmatrix}
C_A q_{r+1} \ldots, C_A q_{r+1} \ldots, C_s(A - D_s \beta) q_{r+1} \ldots, C_s(A - D_s \beta) q_{r+1} \ldots, w_{r+1} \ldots, w_m \n\end{bmatrix}
\]

\( T^{-1} \) transforms the state vector to base normal form and \( T \) transforms it back to the original coordinate system. We have the relations
This transformation is used because the design of the detection filter in the canonical basis is straightforward: due to the two relations

\[ \rho < A \begin{pmatrix} q_{i}^{-1} \\ g_{i}^{-1} \end{pmatrix} = \rho_{i} q_{i} + \rho_{i} A q_{i} + \cdots + \rho_{i} A^{k-1} q_{i} + \rho_{i} A^{k} q_{i} \]  

(3.2)

where the \( \rho_{i} \) are the coefficients of the polynomial

\[ \prod_{i=0}^{k-1} (x - \lambda_{i}) \]

of the eigenvalues of

\[ R_{i} \begin{pmatrix} q_{i} \\ g_{i} \end{pmatrix} + \rho_{i} q_{i} + \cdots + \rho_{i} A^{k-1} q_{i} + \rho_{i} A^{k} q_{i} \]

and

\[ A R_{og} = R_{og} T + G \Theta \]

we have:

(3.3)

\[ \hat{A} \]

is of the form

\[ \hat{A} = \begin{bmatrix} \hat{A}_{11} & \hat{A}_{12} & \cdots & \hat{A}_{1n} & \hat{A}_{1n+1} \\ \hat{A}_{21} & \hat{A}_{22} & \cdots & \hat{A}_{2n} & \hat{A}_{2n+1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \hat{A}_{n1} & \hat{A}_{n2} & \cdots & \hat{A}_{nn} & \hat{A}_{nn+1} \\ \hat{A}_{n+1} & \hat{A}_{n+2} & \cdots & \hat{A}_{n+n} & \hat{A}_{n+n+1} \end{bmatrix} \]

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

\[ T \]

is a \( \mathcal{V}_{0} \times \mathcal{V}_{0} \) matrix associated with \( R_{og} \), given by rela-
tion (3-3)

\[ \hat{A}_{ij} = \begin{bmatrix} 0 & \cdots & 0 & \hat{A}_{ij} \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 0 \end{bmatrix} \text{ is a } \mathcal{V}_i \cdot \mathcal{V}_j \text{ matrix, for } i \neq j \]

\[ \hat{A}_{ii} = \begin{bmatrix} 1 & \cdots & 0 \\ 0 & \ddots & \vdots \\ 0 & \cdots & 1 - \hat{\rho}_i \end{bmatrix} \text{ is a } \mathcal{V}_i \cdot \mathcal{V}_i \text{ matrix} \quad (3-4) \]

\[ \hat{C}_{rr} \text{ is } (n - \mathcal{V}_A) \times (n - \mathcal{V}_A) \]

\[ \hat{\Gamma}_i' \text{ is } \mathcal{V}_i \times (n - \mathcal{V}_A) \]

\[ \hat{\Gamma}_o' \text{ is } \mathcal{V}_o \times (n - \mathcal{V}_A) \]

\( \hat{C} \) is of the form

\[ \hat{C} = \begin{bmatrix} \hat{C}_1 & 0 & \cdots & 0 \\ 0 & \hat{C}_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \hat{C}_m \end{bmatrix} \]

where \( \hat{\xi}_i = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \) is a \( \mathcal{V}_i \) vector

In order for \( D \) to be a detection filter for \( [b_1, \ldots, b_r] \), each space \( \overline{R}_i \) has to be invariant for \( \hat{A} - \hat{D}\hat{C} \), or, equivalently, for \( \hat{A} - \hat{D}\hat{C} \). If the \( \hat{\rho}_{ik} \) are the coefficients of the desired set of eigenvalues of \( \overline{R}_i \), \( \hat{A} - \hat{D}\hat{C} \) is equal to

\[ \hat{A} - \hat{D}\hat{C} = \begin{bmatrix} \hat{A}_{ii} & 0 & \cdots & 0 \\ 0 & \hat{A}_{i2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \hat{A}_{in} \end{bmatrix} \]

where \( \hat{A}_{ii} = \begin{bmatrix} 1 & \cdots & 0 \\ 0 & \ddots & \vdots \\ 0 & \cdots & 1 - \hat{\rho}_i \end{bmatrix} \)

If a \( \hat{D} \) is selected so that \( \hat{A} - \hat{D}\hat{C} \) is of this form, outputs associated with each \( b_i \) will be unidirectional and the eigenvalues
associated with each $\bar{R}_i$ can be chosen (if the dimension of $\bar{R}_d$ is less than $n$, the state space dimension, $n - \nu'$ 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 $\hat{A} - \hat{D}C$ in the desired form, $\hat{D}$ is selected to be the sum of two terms

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

with

$$\hat{D}_F = \begin{bmatrix}
0 & \hat{a}_{12} & \cdots & \hat{a}_{1r} & 0 & \cdots & 0 \\
0 & 0 & \cdots & \hat{a}_{2r} & 0 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots & \ddots & \ddots & \vdots \\
0 & \hat{a}_{r1} & \cdots & 0 & \hat{a}_{rr} & \cdots & 0 \\
0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\
0 & 0 & \cdots & 0 & 0 & \cdots & 0
\end{bmatrix}$$

and

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

with

$$\hat{d}_\psi_i = \begin{bmatrix}
-\hat{p}_i' + \hat{p}_i \\
-\hat{p}_i' + \hat{p}_i \\
\vdots \\
-\hat{p}_i' + \hat{p}_i
\end{bmatrix} \quad i = 1, \ldots, r$$

with the $\hat{p}_i', \ldots, \hat{p}_i$ 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, $b_i$ dimensions $A(n,n)$, $B(n,q)$, $C(m,n)$, $\overline{B}(n,r)$

Step 2. Compute rank of C.

Step 3. If there are more than rank C events to be detected, divide the $b_i$ 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 $b_1, \ldots, b_r$ compute $C b_1, \ldots, C b_r$ 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, $\overline{R}_1, \overline{R}_2, \ldots, \overline{R}_r$ will be computed, as well as $\overline{g}_1, \ldots, \overline{g}_r$.

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

5.b Compute

$$D_{\delta} = A \overline{B} [(C \overline{\delta}) (C \overline{\delta})^T] (C \overline{\delta})^T$$

$$\overline{\zeta} = \left[ E_m - C \overline{\delta} [(C \overline{\delta})^T (C \overline{\delta})] (C \overline{\delta})^T \right] C$$

Compute

$$M_{\delta} = \begin{bmatrix} \overline{\zeta} \\ \overline{\zeta} (A - D_{\delta} C) \\ \overline{\zeta} (A - D_{\delta} C)^n \end{bmatrix}$$

Apply orthogonal reduction to $M_{\delta}$ starting with $\overline{J}_c = \overline{J}_c$. 
Let $\mathcal{J}_d$ be the terminating matrix, $\mathcal{Y}_d = r_k(\mathcal{J}_d)$.

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

5.c For each $b_i$, $i = 1, r$

Compute

$$P_{b_i} = A_{b_i} \left[ (c_{b_i}) (c_{b_i}) \right] (c_{b_i})^T$$

$$c' = \left[ E_m - (c_{b_i}) (c_{b_i}) (c_{b_i}) (c_{b_i})^T \right] c$$

Reduce

$$M_{c'} = \left[ \begin{array}{c}
  c' (A - D_{b_i} c') \\
  \vdots \\
  c' (A - D_{b_i} c')^{n-1}
\end{array} \right]$$

starting with $\mathcal{J}_d = \mathcal{J}_d$

Let $\mathcal{J}_i$ be the terminating matrix $r_k(\mathcal{J}_i) = \mathcal{Y}_i$

5.d Compute

$$M = \left[ \begin{array}{c}
  c \\
  cA \\
  \vdots \\
  cA^{n-1}
\end{array} \right]$$

Apply orthogonal reduction to $M$, starting with each $\mathcal{J}_i$, $i = 1, \ldots, r$. Each reduction ends on a zero matrix. The last vector to be removed from the range space of $\mathcal{J}_i$ is a multiple of the detection generator $g_i$. 
5.e Compute \( \nu_s = \nu_i^* + r \) \( \nu_s = \text{dimension of } \overline{R}_s \)

\[ \nu_i = \nu_i^* + 1 \quad \text{for all } i = 1, \ldots, r. \quad \nu_i = \text{dimension of } \overline{R}_i. \]

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 \( \nu_o \) eigenvalues of \( A - D_s C \) are unassignable; they will be computed.

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

(partly done in step 5.b)

- Compute

\[
M_o = \begin{bmatrix}
\overline{c}_1 (A - D_s C) \\
\vdots \\
\overline{c}_r (A - D_s C)^{\nu_i - 1} \\
\vdots \\
\overline{c}_{\nu} (A - D_s C)^{\nu_{\nu} - 1}
\end{bmatrix}
\]

- Apply orthogonal reduction to \( M_o \) starting with \( \overline{S}_s \). Let \( \overline{S}_{og} \) be the terminating matrix \( \mathcal{N}(\overline{S}_{og}) = \nu_s \)

- Define \( R_{og} = [\overline{s}_1, \ldots, \overline{s}_{\nu}] \) where the \( \overline{s}_i \)'s are \( \nu_o \) linearly independent columns of \( \overline{S}_{og} \)

- Compute \( \Theta_i = \overline{c}_i (A - D_s C) R_{og} \) for \( i = 1, \ldots, r \)

Form matrix \( \Theta \) whose rows are the \( \Theta_i \)'s

- Compute \( \overline{\theta} = [R_{og}^T R_{og}]^{-1} R_{og}^T \begin{bmatrix} A & 0 \end{bmatrix} \overline{C} \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} 0 & \theta \end{bmatrix} \) where \( \begin{bmatrix} 0 & \theta \end{bmatrix} \) matrix of the generators.

- Compute eigenvalues of \( \overline{\theta} \) which are the unassignable eigenvalues associated with the set of events.
5.g Three options open:
- accept the eigenvalues of \( \Pi \); go to step 6
- find a subset of \( b_1, \ldots, b_r \) 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 \( b_i \)).

For each \( i, i = 1, r \) compute

\[
G_i = G \text{ with the } i\text{th column deleted (} G \text{ defined in step 5f)}
\]
\[
\Theta_i = \Theta \text{ with the } i\text{th row deleted (} \Theta \text{ defined in step 5f)}
\]
\[
\Theta_i^c = \text{ith row of } \Theta
\]
\[
M_{\Theta_i} = \begin{bmatrix}
\Theta_i^c \\
\Theta_i^c \Pi \\
\vdots \\
\Theta_i^c \Pi \_ i \_ - 1
\end{bmatrix}
\]

Apply orthogonal reduction to \( M_{\Theta_i} \). Let \( \beta_i' \) be the terminating matrix. Find a matrix \( \beta_i \) whose columns span the column space of \( \beta_i' \). Find a matrix \( \Delta \) whose columns span the row space of \( M_{\Theta_i} \). Form \([\Delta \quad \beta_i]_i\)

Compute

\[
\Pi_\beta = \left[ \Delta \quad \beta_i \right]_i \Pi \left[ \Delta \quad \beta_i \right]_i = \begin{bmatrix}
\Pi_\beta \\
\Pi_\beta \\
\vdots \\
\Pi_\beta
\end{bmatrix}
\]

Find eigenvalues of \( \Pi_\beta \). Let \( \Lambda_i \) be this set. \( \Lambda_i \) is the set of unassignable eigenvalues associated with \([b_1, \ldots, b_{i-1}, b_{i+1}, \ldots, b_r]\).

5.i Test for output stationarity: determine whether it is possible to make an event \( b_a \) output stationary with \( b_1, \ldots, b_r \).
compute $S_a$, the detection space associated with $b_a$ (same computation as in 5.c except that orthogonal reduction of $M_D$ associated with $b_a$ starts with identity matrix)

find the subset of $b_1, ..., b_r$ of which $b_a$ is a linear combination. Call $J$ the set of indices of $b_i$ such that

$$b_a = \sum_{k \in J} \sum_{k} b_k \quad J \subset [1, ..., r]$$

Form matrix $R_r = [.. R_k ..]$ for $k \in J$. Output stationarity will be possible if there exists some $\Xi$ such that

$$\Xi = S_a$$

If $\lambda_a = r_k(\Xi_a), \lambda_a$ eigenvalues associated with each $R_i, i \in J$, are equal to the eigenvalues selected for $R_a$. If, for some $i \in J, r_k(\Xi_i) = \lambda_i > \lambda_a$, $\lambda_i - \lambda_a$ eigenvalues of $R_a$ 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 $\{b_i\}_{i=1}^r$ be the set of events at this point.

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

$$T_0 = \left[ R_{og}, \omega_{n+1}, \ldots, (A-D_3 C)^{q_{r+1}}, \omega_{n+1}, \ldots, (A-D_3 C)^{q_{n+1}}, \omega_{n+1} \right]$$

6.b Form the matrix
\[ T = \begin{bmatrix} \frac{g_1}{A_1} & \frac{g_2}{A_2} & \cdots & \frac{g_n}{A_n} \end{bmatrix} \]

Compute \( T^{-1} \)

Form \( T_m = \begin{bmatrix} \frac{g_1}{A_1} & \frac{g_2}{A_2} & \cdots & \frac{g_{n+1}}{A_{n+1}} & \cdots & \frac{g_m}{A_m} \end{bmatrix} \)

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

\[ \hat{A} = \begin{bmatrix} \hat{A}_{11} & \hat{A}_{12} & \cdots & \hat{A}_{1s} \\ \hat{A}_{21} & \hat{A}_{22} & \cdots & \hat{A}_{2s} \\ \vdots & \vdots & \ddots & \vdots \\ \hat{A}_{r1} & \hat{A}_{r2} & \cdots & \hat{A}_{rr} \end{bmatrix} \]

with \( \hat{A}_{rr} \) a \( V_0 \cdot V_0 \) matrix

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

\[ \hat{A}_{ij} = \begin{bmatrix} \theta_i \\ 0 \end{bmatrix} \] \( V_1 \cdot V_j \) matrix for \( i \neq j \)

\[ \hat{A}_{ii} = \begin{bmatrix} 1 \\ 0 \\ \cdots \\ 0 \\ 1 - \rho_i \end{bmatrix} \] \( V_1 \cdot V_i \) matrix
6.d Compute

\[ \hat{D}_{FR} = \begin{bmatrix} 0 & \hat{a}_{11} & \cdots & \hat{a}_{1r} & 0 & 0 \\ \hat{a}_{21} & 0 & \cdots & \hat{a}_{2r} & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ \hat{a}_{r1} & \hat{a}_{r2} & \cdots & 0 & \hat{a}_{rr} & 0 \end{bmatrix} \]

- Enter the desired \( \lambda_1, \ldots, \lambda_r \) (eigenvalues for \( \bar{R}_i \)), \( i = 1, r \).

Compute

\[ \psi_i(\lambda) = (\lambda - \lambda_{i+1}) \cdots (\lambda - \lambda_r) \]
\[ = \lambda_i + \rho_i \psi_i \lambda + \cdots + \rho_r \psi_i \lambda + \rho_i \]

Compute

\[ d_{\psi_i} = \begin{bmatrix} -\rho_i + \rho_i \lambda_i \\ \vdots \\ -\rho_r \psi_i + \rho_i \psi_i \end{bmatrix} \]

Compute

\[ \hat{\varphi} = \begin{bmatrix} d_{\psi_1} & 0 & \cdots & 0 \\ 0 & d_{\psi_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & d_{\psi_r} \end{bmatrix} \]

- Compute \( \hat{D} = \hat{D} \varphi + \hat{D}_{FR} \)

- 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.

\[
A = \begin{bmatrix}
0 & 1 & 0 & 0 & 0 \\
0 & -1 & 0 & 6 & 7 \\
0 & -1 & 0 & 6 & 7 \\
-3 & c & -4 & -8 & 0 \\
-15 & 0 & -2 & 0 & -9
\end{bmatrix}
\]

\[
B = \begin{bmatrix}
0 \\
0 \\
3 \\
1.5
\end{bmatrix}
\]

\[
C = \begin{bmatrix}
-1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 1
\end{bmatrix}
\]

\[
\mathbf{b}_1 = \begin{bmatrix}
1 \\
c \\
c
\end{bmatrix}
\]

\[
\mathbf{b}_2 = \begin{bmatrix}
0 \\
1 \\
c
\end{bmatrix}
\]

Running the program, we find:

- Dimension of \( R_\phi \) is 1
- Dimension of \( R_1 \) is 0
- Dimension of \( R_2 \) is 0

Then, dimension of \( \overline{R}_\phi \) is 3 (as \( \overline{R} = [b_1 : b_2] \oplus R_\phi \))

- Dimension of \( \overline{R}_1 \) is 1 (as \( \overline{R}_1 = b_1 \oplus R_1 \))
- Dimension of \( \overline{R}_2 \) is 1 (as \( \overline{R}_2 = b_2 \oplus R_2 \))
we have

\[ y_4 > y_1 + y_2 \quad \text{and} \quad y_3 - (y_1 + y_2) = 1 \]

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

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

\[
\begin{align*}
\begin{pmatrix}
c & 0 \\
0 & 1 \\
0 & 0 \\
0 & 0
\end{pmatrix}
\begin{pmatrix}
c \\
o \\
1 \\
0
\end{pmatrix}
\end{align*}
\]

Then

\[
T_c = \begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1
\end{pmatrix}
\]

\[
\hat{A} = \begin{pmatrix}
c & 0 & 0 & 0 & 0 \\
c & 0 & 0 & 0 & 0 \\
c & 0 & 0 & 0 & 0 \\
c & 0 & 0 & 0 & 0 \\
c & 0 & 0 & 0 & 0
\end{pmatrix}
\]
Then

$$\hat{\mathbf{p}}_{FR} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ -3 & -4 & 0 & 0 \\ -15 & -2 & 0 & 0 \end{bmatrix}$$

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

$$\hat{\mathbf{p}}_4 = \begin{bmatrix} 8 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Then

$$\hat{\mathbf{p}} = \begin{bmatrix} 8 & 0 & 0 & 0 \\ 0 & 9 & 0 & 0 \\ -3 & -4 & 0 & 0 \\ -15 & -2 & 0 & 0 \end{bmatrix} \quad \mathbf{y} = \begin{bmatrix} 8 & 0 & 0 & 0 \\ 0 & 9 & 0 & 0 \\ -3 & -4 & 0 & 0 \\ -15 & -2 & 0 & 0 \end{bmatrix}$$

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 $(A - DC) \mathbf{q}_1 = -8 \mathbf{q}_1$

$(A - DC) \mathbf{q}_2 = -9 \mathbf{q}_2$
Example 2.

Same matrices $A$, $B$, $C$

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 7 \\ 0 & -1 & 0 & 0 & 7 \\ -3 & 0 & -4 & 0 & 0 \\ -1.5 & 0 & -2 & 0 & -2 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 3 \\ 1.5 \end{bmatrix}$$

$$C = \begin{bmatrix} -1 & c & c & c & c \\ c & c & 1 & c & c \\ c & c & 1 & c & c \\ c & c & c & c & 1 \end{bmatrix}$$

But, we shall use the events

$$b_1 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \quad b_2 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \quad b_3 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ c \end{bmatrix} \quad b_4 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

Running the program, we find

- dimension of $R_2$ 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
Then dimension of $\bar{R}_j = 5$
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

$$\bar{\gamma}_j > \bar{\gamma}_1 + \bar{\gamma}_2 + \bar{\gamma}_3 + \bar{\gamma}_4$$

and

$$\bar{\gamma}_j - (\bar{\gamma}_1 + \bar{\gamma}_2 + \bar{\gamma}_3 + \bar{\gamma}_4) = 1$$

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

The designer accepts this eigenvalue and goes on. There is no $w_1$ in this case ($\dim \bar{R}_j = 5 = \dim$ of state space).

$$T_c = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad T_0 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$
\[
\hat{A} = \begin{bmatrix}
-\delta & 0 & -4 & -3 & 0 \\
0 & 0 & -2 & -1.5 & 0 \\
6 & 7 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

Then

\[
\hat{J}_{\mathcal{F}_R} = \begin{bmatrix}
0 & 0 & -4 & -3 \\
0 & 0 & -2 & -1.5 \\
6 & 7 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0
\end{bmatrix}
\]

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

\[
\hat{J}_{\mathcal{F}_T} = \begin{bmatrix}
6.2 & 0 & 0 & 0 \\
0 & 7.1 & 0 & 0 \\
0 & 0 & 8 & 0 \\
0 & 0 & 0 & 10
\end{bmatrix}
\]

Then
And
\[ A = \begin{bmatrix}
10 & 1 & 0 & 0 \\
0 & 8 & 6 & 7 \\
0 & 8 & 6 & 7 \\
-3 & -4 & 6.2 & 0 \\
-1.5 & -2 & 0 & 7.1
\end{bmatrix} \]

As a further check
\[ (A - DC) \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix} = -7 \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix} \]
\[ (A - DC) \begin{bmatrix} 2 \\ 2 \\ 3 \\ 4 \end{bmatrix} = -8 \begin{bmatrix} 2 \\ 2 \\ 3 \\ 4 \end{bmatrix} \]
\[ (A - DC) \begin{bmatrix} 3 \\ 2 \\ 3 \\ 4 \end{bmatrix} = -9 \begin{bmatrix} 3 \\ 2 \\ 3 \\ 4 \end{bmatrix} \]
\[ (A - DC) \begin{bmatrix} 4 \\ 2 \\ 3 \\ 4 \end{bmatrix} = -10 \begin{bmatrix} 4 \\ 2 \\ 3 \\ 4 \end{bmatrix} \]

If the designer had decided not to accept the unassignable eigenvalue, the program can help him in selecting the subset of $b_i$'s, in computing the $\wedge_i$ associated with each $b_i$. In this case
\[ \wedge_1 = \begin{bmatrix} 0 \end{bmatrix} \quad \wedge_2 = \begin{bmatrix} 0 \end{bmatrix} \quad \wedge_3 = \emptyset \quad \wedge_4 = \emptyset \]
which means that

- with the set \( \{b_2, b_3, b_4\} \), the unassignable eigenvalue is 0
- with the set \( \{b_1, b_3, b_4\} \), the unassignable eigenvalue is 0
- with the set \( \{b_3, b_4\} \) the unassignable eigenvalue is 0,
- the sets of events \( \{b_1, b_2, b_3\}, \{b_1, b_2, b_4\}, \{b_1, b_3\}, \{b_1, b_4\}, \{b_2, b_3\}, \{b_2, b_4\}, \{b_1\}, \{b_2\}, \{b_3\}, \{b_4\} \)

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. 4.4 Dynamics of the velocity control loop
Vehicle dynamics

\[ V_{\text{ind}} = K_4 [x_v + n(t)] \]

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

\[ K_4 = 1.2 \]

where

\[ F_p = \text{propulsion system force} = T \]
\[ F_c = \text{Coulomb friction force} = -100 \cdot \text{sgn}(x_v) \text{lbs} \]
\[ = -f_c \text{sgn}(x_v) \]
\[ F_G = \text{grade force} \]
\[ = -mg \frac{\%\text{ grade}}{100} \text{ lbs, up to 6\%} \]
\[ F_A = \text{aerodynamic forces} \]
\[ = -(0.03)(x_v - v_w)(x_v - v_w) \text{ lbs} = -c_{\text{aero}} (x_v - v_w)(x_v - v_w) \]

with \( v_w \) = wind velocity in ft/sec

Compensator: proportional + integral

\[ K_1 + \frac{K_2}{s} \text{ with } K_1 = 1500 \quad K_2 = 1000 \quad \text{(Units lbs ft sec)} \]

Propulsion system

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

\[ \frac{T}{T_{\text{comm}}} = \frac{1}{(1 + \frac{s}{13.5})(1 + \frac{2(0.7)s}{30} + \frac{s^2}{30^2})} \]

We shall use the form

\[ T = a_2 T + a_3 T + a_4 T + K_3 T_{\text{comm}} \]

with

\[ a_2 = -55.5 \]
\[ a_3 = -1467 \]
\[ a_4 = -12150 \]
\[ K_3 = 12150 \]
(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: $m = 373$ $m_v = 350$ (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 ($\dot{z} = v_c - v_{\text{ind}}$)

The inputs are

- $a_c$: acceleration command
- $(x_V - V_w)/|\dot{x}_v - V_w|$ 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)
- $V_{\text{ind}}$

These were the only physically accessible signals

We have the relations

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

2. $(x_v) = \frac{T}{m} - \frac{f_c}{m} \frac{\dot{x}_v}{|\dot{x}_v|} - \frac{c_{\text{aero}}}{m} (x_V - V_w) \left| \dot{x}_v - V_w \right| - g \sin \theta$

where $\theta$ is the grade

3. $(T) = \dot{T}$

4. $(T) = \ddot{T}$

\[ \dddot{T} = a_2 \dddot{T} + a_3 \dddot{T} + a_4 T + k_3 T_c \]
But
\[ T_c = T_e + m_v a_c \]
\[ = K_1 \dot{z} + K_2 z + m_v a_c \]
\[ = K_1 (V_c - V_{ind}) + K_2 z + m_v a_c \]
\[ = K_1 (V_c - K_4 (x_v + n(t))) + K_2 z + m_v a_c \]

Then
\[ (5) \quad \dot{T} = -K_1 K_3 K_4 \dot{x}_v + a_4 T + a_3 \dot{T} + a_2 T + K_2 K_3 z + K_1 K_3 V_c + K_3 m_v a_c \]
\[ - K_1 K_3 K_4 n(t) \]
\[ (6) \quad \dot{z} = V_c - V_{ind} = V_c - K_4 \dot{x}_v - K_4 n(t) \]
\[ (7) \quad \dot{V}_c = a_c \]

In matrix form this gives
\[
\begin{pmatrix}
\dot{\chi}_v \\
\dot{\chi}_v \\
\dot{T} \\
\dot{T} \\
\dot{z} \\
\dot{V}_c \\
\end{pmatrix} =
\begin{pmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \frac{1}{m} & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & -K_1 K_3 K_4 a_4 & a_3 & a_2 & K_2 K_3 & K_2 K_3 & 0 \\
0 & -K_4 & 0 & 0 & 0 & 0 & 1 \\
\end{pmatrix}
\begin{pmatrix}
\chi_v \\
\chi_v \\
T \\
T \\
z \\
V_c \\
\end{pmatrix}
\]

\[
+ \begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
- \frac{g}{m} & 0 & -c \sin\theta & -1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & K_3 m_v & 0 & 0 & -K_1 K_3 K_4 & 0 & 0 \\
0 & 0 & 0 & 0 & -K_4 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 \\
\end{pmatrix}
\begin{pmatrix}
\dot{\chi}_v \\
1x_V \dot{v}_w \\
\dot{a}_c \\
(\dot{x}_v - v_w) | \dot{x}_v - v_w | \\
g \sin \theta \\
n(t) \\
\end{pmatrix}
\]
\[ T_c = -K_1 K_4 \dot{x}_v + K_2 z + K_4 V_c + m_v a_c - K_1 K_4 n(t) \]

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

In matrix form

\[
\begin{pmatrix}
T_c \\
V_{\text{ind}}
\end{pmatrix} =
\begin{pmatrix}
0 & -K_1 K_4 & 0 & 0 & 0 & K_2 & K_1 \\
0 & K_4 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
\dot{x}_v \\
\dot{V}_c \\
\dot{z} \\
\dot{x}_v - \dot{v}_w \\
\dot{z}/x_v \\
\dot{x}_v - \dot{v}_w
\end{pmatrix}
\]

Figure 4.5 shows the action of the velocity profiler on a wayside 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.
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

\[
\begin{align*}
\dot{X} & = A X + B \omega + C \tilde{y} + D y_m + \tilde{y}_m \\
\tilde{y}_m & = Q_y + \gamma
\end{align*}
\]

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 friction was modeled in the reference model as a bias).

The states selected are \( X_V, T, \dot{T}, \ddot{T}, T_e, V_c \) (vehicle velocity, thrust, first and second derivatives of thrust, compensator output and velocity command).

The outputs are \( T_c \) and \( V_{\text{ind}} \)

We have the relations
Fig. 4.7 Redundancies in velocity control loop
Fig. 4.8 Detection filter reference model
\[ T_c = T_c + m_v a_c \]
\[ T_e = K_1 (V_c - V_{\text{ind}}) + K_2 (V_c - V_{\text{ind}}) \]
\[ = K_1 (a_c - K_4 \dot{X}_v) + K_2 (V_c - V_{\text{ind}}) \]
\[ = K_1 a_c - K_4 T \frac{K_1 K_4}{m} + K_1 K_4 \frac{F_{\text{coul}}}{m} + K_2 V_c \]
\[
\dot{X}_v = \frac{T}{m} - \frac{F_{\text{coul}}}{m}
\]
\[ (T) = \dot{T} \]
\[ (\ddot{T}) = \ddot{T} \]
\[ (T) = K_3 T_c + a_2 \dot{T} + a_3 \dot{T} + a_4 T \]
\[ = K_3 T_e + K_3 m_v a_c + a_2 \dot{T} + a_3 \dot{T} + a_4 T \]
\[ (a_c) = a_c \]

In matrix form

\[
\begin{pmatrix}
\dot{X}_v \\
\dot{T} \\
\dddot{T} \\
\dddot{T}_e \\
\dot{V}_c
\end{pmatrix} =
\begin{pmatrix}
0 & \frac{1}{m} & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 \\
0 & a_2 & a_3 & a_4 & K_3 & 0 \\
-\frac{K_1 K_4}{m} & 0 & 0 & 0 & K_2 & \dot{T}_e \\
0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
\dot{X}_v \\
\dot{T} \\
\dddot{T} \\
\dddot{T}_e \\
\dot{V}_c
\end{pmatrix} +
\begin{pmatrix}
0 \\
0 \\
0 \\
0 \\
0 \\
-\frac{F_{\text{coul}}}{m}
\end{pmatrix}
\begin{pmatrix}
a_c \\
1
\end{pmatrix}
\]

\[
\begin{pmatrix}
\dot{T}_c \\
\dot{V}_{\text{ind}}
\end{pmatrix} =
\begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 1 & 0 \\
K_4 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
\dot{T}_c \\
\dot{V}_{\text{ind}}
\end{pmatrix} +
\begin{pmatrix}
m_v & 0 & 0 \\
0 & 0 & 0 \\
0 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
a_c \\
1
\end{pmatrix}
\]
This is not of the form
\[
\begin{align*}
\dot{x}_m &= A_m x_m + B_m u \\
y_m &= C x_m
\end{align*}
\]

There is a matrix $E_m$ such that
\[
\begin{align*}
\dot{x}_m &= A_m x_m + B_m u \\
y_m &= C_m x_m + E_m u
\end{align*}
\]
The system configuration, with the filter, is in fact

![Block diagram of system configuration](image)

We have then the equations
\[
\begin{align*}
\dot{x} &= A x + B u \\
y &= C x + E u \\
\dot{x}_m &= A_m x_m + B_m u + D(y - y_m) \\
y_m &= C_n x_m + E u
\end{align*}
\]

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

Then
\[
\dot{\hat{\xi}} - \dot{x}_m = \dot{\xi} - A \hat{x} + \beta \hat{u} - A x_m - B u - D(c x + E u - C x_m - E u)
\]

Then
\[
\dot{\hat{\xi}} = (A - DC) \hat{\xi}
\]
in the absence of failure. So long as the failures considered entail no variation in the $E$ matrix, the turns $DE \hat{u}$ and $DE_m \hat{u}$ simplify out in the equation of $\dot{\hat{\xi}}$, 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 $A = A_m \ B = B_m \ C = C_m$ 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 $C_n$'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, $K_1k_1(t)$ being its new value, the system equations become
\[
\begin{align*}
\dot{x} &= A\dot{x} + B\omega + \begin{pmatrix}
0 \\
0 \\
0
\end{pmatrix} (-\frac{TK_1^2}{m} + a_c + K_4 + \frac{F_{coul}}{m})(\dot{k}_1(t) - 1) K_1 \\
\gamma &= Cx + E\omega
\end{align*}
\]

The event associated with a failure in \( K_1 \) is then \( \xi_{65} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \)

We have \[ C_{-65} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \]

Event associated with a failure in \( K_2 \). If \( K_2 \) fails and becomes arbitrarily time-varying, its new value being \( K_2 k_2(t) \), the system equations become
\[
\begin{align*}
\dot{x} &= A\dot{x} + B\omega + \begin{pmatrix}
0 \\
0 \\
0
\end{pmatrix} (-K_4 \dot{k}_2 + a_c + \dot{k}_2) (\dot{k}_2(t) - 1) K_2 \\
\gamma &= Cx + E\omega
\end{align*}
\]

The event associated with a failure in \( K_2 \) is \( \xi_{65} \).

Event associated with a failure in \( K_3 \). If \( K_3 \) fails and becomes arbitrarily time-varying, its new value being \( K_3 k_3(t) \), the system equations become
\[
\begin{align*}
\dot{x} &= A\dot{x} + B\omega + \begin{pmatrix}
0 \\
0 \\
0
\end{pmatrix} (T_{c} + m a_c + \dot{k}_3) (\dot{k}_3(t) - 1) K_3 \\
\gamma &= Cx + E\omega
\end{align*}
\]

The event associated with a failure in \( K_3 \) is \( \xi_{64} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \)
In this case, we shall use the first vector of the series $A \mathbf{e}_{64}, A^2 \mathbf{e}_{64}, \ldots, A^n \mathbf{e}_{64}$ for which $C A^i \mathbf{e}_{64} \neq (0)$

\[
A \mathbf{e}_{64} = \begin{pmatrix}
0 \\
1 \\
a_2 \\
0 \\
o
\end{pmatrix}, \quad C A \mathbf{e}_{64} = (0)
\]

\[
A^2 \mathbf{e}_{64} = \begin{pmatrix}
0 \\
1 \\
a_2 \\
a_3 + a_2^2 \\
o
\end{pmatrix}, \quad C A^2 \mathbf{e}_{64} = (0)
\]

\[
A^3 \mathbf{e}_{64} = \begin{pmatrix}
1/m \\
a_2 \\
a_3 + a_2^2 \\
a_4 + 2a_3 a_2 + a_2^3 \\
-k_1 k_4/m \\
o
\end{pmatrix}, \quad C A^3 \mathbf{e}_{64} \neq (0)
\]

The event associated with $K_3$ is then
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

$$
\begin{align*}
\dot{\chi} &= A\chi + B u + \left(\begin{array}{c}
0 \\
0 \\
0 \\
C
\end{array}\right) \left[ K_2 \dot{x}_v - K_1 \frac{T}{m} + K_1 \frac{F_{\text{coul}}}{m} \right] (\xi_4(t) - 1) K_4
\end{align*}
$$

Let us call

$$
\xi_{65} = \left(\begin{array}{c}
0 \\
0 \\
0 \\
1
\end{array}\right) \quad \xi_{22} = \left(\begin{array}{c}
0 \\
0 \\
0 \\
1
\end{array}\right)
$$

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

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

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

$$
\begin{align*}
\dot{x} &= A x + B u + n_5(t) \xi_{65} \\
y &= C x + E u + n_{c2}(t) \xi_{22}
\end{align*}
$$

The error equations will then be

$$
\dot{\xi} = (A - D C) \xi + n_5(t) \xi_{65} - d_2 n_{c2}(t)
$$

$$
\dot{\xi} = C \xi + n_{c2}(t) \xi_{22}
$$

where $d_2$ is the 2nd column of the filter $D$. 
$e_{22}$ corresponds to a variation in $C$. Then (see ref 2 page 193-194), the error output of a failure associated with $e_{22}$ can only be contained to the plane spanned by $(C^F, CA^F)$ where $F$ is such that $C^F = e_{22}$.

If $e_{61} = \begin{pmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \end{pmatrix}$ we notice that $C^e_{61} = \begin{pmatrix} 0 \\ \kappa_4 \end{pmatrix}$.

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 $e_{65}$. The sensor failure will be represented by the couple of events $(e_{61}, A e_{61})$. However

$$A e_{61} = \begin{pmatrix} 0 \\ 0 \\ \kappa_2 \kappa_4 \end{pmatrix}$$

then

$A e_{61}$ is parallel to $e_{65}$.

In short, we have

- event associated with failure in $K_1$ is $e_{65} = b_1$
- event associated with failure in $K_2$ is $e_{65} = b_1$
- event associated with failure in \( K_3 \) is \( b_2 = \begin{pmatrix} 1/x_n \\ -a_2 \\ a_3 + a_2 \\ a_4 + 2a_3 a_2 + a_3 \\ -k_1 k_4 /m \\ 0 \end{pmatrix} \)

- events associated with failure in \( K_4 \) are \( e_{65} \) and \( e_{61} \)

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 \((C e_{65}^T, C e_{61}^T)\). \( C \) is of dimension 2; this means that a failure in \( K_4 \) will span the whole output space \((C e_{65}^T \) and \( C e_{61}^T \) are independent).

It was decided to design a detection filter for the events \( b_1 \) and \( b_2 \). A failure in \( K_1 \) or \( K_2 \) would generate outputs along \( Cb_1 \), a failure in \( K_3 \) along \( Cb_2 \), a failure in \( K_4 \) along \( Cb_1 \) and \( Cb_2 \). 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 \( Cb_1 \) and \( Cb_2 \): 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

\[
A = \begin{pmatrix}
0 & 2681 \times 10^{-2} & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 \\
0 & -12150 & -1466.5 & -55.5 & 12150 & 0 \\
-1200 & -482.6 & 0 & 0 & 0 & 1000 \\
0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
\]

\[
B = \begin{pmatrix}
0 & -0.268 \\
0 & 0 \\
0 & 0 \\
4.25 \times 10^5 & 0 \\
4500 & 482.6 \\
1 & 0
\end{pmatrix}
\]

\[
C = \begin{pmatrix}
0 & 0 & 0 & 0 & 0 & 1 & 0
\end{pmatrix}
\]

\[
E = \begin{pmatrix}
350 & 0 \\
0 & 0
\end{pmatrix}
\]

\[
L_1 = \begin{pmatrix}
0 \\
0 \\
0 \\
0 \\
0 \\
1
\end{pmatrix}
\]

\[
L_2 = \begin{pmatrix}
2681 \times 10^{-2} \\
-55.5 \\
-1613.75 \\
-28322.37 \\
-482.6 \\
0
\end{pmatrix}
\]
Using the computer program developed from the algorithm presented in Chapter 3, it was found that:

- \( R_\mathcal{F} \) has a dimension 4
- \( R_1 \) has a dimension 1
- \( R_2 \) has a dimension 3

\[
\begin{align*}
\mathcal{F}_1 & = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \\
\mathcal{F}_2 & = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}
\end{align*}
\]

Then \( \mathcal{R}_\mathcal{F} \) has a dimension 6
- \( \mathcal{R}_1 \) has a dimension 2
- \( \mathcal{R}_2 \) has a dimension 4

\( \mathcal{V}_1 + \mathcal{V}_2 = \mathcal{V}_3 \) The events \( b_1 \) and \( b_2 \) are mutually detectable.

We assigned the eigenvalues -10 and -10 to \( \mathcal{R}_1 \), -10, -10, -10, -10, -10 to \( \mathcal{R}_2 \)

The filter computed was

\[
D = \begin{pmatrix}
0 & -12.9166 \\
0 & -1942.937 \\
0 & 4639.965 \\
12150 & -193023.664 \\
20 & 52252.0625 \\
0.1 & 150.
\end{pmatrix}
\]
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 \( C_{b1} \) and \( C_{b2} \), where \( b_1 \) is the event associated with a failure in \( K_1 \) or \( K_2 \), and \( b_2 \) 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 \( Y - Y_m \) 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, \( C_{b1} \) has the value \( \begin{pmatrix} 2 \\ 0 \end{pmatrix} \) and \( C_{b2} = \begin{pmatrix} -4.826 \\ 3.217 \times 10^{-3} \end{pmatrix} \). It is necessary to change the basis of the output space to measure the error outputs along \( C_{b1} \) and \( C_{b2} \). The new basis \( \begin{pmatrix} 1 & 0 \\ 1.2 & 1 \end{pmatrix} \) has the components \( \begin{pmatrix} 1 & -4.826 \\ 0 & 3.217 \times 10^{-3} \end{pmatrix} \) along the old one. We have then \( x_{\text{new}} = P^{-1} x_{\text{old}} \) with

\[
P = \begin{pmatrix}
1 & -4.826 \\
0 & 3.217 \times 10^{-3}
\end{pmatrix}
\]

Hence \( P^{-1} = \begin{pmatrix}
1 & 1.506 \\
0 & 3.10^{-3}
\end{pmatrix}
\)

In the following plots, \( \xi_1 = [P^{-1}(Y - Y_m)]_1 \) is the error output along \( 1.2 C_{b1} \), \( \xi_2 = [P^{-1}(Y - Y_m)]_2 \) is the error output along \( C_{b2} \).

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
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 \( \xi_1 \) of less than 0.1 in absolute value is not significant, and that an output along \( \xi_2 \) 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 \( \xi_1 \) is not significant. There is only an output along \( \xi_2 \), 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
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 $\xi_1$ is not significant; there is an output along $\xi_2$ 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 $\xi_2$. At steady state, the error along $\xi_2$ 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 $\xi_1$ whose maximum value happened to be .68, and an output along $\xi_2$, 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 $\xi_2$ only, whose maximum value is roughly $6 \times 10^3$ times greater than the residue due to numerical precision loss. The only neglected effect in the reference model which produces an output along $\xi_1$ is the noise in the tachom-
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 $\xi_1$ was initialized to 3000; the error on $\xi_2$ 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 $R_1$ are -10 and -10, and the eigenvalues assigned with $R_2$ are -10, -10, -10, -10, the initial errors should decay in several tenths of a second. As can be seen from Fig. 5.5, $\xi_1$ decays to 5 per cent of its initial value within 0.2 sec and $\xi_2$ does within 0.7 sec. It appears that even if $x_m$ is not initialized in the reference model with the initial values of $x$, i.e., $x_m(0) \neq x(0)$, 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

<table>
<thead>
<tr>
<th>Time (sec)</th>
<th>Wayside velocity command (ft/sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 - 4</td>
<td>30</td>
</tr>
<tr>
<td>4 - 8</td>
<td>50</td>
</tr>
<tr>
<td>8 - 12</td>
<td>30</td>
</tr>
<tr>
<td>12 - 20</td>
<td>50</td>
</tr>
</tbody>
</table>
Error Decay: Initial Condition not Equal to Zero
No Failure
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 $E_1$ are roughly 10 times larger than residual outputs along $E_1$ without failures; failure outputs along $E_2$ are roughly 5 times larger than residual outputs along $E_2$ 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 $\xi_1$, failures in $K_3$ along $\xi_2$, failures in $K_4$ along $\xi_1$ and $\xi_2$. 
Fig. 5.11

Standard Test—Every Nonlinearity And Failure in $k_4$ at $T = 10s$

$k_4 = 120$

$k_4_{\text{old}} = 50$

$k_4_{\text{new}} = .50$
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 eigenvalues 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


APPENDIX A

LISTING OF DETECTION FILTER DESIGN PROGRAM
SUBROUTINE MAIN
COMMON/MAIN1/A,B,C,I31,N,P,O,R,IECR,EPS,BUFF
COMMON/MAIN3/IROW,ICOL
COMMON/MAIN4/OMEGC,DS,CS
DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12),CBI(12,12)
DIMENSION RUFF(12,12)
DIMENSION IROW(12),ICOL(12)
DIMENSION OMEGC(12,12),DS(12,12),CS(12,12)
INTEGER P,O,R
WRITE(6,100)
100 FORMAT(1X,'ERROR CODE COMPUTATION OF RANK OF C IS IER')
WRITE(6,101) IER
101 FORMAT(1X,16I16)
RETURN
FND
SUBROUTINE PROCBI(C,BI,CBI,N,P,R,IECR)
DIMENSION C(12,12),BI(12,12),CBI(12,12)
INTEGER P,O,R
DO 1 I=1,P
DO 1 J=1,R
C(I,J)=0.E0
1 CONTINUE
DO 2 I=1,P
DO 2 J=1,R
B(I,J)=BUFF(I,J)
2 CONTINUE
RETURN
END
SUBROUTINE PROCBI(C,BI,CBI,N,P,R,IECR)
DIMENSION C(12,12),BI(12,12),CBI(12,12)
INTEGER P,O,R
DO 1 I=1,P
DO 1 J=1,R
C(I,J)=CBI(I,J)+C(I,K)*BI(K,J)
1 CONTINUE
RETURN
END
WRITE(6,101)
101 FORMAT(1X,'THE PRODUCT OF C AND BI HAS BEEN COMPUTED AND:

* PUT IN MATRIX CBI(P,R)*)
RETURN
END
102 FORMAT(1X, 'IF SATISFIED, TYPE 11. OTHERWISE TYPE 00')
READ(5, 103) ISIGN
103 FORMAT(12)
   IF(ISIGN .EQ. 0) GO TO 1
RETURN
END
**SUBROUTINE MAIN2**

COMMON/MAN17/A,B,C,RI,CBI,N,P,Q,R,IECR,EPS,BUFF
COMMON/MAN13/IROW,ICOL
DIMENSION A(12,12),R(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 3I 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(A12)
IF(IECR GT 5) WRITE(6,1000) ISIGN1
STOP
1 WRITE(6,101)

101 FORMAT(1X,*TYPE THE NEW VALUE OF R,13*)
READ(5,1001) R
WRITE(6,102)

102 FORMAT(1X,*IF THE DATA ARE CORRECTLY ENTERED, TYPE 11, ELSE TYPE 01*)
READ(5,1000) ISIGN2
IF(IECR GT 5) WRITE(6,1000) ISIGN2
IF(ISIGN2 NE 0.0) GO TO 1
STOP
1 WRITE(6,101)

2 BUFF(I,J)=0.0
IF(ISIGN2 LE 0.0) GO TO 2
3 BUFF(I,J)=CBI(I,J)
CALL MFGR(BUFF,ICRIPRIRANKIROW,ICOLERS,IER)
WRITE(6,103) IER,RANK
103 FORMAT(1X,*ERROR CODE IN THE COMPUTATION OF CBI RANK IS ' ,13 ,/,
1* RANK OF CBI IS ' ,13 ,/)
WRITE(6,104) (IROW(I),I=1,N),(ICOL(I),I=1,N)
104 FORMAT(1X,*IROW=' ,13 ,/ICOL=' ,13 ,/)
WRITE(6,105)

105 FORMAT(1X,*IF THE RANK OF CBI MATRIX IS NOT EQUAL TO THE */
1*NUMBER OF EVENTS, YOU MUST CHANGE THE GROUP OF BI TO DO*/
3* THIS ENTER 00, ON THE CONTRARY, IF THE CBI ARE LINEARLY INDEPENDENT*/
4,/*DETECT ENTRY 11*/)
READ(5,1002) ISIGN3
1002 FORMAT(A12)
IF(IECR GT 5) WRITE(6,1002) ISIGN3
IF(ISIGN3 GT 0.0) 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/MANT2/A,B,C,RI,CBI,N,P,Q,R,IECR,EPS,BUFF
COMMON/MAN14/OMEGC,DS,CS
COMMON/MAN16/BUFF1
COMMON/TRASH2/CBVEC,CRVIV
DIMENSION A(12,12),B(12,12),C(12,12),HI(12,12),CBI(12,12)
DIMENSION BUFF(12,12),BUFF1(12,12)
DIMENSION OMEGC(12,12),CBVEC(78),CRVIV(78),DS(12,12),CS(12,12)
INTEGER P,Q,R
IF(IECR.GT.2)WRITE(6,1000)
1000 FORMAT(1X,"SUBROUTINE MAIN3")

**COMPUTATION OF OMEGC**

Eps1 = 1.E-9
Do 11 I=1,N
Do 12 J=1,N
10 OMEGC(I,J) = 0.0
11 OMEGC(I,I) = 1.E0
12 C = 12
CALL ORTRED(C,OMEGC,P,N,IECR,EPS1,IECR)
IF(IECR.GT.3)WRITE(6,100) ((OMEGC(I,J)*J=1,N),I=1,N)
100 FORMAT(4(1X,OMEGC=,E10.4))

**COMPUTATION OF CBIT*CBI**

Do 1 I=1,R
Do 2 J=1,P
1BUFF(I,J) = 0.0
2 I = 1,P
2 J = 1,R
2 K = 1,P
3 BUFF(I,J) = BUFF(I,J) + CRI(I)*CRI(K,J)
3 I = 1,R
3 J = 1,P
3 K = 1,P
4 BUFF(I,J) = BUFF(I,J) + BUFF1(I,J)
4 I = 1,R
4 J = 1,P
4 K = 1,R

**TRANSITION TO SYMMETRIC STORAGE**

I = 12
CALL VCVTSF(BUFF,R,1A,CRVEC)
IF(IECR.GT.5)WRITE(6,400) ((BUFF(I,J),J=1,R),I=1,R)
400 FORMAT(4(1X,BUFF=,E10.4))

**COMPUTATION OF INVERSE OF CBIT*CBI**

CALL LINVID(CBVEC,CRVIV,I102,IER)
IF(IECR.GT.5)WRITE(6,400) (CRVIV(I),I=1,R)
400 FORMAT(5(1X,CRVIV=,E10.4))

**TRANSITION TO FULL MODE (CBIT*CBI)-1=BUFF**

CALL VCVTSF(CRVEC,R,BUFF,1B)
IF(IECR.GT.5)WRITE(6,500) ((BUFF(I,J),J=1,P),I=1,N)
500 FORMAT(4(1X,BUFF=,E10.4))

**COMPUTATION OF BI*(CRIT*CBI)-1*CBIT=BUFF**

Do 3 I=1,R
3 I = 1,P
3 J = 1,R
3 K = 1,R
4 BUFF(I,J) = BUFF(I,J) + CBI(J,K)*BUFF1(I,J)
4 I = 1,N
4 J = 1,R
4 K = 1,R
5 BUFF(I,J) = BUFF(I,J) + CBI(J,K)*BUFF1(I,J)
5 I = 1,N
5 J = 1,P
5 K = 1,R
6 BUFF(I,J) = BUFF(I,J) + CBI(J,K)*BUFF1(I,J)
6 I = 1,N
6 J = 1,P
6 K = 1,R

**COMPUTATION OF CBIT*CBI**

Do 7 I=1,N
7 I = 1,P
7 J = 1,N
7 K = 1,P
8 BUFF(I,J) = BUFF(I,J) + CBI(J,K)*BUFF1(I,J)
8 I = 1,N
8 J = 1,P
8 K = 1,R

**COMPUTATION OF BI*(CRIT*CBI)-1*CBIT IN COMMON MAN16**

IF(IECR.GT.5)WRITE(6,600) ((BUFF(I,J),J=1,P),I=1,R)
600 FORMAT(4(1X,BUFF=,E10.4))
On 6 J=1,P
On 6 K=1,P
6 BiFF(I,J)=3I(I,K)*BUFF(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,'BUFF=',E10.4))

C** COMPUTATION OF NS
On 7 I=1,N
On 7 J=1,P
7 D< (I,J)=0.0, E0
On 8 I=1,N
On 8 J=1,P
On 8 K=1,N
8 D< (I,J)=DS(I,J)+A(I,K)*BJFF(K,J)
   IF (IECR*GT,5) WRITE(6,900) ((DS(I,J)+J=1,P)*I=1,N)
900 FORMAT(4(1X,'DS=',E10.4))

C** COMPUTATION OF CS
On 9 I=1,P
On 9 J=1,P
9 C< (I,J)=0.0, E0
On 10 I=1,P
On 10 J=1,P
On 10 K=1,N
10 C< (I,J)=CS(I,J)+C(I,K)*BUFF(K,J)
   IF (IECR*GT,5) WRITE(6,900) ((CS(I,J)+J=1,P)*I=1,P)
900 FORMAT(4(1X,'CS=',E10.4))
RETURN
END
SUBROUTINE MAIN4
COMMON/MAND4/A,B,C,RI,CBI,N,P,Q,R,IECR, EPS, RUFF
COMMON/MAN5/IRW,ICOL
COMMON/MAN6/OMEGC,DS,CS
COMMON/MAN7/INU,INJO
COMMON/MAN8/ADSC
COMMON/TRANS/XMD,UFF1,UFF2,UFF3
DIMENSION A(12,12),B(12,12),C(12,12),DI(12,12),CB(12,12)
DIMENSION RUFF1(12,12), XD(12,12),CS(12,12),DS(12,12)
DIMENSION RUFF(12*P1)
DIMENSION RUFF1(12,12), XD(12,12),CS(12,12),DS(12,12)
DIMENSION Ci(1,2)
DIMENSION .UFF3(12,12),IOW(12),ICOL(12),OEGC(12,12)
DIMENSION TNU(12),ADSC(12,12)
INTEGER P,Q,R
PR(IECR.GT.2)WPITE(6,100)
1100 FORMAT(1X,'SURROUINE MAIN4')
  !**COMPUTATION OF CS (CONTINUED)
  DO 1 I=1,P
     DO 2 J=1,P
       1 BijFF1(I,J)=-CS(I,J)
     2 BijFF1(I,J)=1.E08+UFF1(I,J)
     IF(IECR.GT.6)WRITE(6,100)((UFF1(I,J),J=1,N),I=1,P)
     100 FORMAT(4(1X,9E10.4))
  DO 3 I=1,P
     DO 4 J=1,N
       Cs(I,J)=0.F0
     3 Cs(I,J)=CS(I,J)*UFF1(I,J)
     DO 4 K=1,P
       Cs(I,J)=Cs(I,J)*UFF1(I,J)
     4 Cs(I,J)=Cs(I,J)*UFF1(I,J)
     IF(IECR.GT.6)WRITE(6,300)((UFF1(I,J),J=1,N),I=1,P)
     300 FORMAT(4(1X,9E10.4))
  DO 5 I=1,N
     DO 6 J=1,N
       A(I,J)=A(I,J)-UFF2(I,J)
     5 A(I,J)=A(I,J)-UFF2(I,J)
     6 A(I,J)=A(I,J)-UFF2(I,J)
     IF(IECR.GT.6)WRITE(6,400)((BUFF2(I,J),I=1,N),J=1,P)
     400 FORMAT(4(1X,9E10.4))
  C**COMPUTATION OF A-DS*C
  DO 4 I=1,N
     DO 5 J=1,N
       BUFF2(I,J)=0.E0
     4 Buff2(I,J)=0.E0
     5 Buff2(I,J)=0.E0
     DO 5 K=1,P
       Buff2(I,J)=Buff2(I,J)+DS(I,K)*C(K,J)
     5 Buff2(I,J)=Buff2(I,J)+DS(I,K)*C(K,J)
     IF(IECR.GT.6)WRITE(6,500)((BUFF2(I,J),I=1,N),J=1,P)
     500 FORMAT(4(1X,9E10.4))
  DO 6 I=1,N
     DO 7 J=1,N
       ASC(I,J)=A(I,J)-BUFF2(I,J)
     6 ASC(I,J)=A(I,J)-BUFF2(I,J)
     7 ASC(I,J)=A(I,J)-BUFF2(I,J)
     IF(IECR.GT.6)WRITE(6,600)((BUFF2(I,J),I=1,N),J=1,N)
     600 FORMAT(4(1X,9E10.4))
  C**COMPUTATION OF XMD=MD*
  DO 6 I=1,P
     DO 7 J=1,N
       XM(D(I,J)=CS(I,J)
     6 XM(D(I,J)=CS(I,J)
     7 XM(D(I,J)=CS(I,J)
     IF(NN.EQ.0)GO TO 11
**A-1**

```fortran
DO 11 L=1,N
N=1+P
DO 10 A I=1,P
DO 9 B J=1,N
BUFF2(I,J)=0.0
DO 8 K=1,N
10 IF(IECR.GT.1) STOP 11
BUFF2(I,J)=BUFF2(I,J)+BUFF3(I,K)*BUFF1(K,J)
WRITE(6,1500)((BUFF2(I,J),J=1,N),I=1,N)
11 CONTINUE
```

```fortran
DO 9 B I=1,P
DO 8 J=1,N
BUFF3(I,J)=BUFF2(I,J)
DO 7 I=1,N
10 I=I+L*P
YMD(I,J)=BUFF2(-I,J)
WRITE(6,2000)((YMD(I,J),J=1,N),I=1,N)
7 CONTINUE
```

**A-2**

```fortran
BUFF2(I,J)=0.0
RI=4
CALL ORTRED(XMD,OMEGC,NPNIXMDEPSI1,IECR)
WRITE(6,700)((OMEGC(I,J),J=1,N),I=1,N)
800 CONTINUE
```

```fortran
IF(IECR.GT.1) STOP 900
IF(SEG.EQ.11) G0 TO 15
900 CONTINUE
```

**A-3**

```fortran
DO 13 A I=1,P
DO 12 B J=1,N
BUFF2(I,J)=OMEGC(I,J)
CALL MFGR(RUFF2,OMEGC,N,P,N,I,OMEGC,12,IRANK,IRANK,EPS1,IER)
WRITE(6,9000)(IER)
900 CONTINUE
```

```fortran
WRITE(6,10000)(IRANK)
1000 CONTINUE
```

**A-4**

```fortran
IF(SEG.EQ.11) G0 TO 15
READ(15,1000) ISIGN
```

```fortran
1000 CONTINUE
```

```fortran
```
```
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(IECQ.GT.0) WRITE(6,1001)(IROW(I),ICOL(I),I=1,N)
1001 FORMAT(2(1X,IPOW=,I3,ICOL=,,I3))
If(IECQ.GT.0) WRITE(6,1002)((BUFF2(I,J),J=1,N),I=1,N)
1002 FORMAT(4(1X,'BUFF2=,F10.4))
RETURN
END
SUBROUTINE MAINS
COMMON/MAIN13/ROW,ICOL
COMMON/MAIN4/OMEGC,DS,CS
COMMON/MAIN5/INU,INUS,INJO
COMMON/MAIN5A/GG
COMMON/TRAS42/XMD2*,XMD2+,BUFF1,BUFF2+,RJFF3
DIMENSION A(12,12),B(12,12),C(12,12),B1(12,12),CBI(12,12)
DIMENSION A1F(12,12)
DIMENSION CS(12,12),B11(12,12),M100050
DIMENSION VECRUIJ(12),VECBU(I)=0.
DIMENSION INU(12),GG(12,12)
DIMENSION CS(12,12),nsl(12,12),OMEGC(12,12),ICOL(12)
DIMENSION STUDY(12)*IDX(12,12),BUFF2(12,12),BUFF3(12,12)
DIMENSION YWD(144,12),X432(144,12)
DIMENSION TAD(12)
DIMENSION INU(12),GG(12,12)
C**COMPUTATION OF M
CALL MAINS(XMD2)
INTEGER P,J,R
C**COMPUTATION FOR EACH BI,J=1,P
On 15 JJ=1,R
C**COMPUTATION OF DBI
C'=0.
On 1 I=1,P
C'=CC+CBI(I,JJ)**2
C'=1./CC
IF(IEC.R.GT.6) WRITE(6,100)CC,JJ
100 FORMAT(1X,*(CC,JJ) = F10.4, FOR I = 1,13)
On 2 I=1,N
VECBU(I)=0.
On 2 K=1,N
2 VECB1(I)=VECBU(I)+A(1I,K)*BI(K,JJ)
IF(IECR.GT.6) WRITE(6,200)(VECBU(I),I=1,N)
200 FORMAT(4(1X,VECBU=*,F10.4))
On 3 I=1,P
On 3 J=1,P
3 Buff(J,J) = VECBUI(I,K)*BB(K,JJ) +CC
IF(IEC.R.GT.6) WRITE(6,300)((BUFF(J,J),J=1,P),I=1,N)
300 FORMAT(4(IXTBUFFJ =*,E10.4))
C**COMPUTATION OF C
On 4 I=1,P
On 4 J=1,P
4 Buff(I,J) = - CRI(I,JJ)*CRI(J,JJ)*CC
On 5 I=1,P
5 Buff(I,I) =1.,BUFF1(I,1)
IF(IECR.GT.6) WRITE(6,400)((BUFF1(I,J),I=1,P),I=1,P)
400 FORMAT(4(1X,BUFF1 = *,E10.4))
On 6 I=1,P
On 6 J=1,N
6 Buff2(I,J) =0.
On 6 K=1,P
6 Buff2(I,J) =BUFF2(I,J)*BUFF1(I,K)*C(K,J)
IF(IEC.R.GT.6) 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
On 7 I=1,N
On 7 J=1,N
7 Buff3(I,J) =0.
On 7 K=1,P
7 Buff3(I,J) =BUFF3(I,J)*BUFF1(I,K)*C(K,J)
IF(IEC.R.GT.6) WRITE(6,600)((BUFF3(I,J),J=1,N),I=1,P)
600 FORMAT(4(1X,*,CPRIME =*,E10.4))
A-14

8 \text{BUFF}(I, J) = A(I, J) - BLF(I, J)
\text{WRFAT}(4(1X,*A=DSUP+C=*,E10.4))
\text{CONTUATION OF XMD=MD'}
\text{On 9 I=1,P}
\text{On 9 J=1,N}
XwD(I, J) = BUFF2(I, J)
\text{BUFF}(I, J) = BUFF2(I, J)
N' = N + 1
\text{On 10 K=1,N}
10 \text{BUFF}(I, J) = BUFF2(I, J) + BUFF3(I, K) * BUFF1(K, J)
\text{WRFAT}(4(1X,*A=DSUP+C=*,E10.4))
A-15

```fortran
908 FORMAT(1X,†IF YOU WANT TO OVERCOME THE RANKS OF OMEGI, =1, R†)
       1†IVEN BY AUTOMATIC COMPUTATION, TYPE 00, OTHERWISE, TYPE 11†)
909 FORMAT(1X,†TYPE THE RANKS OF OMEGI, =1, R†)
6 = R
8 = AD(5,*) (INU(I), J=1, IR)  
9010 FORMAT(1X,†IF SATISFIED, TYPE 11, IF NOT TYPE 00†)
       R = AD(5,100) ISIGN
       IF(ISIGN.EQ.0) GO TO 17
      CONTINUE
?ALL MAINSA
RETURN
ED
SUBROUTINE MAINSA
** BUILDS THE MATRIX GG OF THE GENERATORS
COMMON/MANSA/GG
COMMON/MANSA/A,B,C,RI,CBI*NP*Q,R,IECR,EPS,RAUFF
DIMENSION A(12,12), R(12,12), C(12,12), B1(12,12), CBI(12,12)
DIMENSION RAUFF(12,12), GG(12,12)
1 = INTEGER P, Q, R
IF(IECR.GT.2) WRITE(6,100)
      CONTINUE
ON 3 II=1,N
2 WRITE(6,101) II
901 FORMAT(1X,†WRITE GG(I,J), J=1, IR, I=1, I3†FREE FORMAT†)
      IN = R
8 = AD(5,*) (GG(II,J), J=1, IR)
902 FORMAT(1X,†IF SATISFIED, TYPE 11, IF NOT, TYPE 00†)
       R = AD(5,103) ISIGN
5 FORMAT(1X,†SUBROUTINE MAINSA†)
1 CONTINUE
3 CONTINUE
6 WRITE(6,104) (GG(I,J), J=1, IR, I=1, N)
904 FORMAT(4(1X,*GG=*,E10.4))
6 WRITE(6,102)
5 READ(5,103) ISIGN
6 IF(ISIGN.EQ.0) GO TO 1
7 RETURN
ED
```

A01230
A01240
A01240
A01250
A01260
A01270
A01280
A01290
A01300
A01310
A01320
A01330
A01340
A01350
A01360
A01370
A01380
A01390
A01400
A01410
A01420
A01430
A01440
A01450
A01460
A01470
A01480
A01490
A01500
A01510
A01520
A01530
A01540
A01550
A01560
A01570
A01580
A01590
A01600
A01610
A01620
A01630
A01640
A01650
A01660
A01670
A01680
A01690
A01700
SUBROUTINE AIN5P(XMD?)

COMMON/MONZ/A,B,C,BI,CBI,N,P,D,R,IEXC,EPS,BUFF
COMMON/TRASH2/RUFFI MAL(0030
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,D,R

IE=IECR,G,20 WRITE(6,99)

99 FORMAT(1X,'SUBROUTINE AIN5P')

**COMPUTATION OF M**

1 = 1
DO 1 J=1,N
XMD2(I,J)=C(I,J)
1 N=N-1
D 5 L=1,N
N'=N-1
D 2 I=1,N
DO 6 J=1,N
BUFF1(I,J)=0.E0
6 D 2 K=1,N
BUFF1(I,J)=BUFF1(I,J)+BUFF(I,K)*A(K,J)
D 5 I=1,N
DO 10 J=1,N
BUFF1(I,J)=BUFF1(I,J)
10 D 3 P=I+N
D 3 J=1,N
I=I+L*P
3 BUFF1(I,J)=BUFF1(I,J)
D 4 J=1,N
I=I+L*P
4 XMD2(I,J)=BUFF(I,J)
5 CONTINUE
N=N*P
IE=IECR,G,30 WRITE(6,200)((XMD2(I,J),J=1,N),I=1,NP)
200 FORMAT(4(1X,'XMD2=',E10.4))
RETURN
END
SUBROUTINE MAIN6
COMMON/MANT?/A,B,C,B1,CI,N,P,Q,R,IECR,ESP,RUFF
COMMON/MANUS/INUS,INJO
DIMENSION INU(12)
DIMENSION A(12,12),R(12,12),C(12,12),CI(12,12),BA(12,12)
DIMENSION I(12,12)
INTEGER PR
C** CHECK WHETHER SUM NUI=NU
C** COMPUTATION OF DETECTION SPACE DIMENSION
DO 1 I=1,R
   1 I=INU(I)+1
   IF (IECR.GT.1) WRITE(6,100)(INU(I),I=1,R)
100 FORMAT(1X,"DETECTION SPACE OF B1,I3, has a dimension",I3)
   I=INUS+R
   IF (IECR.GT.1) WRITE(6,200)INUS
200 FORMAT(1X,"DETECTION SPACE ASSOCIATED WITH THE SET OF RI",I3)
   I=(SELECTED HAS A DIMENSION",I3)
   SUM=0
   DO 2 I=1,R
      2 SUM=SUM+INU(I)
   WRITE(6,201)SUM
201 FORMAT(1X,"SUM OF NUI IS",I3)
   INUS=INUS-SUM
   WRITE(6,202)
202 FORMAT(1X,"IF THE SUM OF THE NUI IS EQUAL TO NUS-DIMENSION",I3)
   IF THE DETECTION SPACE ASSOCIATED WITH THE SET OF RI-THES",I3)
   IF THEM IS MUTUALLY DETECTABLE,A DETECTION FILTER CAN BE "",I3)
   IF NEEDED FOR THIS SET WITH ASIGNABLE EIGENVALUES",I3)
   IF IT IS NOT A TOTAL OF NUS-SUM OF NUI) EIGENVALUES ARE",I3)
   IF NASSIGNABLE,IF YOU WANT TO CHECK THEIR VALUE,TYPE 1",I3)
   OTHERWISE TYPE 00")
READ(5,10)SIGN
WRITE(5,10)SIGN
10 IF (SIGN.NE.11) GO TO 4
WRITE(6,203)
203 FORMAT(1X,"THE PROGRAM WILL PROCEED ON COMPUTING THE UNASSIGNED",I3)
   IF YOU MADE A MISTAKE IN THE OPTION,TYPE 1")
READ(5,10)SIGN
WRITE(6,10)SIGN
10 IF (SIGN1.EQ.0) GO TO 3
CALL MAIN6
RETURN
4 WRITE(6,204)
204 FORMAT(1X,"THE PROGRAM WILL NOT COMPUTE THE UNASSIGNED",I3)
   IF YOU MADE A MISTAKE IN SELECTING THE OPTION TYPE 00")
READ(5,10)SIGN
WRITE(6,10)SIGN
10 IF (SIGN1.EQ.0) GO TO 3
RETURN
END
DIMENSION a(12,12), B(12,12), C(12,12), I(12,12), CB[12,12)

DIMENSION MEGC(12,12), DS(12,12), CS(12,12), RUFF(12,12)

DIMENSION SUFF(12,12), ADC(12,12), XM3(12,12), BUFF3(12,12), INU(12)

INTEGER PR

IF (IECR.GT.2) WRITE(6,100)

FORMAT(1X, 'SUBROUTINE MAIN6A')

C** SUBROUTINE MAIN6A


C** COMMON/MAN4/MEGC:DS:CS

C** COMMON/MAN5/INU:INUS:INUO

C** COMMON/MAN6/RUFF1

C** COMMON/MAN6A/RUFF1

C** COMMON/MAN6A/ADSC

C** COMMON/MAN6B/XMO:IND

C** COMMON/TRASH2/PUFF3

10 FORMAT(1X, 'RORMAT(INU)')

I=1:R

J=1:N

BUFF(I,J)=0.0

K=1:P

BUFF(I,J)=BUFF(I,J)+BUFF(I,K)*C(K,J)

BUFF(I,J)=BUFF(I,J)

BUFF3(I,J)=BUFF3(I,J)+BUFF3(I,K)*ADSC(K,J)

CONTINUE
CONTINUE
IF (IECR .GT. 5) WRITE (6, 300) ((XMO(I,J), J=1,N), I=1,IND)
300 FORMAT (4(1X, M0*, E10.4))
DO 10 I=1,N
   DO 10 J=1,N
10   BUFFER(I,J) = BUFFER(I,J)
C** BUFFER = CI IN COMMON MANI6
CALL MAIN6B
RETURN
SUBROUTINE MAIN6A
COMMON/MAN12/A,B,C,B1,CRI+N+P+R+ICR,EPS,BUFF
COMMON/MAN14/OMEGC,DS,CS
COMMON/MAN15/INU,INUS,INJO
COMMON/MAN16/BUFF1

C** BUF1=(CBI*CBI)-1*C8IT

C** COMMONS FROM MAIN3
COMMON/MAN16A/ADSC
COMMON/MAN16B/XMO,IND
COMMON/TRASH2/BUFF3

DIMENSION A(91,2),R(12,12),C(12,12),8I(12,12),CBI(1212)
DIMENSION OMEGC(1212),DS(12,12),CS(12,12),BUFF1(1212)
DIMENSION UFF(12912),ADSC(12,12),XM3(1212),BUFF3(12912),INUJ(12)

INTEGER P2,R

IF(IECR.GT.2)WRITE(6,100)
100 FORMAT(I4,'SURROUTINE MAIN6A')

C** COMPUTATION FROM MAIN3

DO 9 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)+BUFF(I,K)*C(K,J)
IF(IECR.GT.2)WRITE(6,101)((BUFF(I,J),J=1,N),I=1,R)
101 FORMAT(4(I4,'=',1X,10.4))

DO 9 I=1,R
DO 9 J=1,N
BUFF(I,J)=0.E0
DO 9 K=1,N
3 BUFF(I,J)=BUFF(I,J)+BUFF(I,K)*BUFF3(K,J)
9 CONTINUE

C** COMPUTATION OF MO=XMO

IF(IECR.GT.2)WRITE(6,701)((BUFF(I,J),J=1,N),I=1,N)
701 FORMAT(4(I4,'=',1X,10.4))

DO 10 I=1,N
10 BUFFER(I,J)=ADSC(I,J)

DO 11 J=1,N
XUO(I,J)=BUFF(I,J)
11 CONTINUE

DO 12 J=1,N
BUFF(I,J)=BUFF(I,J)+BUFF(I,J)*ADSC(I,J)
12 CONTINUE

IF(IECR.GT.2)WRITE(6,300)((BUFF(I,J),J=1,N),I=1,N)
300 FORMAT(4(I4,'=',1X,10.4))

DO 13 J=1,N
BUFF(I,J)=BUFF(I,J)+BUFF(I,J)*BUFF3(I,J)
13 CONTINUE

DO 14 J=1,N
BUFF(I,J)=BUFF(I,J)+BUFF(I,J)*BUFF3(I,J)
14 CONTINUE

DO 15 J=1,N
BUFF(I,J)=BUFF(I,J)+BUFF(I,J)*BUFF3(I,J)
15 CONTINUE

DO 16 J=1,N
BUFF(I,J)=BUFF(I,J)+BUFF(I,J)*BUFF3(I,J)
16 CONTINUE

DO 17 J=1,N
BUFF(I,J)=BUFF(I,J)+BUFF(I,J)*BUFF3(I,J)
17 CONTINUE

DO 18 J=1,N
BUFF(I,J)=BUFF(I,J)+BUFF(I,J)*BUFF3(I,J)
18 CONTINUE

DO 19 J=1,N
BUFF(I,J)=BUFF(I,J)+BUFF(I,J)*BUFF3(I,J)
19 CONTINUE

DO 20 J=1,N
BUFF(I,J)=BUFF(I,J)+BUFF(I,J)*BUFF3(I,J)
20 CONTINUE

DO 1 I=1,R
1 INU1=INU(I)-2
PRINT 100
100 FORMAT(4(I4,'=',1X,10.4))

DO 2 J=1,N
2 XUO(I,J)=BUFF(I,J)

DO 3 J=1,N
3 XUO(I,J)=BUFF(I,J)

DO 4 J=1,N
4 XUO(I,J)=BUFF(I,J)

DO 5 J=1,N
5 XUO(I,J)=BUFF(I,J)

DO 6 J=1,N
6 XUO(I,J)=BUFF(I,J)

DO 7 J=1,N
7 XUO(I,J)=BUFF(I,J)

DO 8 J=1,N
8 XUO(I,J)=BUFF(I,J)

DO 9 J=1,N
9 XUO(I,J)=BUFF(I,J)

DO 10 J=1,N
10 XUO(I,J)=BUFF(I,J)

DO 11 J=1,N
11 XUO(I,J)=BUFF(I,J)

DO 12 J=1,N
12 XUO(I,J)=BUFF(I,J)

DO 13 J=1,N
13 XUO(I,J)=BUFF(I,J)

DO 14 J=1,N
14 XUO(I,J)=BUFF(I,J)

DO 15 J=1,N
15 XUO(I,J)=BUFF(I,J)

DO 16 J=1,N
16 XUO(I,J)=BUFF(I,J)

DO 17 J=1,N
17 XUO(I,J)=BUFF(I,J)

DO 18 J=1,N
18 XUO(I,J)=BUFF(I,J)

DO 19 J=1,N
19 XUO(I,J)=BUFF(I,J)

DO 20 J=1,N
20 XUO(I,J)=BUFF(I,J)
BUFF1(I,J) = BUFF3(I,J)
CONTINUE
CONTINUE
I = IECR.GT.5) WRITE (6,300) ((XMO(I,J), J=1,N), I=1,IND)
FORMAT (4(1X,'MO=',E10.4))
DO 10 I=1,N
DO 10 J=1,N
BUFF1(I,J) = BUFF(I,J)
C** Buff1 = CI in common MANI6
CALL MAIN63
RETURN
END
SUBROUTINE MAIN68
COMMON/MAN12/A,B,C,R,E,CI,N,P,Q,R,IECR,EPS,BUFF
COMMON/MAN13/IROW,ICOL
COMMON/MAN14/OMEGC,DS,CS
COMMON/MAN15B/XMO,IND
COMMON/MAN16C/BUFF1
DIMENSION XMO(12,12),OMEGC(12,12),DS(12,12),CS(12,12)
DIMENSION RUFF(12,12),BUFF1(12,12),DS(12,12),CS(12,12),CBI(12,12)
DIMENSION CI(12,12),BUFF1(12,12),ICOL(12),IROW(12)
INTEGER P,Q,R
IF(IECR.GT.2)WRITE(6,100)
100 FORMAT(1X,SUBROUTINE MAIN68*)
C** ORTHOGONAL REDUCTION OF XMO=W3
DN 1 I=1,N
DN 1 J=1,N
B*FF(I,J)=OMEGC(I,J)
BUFF=12
CALL ORTRED(XMORUFFIND.NIBUFFEPSI1,IECR)
IF(IECR.GT.2)WRITE(6,101)((BUFF(I,J),J=1,N),I=1,N)
101 FORMAT(4(I1,1X,OMEG3G=('E11.4)))
CALL MAIN6C
RETURN
END
SUBROUTINE MAIN6C
COMMON/MAN12/A,B,C,R,E,CI,N,P,Q,R,IECR,EPS,BUFF
COMMON/MAN15/INU,INUS,JNO
COMMON/MAN16C/RUFF1
COMMON/MAN16D/ROG
COMMON/TRASH1/IZ
C** IN COMMON/MAN16C/RUFF1=OMEGG+COMES FROM MAIN68
DIMENSION (12,12),BUFF1(12,12),ICOL(12),IRANK(12,12)
DIMENSION CI(12,12),BUFF1(12,12),ICOL(12),IRANK(12,12)
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*',/X*
1*DEEP BY TYPING THEIR NUMBERS IN FORMAT I3*,IF NNU=NU=(SUM NI)*',/X*
1*INDICATE WHAT ARE THE NNU FIRST LINEARLY INDEPENDENT*',/X*
1*COLUMNS OF OMEGOG*')
READ(5,1000)((7(I),I=1,N),I=1,N)
1000 FORMAT(12I3)
WRITE(6,1010)((IZ(I),I=1,N),I=1,N)
WRITE(6,1010)(IZ(I),I=1,N)
WRITE(6,1010)
101 FORMAT(1X,*IF DATA ARE INCORRECTLY ENTERED,TYPE 00*,OTHERWISE*',/X
A-22
1. *TYPE 11*

READ(5,100) ISIGN

100 FORMAT(12)

IF(ISIGN.EQ.0) GO TO 1

1 J=1,INUO

J=IZ(I)

Dn 2 I=1,N

2 ROG(I,J)=BUFF1(I,J)

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/MAN12/A,B,C,B1,CBI,N,P,Q,R,IECR,EPS,BUFF

COMMON/MAN16/CI

COMMON/MAN15/INU,INUS,INUO

COMMON/MAN6D/ROG

C** CI HAS BEEN COMPUTED IN MAIN6A(WAS BUFF1)

COMMON/MAN16A/ADSC

COMMON/MAN6E/TETA

DIMENSION A(12,12),B(12,12),C(12,12),D(12,12),E(12,12),F(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,FORMAT(*,SUBROUTINE MAIN6D*))

IF(IECR.GT.8) WRITE(6,98)((TETA(I,J),J=1,N),I=1,R)

98 FORMAT(4(1X,TETA='*,E10.4))

IF(IECR.GT.97) WRITE(6,97)((ADSC(I,J),J=1,N),I=1,N)

97 FORMAT(4(1X,ADSC='*,E10.4))

Dn 6 I=1,N

TETA(I,J)=0.0

Dn 1 J=1,N

TeTA(I,J)=TETA(I,J)+CI(I,K)*ADSC(K,J)

Dn 2 J=1,N

2 BUFF(I,J)=TETA(I,J)

IF(INUI.EQ.0) GO TO 6

Dn 5 JJ=1,INU

Dn 3 J=1,N

TETA(I,J)=0.0

Dn 3 J=1,N

3 TETA(I,J)=TETA(I,J)+BUFF(I,K)*ADSC(K,J)

Dn 4 K=1,N

4 BUFF(I,K)=TETA(I,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,TETA='*,E10.4))

Dn 7 I=1,R

Dn 7 J=1,INUO

BUFF(I,J)=0.0

Dn 7 K=1,N

7 BUFF(I,J)=BUFF(I,J)+TETA(I,K)*ROG(K,J)

Dn 8 I=1,R

Dn 8 J=1,INUO

8 TETA(I,J)=BUFF(I,J)
C***TETA IS A (R*INUO) MATRIX
10 (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/MANT2/A,B,C,BI,CR1,INP,R,IECH,EP2,UFF
COMMON/MAN5/INU,INUS,INJO
COMMON/MAN5A/CG
COMMON/MAN5D/ROG
COMMON/MAN6E/XPI
COMMON/TRASH2/CBVEC,CBVIV,UFF1,UFF2,UFF3
DIMENSION A(12,12),B(12,12),C(12,12),DI(12,12),CB(12,12)
DIMENSION ROG(12,12),RO3(12,12),GG(12,12),TETA(12,12)
DIMENSION XPI(12,12),INU(12,12),CBVEC(75),CBVIV(75),BUFF1(12,12)
DIMENSION BUFF2(12,12),BUFF3(12,12)
 INTEGER P,R
IF (IECR GT 2) WRITE (6,100)
100 FORMAT (1X,'SUBROUTINE MAIN6E*)
C** COMPUTATION OF MATRIX XPI=-1
C** COMPUTATION OF (ROG*ROG)
   DO 1 I=1,INUO
      DO 1 J=1,INUO
         B,UFF(I,J)=0.E0
      DO 1 K=1,INUO
         B,UFF(I,J)=BUFF(I,J)+ROG(K,I)*ROG(K,J)
   1 CONTINUE
C** TRANSITION TO SYMMETRIC STORAGE
   IN=12
   IF (IECR GT 5) WRITE (6,99) INUO
   99 FORMAT (1X,'INUO=',I3) GO TO 11
   CALL VCVTFS(BUFF,INUO,IR,CBVEC)
   101 FORMAT (4(1X,'BUFF=',E10.4))
   CALL (4(1X,'BUFF=',E10.4))
102 FORMAT (4(1X,'BUFF=',E10.4))
C** COMPUTATION OF (ROG*ROG)-1
C** TRANSITION TO INVERSE OF CBVEC
   CALL LINV1 (CBVEC,INUO,CBVIV,IDG,T1,D2,IER)
   IF (IECR GT 8) WRITE (6,103) (CBVEC(I),I=1,INUO)
   103 FORMAT (4(1X,'CBVEC=',F10.4))
C** COMPUTATION OF (ROG*ROG)-1*ROG
   DD 2 J=1,INUO
      B,UFF2(1,1)=0.E0
   DD 2 K=1,INUO
      B,UFF2(I,J)=BUFF1(I,J)+BUFF2(I,K)*ROG(J,K)
   105 FORMAT (4(1X,'BUFF1=',E10.4))
C** COMPUTATION OF A*ROG
   DD 3 J=1,INUO
      B,UFF2(I,J)=0.E0
   DD 3 K=1,INUO
      B,UFF2(I,J)=BUFF2(I,J)*A(I,K)*ROG(K,J)
   106 FORMAT (4(1X,'BUFF2=',E10.4))
**COMPUTATION OF G*GAMMA**

![](image)

**COMPUTATION OF A*GAMMA**

![](image)

**COMPUTATION OF XPI**

**XPI IS A INUO*INUO MATRIX**

![](image)

**COMPUTATION OF XPI EIGENVALUES**

![](image)
SUBROUTINE MAIN7
INTEGER P,N
COMMON/MAN1/A,B,C,RI,CBI,N,P,Q,R,IECR,EPS,UFF
COMMON/MAN2/IPOW4,ICOL
COMMON/MAN3/IPOW3,INUUS,INJO
COMMON/MAN4/TETA
COMMON/MAN5/XPI
COMMON/TRASH/RUFF1,9UFF2,9MOI
DIMENSION A(12,12),B(12,12),C(12,12),RI(12,12),CBI(12,12)
DIMENSION INU(12),BUFF(12,12),TETA(12,12),XPI(12,12),BUFF1(12,12)
DIMENSION BUFD(12,12),XMOI(12,12),ICOL(12)
INTEGER R,J
IF(IECR.GT.2)WRITE(6,99)
99 FORMAT(2X,*SUBROUTINE MAIN7*)
C* STEG SG
C* COMPUTATION FOR ALL BIS
D= 12 II=1,R
WRITE(6,98)II
98 FORMAT(1X,*COMPUTATION OF THE EIGENVALUES ASSOCIATED WITH B*12)
I=0=1
C* COMPUTATION OF MOI
Ni=INUUS-1
D= 1 J=1,INUUS
XMOI(IND,J)=TETA(II,J)
1 BiFF(1,J)=TETA(II,J)
IF(INU_EQ.0)GO TO 4
```fortran
C** ORTHOGONAL REDUCTION OF MOI
   DO 6 I=1,INUO
      6 BFF(I)=1.E0
   DO 5 I=1,INUO
      5 BFF(I,J)=BUFF(I,J)
   IF(IECR.GT.12)WRITE(6,19)IER
      19 FORMAT(1X,ERROR CODE IN COMPUTATION OF RANK OF BETA IS ,I3)
   IF(IECR.GT.2)WRITE(6,104)(IROW(I),ICOL(I),I=1,INUO)
      104 FORMAT(4(1X,#DEP OF BFTA'=E0.4))
   CONTINUE
   IANK=I
   IF(ABS(BUFF(I,I)).LT.EPSI) IANK=0
   C** COMPUTATION OF RANK
      103 IF(IECR.GT.2)WRITE(6,103)IRANK
         103 FORMAT(1X,*RANK OF BETA IS ,I3)
   C** COMPUTATION OF BETA
      105 FORMAT(4(1X,*DEP OF BFTA'=E0.4))
   CONTINUE
      66 IF (IECR.GT.5)WRITE(6,103)IRANK
         103 FORMAT(1X,RANK OF BETA IS ,I3)
   CONTINUE
   C** CONSTRUCTION OF DELTA
      106 FORMAT(4(1X,*BETA='*,E10.4))
   CONTINUE
      9 BFF1(I,J)=XMOI(I,J)
```

97 FORMAT(1X*, INUO = 1, 13)
IF (INUO.EQ.1) GO TO 8A
CALL MFORBUFF(1, INMO, INUO, INUO, IRANK1, IROW, ICOL, EPS11, IER)
IF (IECR.GT.2) WRITE(6,107) IER
107 FORMAT(1X*, ERROR CODE IN COMPUTATION OF RANK OF XMOI IS 1, 13)
IF (IECR.GT.2) WRITE(6,109) (IROW(I), ICOL(I), I=1, INUO)
109 FORMAT(1X*, IROW = 1, 13, ICOL =1, 13)
IF (IECR.GT.2) WRITE(6,110) (BUFF(I,J), J=1, INUO), I=1, INUO)
110 FORMAT(4(1X, DEP OF DELTA IS *E10, 4))
GO TO 88
88 CONTINUE.
IRANK1 = 1
IF (ABS(BUFF(I+1,J)).LT.EPS11) IRANK1 = 0
888 CONTINUE.
IF (IECR.GT.2) WRITE(6,108) IRANK1
108 FORMAT(1X*, RANK OF XMOI IS 1, 13)
IF (IRANK1.EQ.0) GO TO 999
CALL MAIN78B(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, DEPA = *E10, 4))
ISUM = IRANK + IRANK1
IF (ISUM.EQ.INUO) GO TO 10
WRITE(6,1112)
112 FORMAT(1X*, IRANK = IRANK1, IRANK1, NE INUO THERE IS A MISTAKE*)
STOP
10 CONTINUE.
C** FORMATION OF DELTA: BETA
C** DELTA: BETA IS A INTO*INUO MATRIX
ON 11 I=1, INUO
IF (IRANK.EQ.0) GO TO 1110
ON 11 J=1, IRANK
ON J: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 LAMBDA1I
CALL MAIN78C(IRANK)
12 CONTINUE.
CALL MAIN78
RETURN
END
SUBROUTINE MAIN7AA(IRANK)
C** SELECTION OF BETA FROM THE COLUMNS OF BUFF=BETAP
C** BUFF - COMES FROM MAIN7A
COMMON/MAN15/INU,INUS,INJO
COMMON/TRASH1/IZ
COMMON/TRASH2/BUFF1
DIMENSION A(12,12),R(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 J=1,INU0
1 B,FF(I,J)=BUFF(I,J)
IF(IECR.GT.12)WRITE(6,100)((BUFF(I,J),J=1,INU0),I=1,INU0)
100 FORMAT(4(1X,'BUFF=',E10.4))
IF(IECR.GT.2)WRITE(6,101)
101 FORMAT(1X,5SUBROUTINE MAIN7AA1)
WRITE(6,102)
102 FORMAT(1X,IDENTIFY WHICH COLUMNS OF ETAP YOU WISH TO KEEP,/,1X)
1+TYPE THEIR NUMBERS IN FORMAT 9.YOU MUST KEEP RANK OF BETAP COL)
READ(5,1000)(IZ(I),I=1,IRANK)
1000 FORMAT(12I3)
WRITE(6,103)(IZ(I),I=1,IRANK)
103 FORMAT(1X,IF DATA ARE INCORRECTLY ENTERED,TYPE 00*,/IX*)
READ(5,1001) ISIGN
1001 FORMAT(12I3)
IF(ISIGN.EQ.0) GO TO 2
D= 3 J=1,IRANK
J''=IZ(J)
D= 3 I=1,INU0
DO 3 J=1,INU0
3 B',FF(I,J)=BUFF(I,J)
IF(IECR.GT.5)WRITE(6,104)((BUFF(I,J),J=1,INU0),I=1,INU0)
104 FORMAT(4(1X,'BETAP=',E10.4))
RETURN
END

SUBROUTINE MAIN7AB(IRANK1)
C** SELECTION OF DELTA FROM THE COLUMNS OF BUFF
C** BUFF - COMES FROM MAIN7A
COMMON/MAN15/INU,INUS,INJO
COMMON/TRASH1/IZ
COMMON/TRASH2/BUFF1
DIMENSION A(12,12),R(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 J=1,INU0
1 B,FF(I,J)=BUFF(I,J)
IF(IECR.GT.12)WRITE(6,100)((BUFF(I,J),J=1,INU0),I=1,INU0)
100 FORMAT(4(1X,'BUFF=',E10.4))
IF(IECR.GT.2)WRITE(6,101)
101 FORMAT(1X,5SUBROUTINE MAIN7AA1)
WRITE(6,102)
102 FORMAT(1X,IDENTIFY WHICH COLUMNS OF XMOI YOU WISH TO KEEP,/,1X)
1+TYPE THEIR NUMBERS IN FORMAT 9.YOU MUST KEEP PK OF XMOI COL)
READ(5,1000)(IZ(I),I=1,IRANK)
1000 FORMAT(12I3)
WRITE(6,103)(IZ(I),I=1,IRANK)
103 FORMAT(1X,IF DATA ARE INCORRECTLY ENTERED,TYPE 00*,/IX*)
READ(5,1001) ISIGN
1001 FORMAT(12I3)
IF(ISIGN.EQ.0) GO TO 2
D= 3 J=1,IRANK
J''=IZ(J)
D= 3 I=1,INU0
DO 3 J=1,INU0
3 B',FF(I,J)=BUFF(I,J)
IF(IECR.GT.5)WRITE(6,104)((BUFF(I,J),J=1,INU0),I=1,INU0)
104 FORMAT(4(1X,'BETAP=',E10.4))
RETURN
END
A-31

103 FORMAT(1X, 'IF DATA ARE INCORRECTLY ENTERED, TYPE 00***/1x*
1* OTHERWISE TYPE 11**', A700560)
1001 FORMAT(1J2)
   IF(ISIGN.EQ.0) GO TO 2
   DO 3 J=1,IRANK 
   J=IZ(J)
   DO 3 I=1,INUO 
   3 BIFF(I,J)=BUFF(I,JJ)
104 FORMAT(4(1X,'BETA=*,E10.4))
RETURN
END

SUBROUTINE MAN7AC(IRANK)
C** COMPUTES THE SETS OF LAMBDAI
C** COMMON/MAN72/A,B,C,BI,CR1,N,P,Q,R,IECR,EPS,BUFF
C** COMMON/MAN71/INU,INUS,INU,INU,INJ
C** COMMON/TRANSH/BUFF1,BUFF2
C** COMPLEX WI(12)
D+ENSION A12,12),A(12,12),C(12,12),BI(12,12)
D+ENSION (11,12),BUFF(12,12),XPI(12,12),BUFF1(12,12),BUFF2(12,12)
D+ENSION INUS(12)
INTEGER PQR
C** COMPUTES THE INVERSE OF DELTA:BETA
   IF(IECR.GT.2) WRITE(6,100) A700560
   IF(IECR.GT.9) WRITE(6,101)((BUFF(I,J),J=1,INUO),I=1,INUO) A700560
101 FORMAT(4(1X,'BUFF(1,10.4))')
   DO 1 J=1,INUO 
   1 BIFF(I,J)=BUFF(I,J)
   UFF1=12
   UFF=1
   MAREA=12
   CALL LINV1(BUFF1,INUO,UFF,BUFF2,10GT,WKAREA,IER)
   IF(IECR.GT.5) WRITE(6,102) ((BUFF(I,J),J=1,INUO),I=1,INUO) A700560
102 FORMAT(4(1X,'BUFF1(*)=*,E10.4))
   DO 3 J=1,INUO 
   3 BIFF2(I,J)=BUFF2(I,J)
   DO 3 K=1,INUO 
   3 BIFF2(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) A700560
104 FORMAT(4(1X,'XPI=',E10.4))
RETURN
END

A-32
C** COMPUTATION OF XPI EIGENVALUES
I"OB=0
1BUFF=12
1=1
KIX=13
CALL EIGRF(BUFF, IRANK, IJFF, IJOB, WI, Z, IZ, KWK, IER)
106 FORMAT(1X, *ERROR CODE IN COMPUTATION OF XPII EIGENVALUES IS 'I13)
WRITE(6,107)(WI(I), I=1, IRANK)
107 FORMAT(1X, *LAMBDAI ARE', 2(E10.4,1X, E10.4))
RETURN
5 CONTINUE
WRITE(6,108)
108 FORMAT(1X, 'THE SET OF LAMBDAI ASSOCIATED WITH THIS BI IS', 1X,
'THE NULL SET')
RETURN
END
SUBROUTINE MAIN78
C* # SELECTS THE SET TO BE RETAINED
COMMON/MAN2/A, B, C, BI, CBI, N, P, R, IECR, EPS, BUFF
COMMON/TRASH1/IZ
COMMON/MAN10/IFLAG
DIMENSION A(12, 12), B(12, 12), C(12, 12), BI(12, 12), CBI(12, 12)
INTEGER P, Q, R
IF (IECR.GT.2) WRITE(6, 100)
1 FORMAT(1X, 'SUBROUTINE MAIN78')
WRITE(6, 101)
101 FORMAT(1X, 'IF YOU WANT TO CHANGE THE SET OF BI, TYPE 00*/1X,')
102 FORMAT(1X, 'OTHERWISE, TYPE 11')
READ(5, 1000) ISIGN
100 FORMAT(1X, 'IF(ISIGN.NE.11) GO TO 2')
WRITE(6, 102)
102 FORMAT(1X, 'THE PROGRAM WILL NOT ALLOW YOU TO CHANG THE SET*/1X,')
103 FORMAT(1X, 'IF YOU MADE A MISTAKE IN SELECTING THIS OPTION*/1X,')
104 FORMAT(1X, 'OTHERWISE, TYPE 11')
READ(5, 1000) ISIGN
105 FORMAT(1X, 'IF(ISIGN.EQ.0) GO TO 1')
RETURN
2 WRITE(6, 103)
103 FORMAT(1X, 'THE PROGRAM WILL ASK IF YOU WANT TO SUBTRACT*/1X,')
104 FORMAT(1X, 'THE NUMBER OF EVENTS YOU WANT TO RETAIN')
READ(5, *) IRI
WRITE(6, 105)
105 FORMAT(1X, 'IF YOU MADE A MISTAKE IN TYPING THE DATA TYPE 00*/1X,')
106 FORMAT(1X, 'OTHERWISE, TYPE 11')
READ(5, 1000) ISIGN
C** FOR.ATION OF THE NEW MATRIX BI

5 R=IZ(IJ)
5 B:FF(I,J)=B1(I,J)
5 Dn 6 I=1,N
5 Dn 6 J=1,IRI
6 Bi(I,J)=BUFF(I,J)
7 F0RMA T(4,107)WRITE(6,107)((B1(I,J),J=1,IRI),I=1,N)

C** FOR.ATION OF THE NEW MATRIX BI

7 R=IZ(IJ)
7 B:FF(I,J)=B1(I,J)
7 Dn 6 I=1,N
7 Dn 6 J=1,IRI
8 Bi(I,J)=BUFF(I,J)
9 F0RMA T(4,107)WRITE(6,107)((B1(I,J),J=1,IRI),I=1,N)

C** FOR.ATION OF THE NEW MATRIX BI

10 R=IZ(IJ)
10 B:FF(I,J)=B1(I,J)
10 Dn 6 I=1,N
10 Dn 6 J=1,IRI
11 Bi(I,J)=BUFF(I,J)
12 F0RMA T(4,107)WRITE(6,107)((B1(I,J),J=1,IRI),I=1,N)

C** FOR.ATION OF THE NEW MATRIX BI

15 R=IZ(IJ)
15 B:FF(I,J)=B1(I,J)
15 Dn 6 I=1,N
15 Dn 6 J=1,IRI
16 Bi(I,J)=BUFF(I,J)
17 F0RMA T(4,107)WRITE(6,107)((B1(I,J),J=1,IRI),I=1,N)

C** FOR.ATION OF THE NEW MATRIX BI

20 R=IZ(IJ)
20 B:FF(I,J)=B1(I,J)
20 Dn 6 I=1,N
20 Dn 6 J=1,IRI
21 Bi(I,J)=BUFF(I,J)
22 F0RMA T(4,107)WRITE(6,107)((B1(I,J),J=1,IRI),I=1,N)

C** FOR.ATION OF THE NEW MATRIX BI

25 R=IZ(IJ)
25 B:FF(I,J)=B1(I,J)
25 Dn 6 I=1,N
25 Dn 6 J=1,IRI
26 Bi(I,J)=BUFF(I,J)
27 F0RMA T(4,107)WRITE(6,107)((B1(I,J),J=1,IRI),I=1,N)

C** FOR.ATION OF THE NEW MATRIX BI

30 R=IZ(IJ)
30 B:FF(I,J)=B1(I,J)
30 Dn 6 I=1,N
30 Dn 6 J=1,IRI
31 Bi(I,J)=BUFF(I,J)
32 F0RMA T(4,107)WRITE(6,107)((B1(I,J),J=1,IRI),I=1,N)

C** FOR.ATION OF THE NEW MATRIX BI

35 R=IZ(IJ)
35 B:FF(I,J)=B1(I,J)
35 Dn 6 I=1,N
35 Dn 6 J=1,IRI
36 Bi(I,J)=BUFF(I,J)
37 F0RMA T(4,107)WRITE(6,107)((B1(I,J),J=1,IRI),I=1,N)

C** FOR.ATION OF THE NEW MATRIX BI

40 R=IZ(IJ)
40 B:FF(I,J)=B1(I,J)
40 Dn 6 I=1,N
40 Dn 6 J=1,IRI
41 Bi(I,J)=BUFF(I,J)
42 F0RMA T(4,107)WRITE(6,107)((B1(I,J),J=1,IRI),I=1,N)

C** FOR.ATION OF THE NEW MATRIX BI

45 R=IZ(IJ)
45 B:FF(I,J)=B1(I,J)
45 Dn 6 I=1,N
45 Dn 6 J=1,IRI
46 Bi(I,J)=BUFF(I,J)
47 F0RMA T(4,107)WRITE(6,107)((B1(I,J),J=1,IRI),I=1,N)

C** FOR.ATION OF THE NEW MATRIX BI

50 R=IZ(IJ)
50 B:FF(I,J)=B1(I,J)
50 Dn 6 I=1,N
50 Dn 6 J=1,IRI
51 Bi(I,J)=BUFF(I,J)
52 F0RMA T(4,107)WRITE(6,107)((B1(I,J),J=1,IRI),I=1,N)

C** FOR.ATION OF THE NEW MATRIX BI

55 R=IZ(IJ)
55 B:FF(I,J)=B1(I,J)
55 Dn 6 I=1,N
55 Dn 6 J=1,IRI
56 Bi(I,J)=BUFF(I,J)
57 F0RMA T(4,107)WRITE(6,107)((B1(I,J),J=1,IRI),I=1,N)

C** FOR.ATION OF THE NEW MATRIX BI

60 R=IZ(IJ)
60 B:FF(I,J)=B1(I,J)
60 Dn 6 I=1,N
60 Dn 6 J=1,IRI
61 Bi(I,J)=BUFF(I,J)
62 F0RMA T(4,107)WRITE(6,107)((B1(I,J),J=1,IRI),I=1,N)
INTEGER R

IF (IECR .GT. 2) WRITE (6, 100)

100 FORMAT (1X, 'SUBROUTINE MAINBA*)

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 MAINBA
RETURN
END
A-35

**COMPUTATION OF THE DETECTION SPACE OF BA**

**SUBROUTINE SIMILAR TO MAINS**

COMMON/MAIN/BA,CBI

**COM** COMMON/MANI/BA,BI,EPS,R

**COM** COMMON/TRAFF1,BUFF2,BUFF3,XMD

DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12)

**COM** DIMENSION A(12,12),B(12,12),C(12,12),BI(12,12)

**COM** DIMENSION IROW(12),ICOL(12)

**COM** INTEGER P

**COM** INTEGER PR

**COM** COMPUTATION OF C*BA

```fortran
  1 DO 1 I=1,P
       B(FF(I,1))=0.E0
  1 CONTINUE
  2 IF (IECR.GT.9) WRITE(6,100)( BUFF(I,1), I=1,P)
  100 FORMAT(4(I1X,'CBA=',F10.4))
```

**COM COMPUTATION OF D*BA**

```fortran
  5 CC=0.
  6 DO 6 I=1,P
       B(FF(I,2))=0.E0
  7 CC=CC+BUFF(I,1)**2
  8 CC=1./CC
       B(FF2(I,J))=-BUFF(I,J)*CC
  9 IF (IECR.GT.10) WRITE(6,102)( Buff2(I,J), J=1,P)
  102 FORMAT(4(I1X,'D2=',E10.4))
```

**COM COMPUTATION OF C'**

```fortran
  11 CC=0.
  12 DO 12 J=1,P
       B(FF(I,J))=0.E0
  13 B(FF2(I,J))=BUFF2(I,J)*CC
  14 IF (IECR.GT.10) WRITE(6,103)( Buff2(I,J), J=1,P)
  103 FORMAT(4(I1X,'D2=',E10.4))
```

**COM COMPUTATION OF A-0*DBA*C**

```fortran
  15 CC=0.
  16 DO 16 J=1,P
       B(FF(I,J))=0.E0
  17 CC=CC+BUFF(I,J)*CC
  18 CC=1./CC
       B(FF2(I,J))=-BUFF(I,J)*CC
  19 IF (IECR.GT.10) WRITE(6,104)( Buff2(I,J), J=1,P)
  104 FORMAT(4(I1X,'D2=',E10.4))
```

**COM**
9  B\cdot FF1(I,J) = A(I,J) - B\cdot FF2(I,J)
   IF (IECR \gt 5) WRITE(6,107)((3\cdot FF1(I,J), J=1,N), I=1,N)
107  F\cdot ORMAT(4(1x, *A-D8A*CE=,10.4))
C** COMPUTATION OF XMD=MD*
   DO 10 I=1,N
       DO 10 J=1,N
           XMD(I,J) = B\cdot FF(I,J)
10   B\cdot FF2(I,J) = B\cdot FF(I,J)
       NV = NV - 1
   END
14  L=1,P
   DO 9 I=1,N
       DO 9 J=1,N
           B\cdot FF3(I,J) = 0.0E0
91    I=I*L*P
   END
13  XMD(I,J) = B\cdot FF3(I,J)
14  CONTINUE
C~* ORTHOGONAL REDUCTION OF XMD
   NVMD = 144
   EPSII = 0.0001
15  I=1,P
   DO 15 J=1,N
       B\cdot FF(I,J) = 0.0E0
15   B\cdot FF(I,J) = 1.0E0
   IF (IECR \gt 3) WRITE(6,108)((3\cdot FF(I,J) , J=1,N), I=1,N)
108  F\cdot ORMAT(4(1x,*XMD=*,10.4))
C** ORTHOGONAL REDUCTION OF RK OMEGA AND LINEAR DEP
   I=MD=144
   EPSII = 0.0001
16  I=1,N
   DO 16 J=1,N
       B\cdot FF(I,J) = B\cdot FF(I,J)
16   INEG = 12
       CALL M\_F\_GR(BUFF, IOMEGR, N, N+IRANK, IROW, ICOL, EPS, IER)
   IF (IECR \gt 1) WRITE(6,110)(3\cdot FF(I,J) , J=1,N), I=1,N)
110  F\cdot ORMAT(4(1x,*OMEGA=*,10.4))
   IF (IECR \gt 1) WRITE(6,111)(*IER=*,I3)
111  F\cdot ORMAT(4(1x,*IER=*,I3)
   WRITE(6,112)(IROW(I), ICOL(I), I=1,N)
112  F\cdot ORMAT(4(1x,*DEP OF SA IS*,10.4))
   CALL MAIN8C(IRANK)
   RETURN
END
SUBROUTINE MAIN8C(IRANK)
C** COMPUTATION OF THE SUBSET OF EVENTS RI FROM WHICH BI IS
C** OUTPUT STATIONARY
C\_COMMON/MAN12/ABA,ABI,CAI,BANP,0,R,IECR,EPS,BUFF
C\_COMMON/MAN12/RA
C\_COMMON/TRASH2/BUFF1
D\_IMENSION A(12,12), R(12,12), B\_1(12,12), C(12,12)
DIMENSION Buffer(12,12),B(12),BUFF1(12,12)
DIMENSION IROW(12),ICOL(12)
INTEGER P,J,R
I=IECR, GT, 2) WRITE(6,100)
100 FORMAT(1X,'SUBROUTINE MAIN8C')
C** COMPUTATION OF CBA
DO 2 I=1,P
   B:FF(I,1)=0.00
   DO 2 J=1,N
2 B:FF(I,1)=BUFF(I,1)+C(I,J)*B(J)
C** COMPUTATION OF LINEAR DEPENDENCY OF CBA..CBI
C** IF n=11, THIS SUBROUTINE WON'T WORK
IF (R. NE. 11) GO TO 3
WRITE(6,101)
101 FORMAT (1X,'YOU CAN NOT TEST OUTPUT STATIONARITY WITH THIS',/1X,'SUBROUTINE.CHECK SOURCE')
STOP
3 CONTINUE
I=R+1
DO 4 I=1,P
4 B:FF(I,1)=BUFF(I,1)
DO 5 J=1,Q
5 B:FF(I,J)=CBI(I,J)
B:FF=12
EPSI=0.0001 E0
CALL MFG(R(BUFF,I,BUFF,P,I,I,RANK1,IROW,ICOL,EPSl,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(6,103) RANK1
103 FORMAT (1X,'RANK OF CBA IS ',I3)
IF (IECR.GT.2) WRITE(6,104)(IROW(I),ICOL(I),I=1,P)
104 FORMAT (1X,'RANK OF CBI IS ',I3)
IF (IECR.GT.2) WRITE(6,105)(BUFF(I,J),J=1,Q),I=1,P)
105 FORMAT (1X,'SOLVING FOR LINEAR DEPENDENCY IN PRECEDING',/1X,'matrix, you know the relation between CBA and other CRI',/1X,'it is dependent, see rules of thumb to see if it is',/1X,'advisable to do so')
RETURN
END
SUBROUTINE MAIN9

** COMPUTATION OF THE MATRIX TO (STEP 6 A)**

** COMMON/MAN?1/A,B,C,RI,CI,N,P,Q,R,IECR,EPS,BUFF**
** COMMON/MAN?2/ABCRICBIN,PQ,R,IECREPS,RUFF**
** COMMON/MAN?3/INU,INJS, INJO**
** COMMON/MAN?4/INU,C**
** COMMON/MAN?5/INU**
** COMMON/T RASH2/WSBUFF2**

** WHAT WAS BEFORE IN MANIAC, OMELOG IS OF NO INTEREST FROM NOW ON**

** DIMENSION A(12,12),R(12,12),C(12,12),BI(11,12),CBI(12,12)**
** DIMENSION UFF(12,12),W(1212),IZ(12),TO(1212),ADSC(12,12)**
** DIMENSION INU(12),POG(12,12),BUFF2(12,12),IROW(12),ICOL(12)**
** INTEGER PR**
** IF(IECR.GT.2)WRITE(6,100)***
** FORMAT(1X,"SUBROUTINE MAIN9")**
** CONTINUE**
** WRITE(6,101)***
** 100 FORMAT(1X,"SUBROUTINE MAIN9")**
** 1 CONTINUE**
** WRITE(6,102)***
** 101 FORMAT(1X,"TYPE THE NUMBER OF RANKS OF CS WHICH HAVE A*1X,1X,ON-ZERO AUXILIARY VECTOR IN THE ORTHOG REDUCTION OF XMD",1X,"STARTING WITH THE IDENTITY MATRIX")**
** IF(DNAT.E.0) GO TO 2**
** CONTINUE**

** ENTERING THE NUMBER OF WR**

** READ(5,*11)II**
** WRITE(6,103)II**
** 103 FORMAT(1X,"IF DATA ARE CORRECTLY ENTERED TYPE 11, OTHERWISE TYPE 0")**
** READ(5,102)ISIGN**
** IF(ISIGN.EQ.0) GO TO 2**
** IF(II.EQ.0) GO TO 9**
** CONTINUE**

** ENTERING THE WR**

** Do N I=1,N**
** WRITE(6,104)I,II**
** 104 FORMAT(1X,"TYPE WI('+',13,''), FOR I=1,+',I3)"**
** READ(5,*11)W(I,J),J=1,II**
** WRITE(6,105)W(I,J),J=1,II**
** 105 FORMAT(1X,"(W(I,J),J=1,II),I=1,N)")**
** CONTINUE**
** WRITE(6,106)W(I,J),J=1,II,N**

** ENTERING THE EXPONENT ASSOCIATED WITH THE WR**

** READ(5,102)ISIGN**
** IF(ISIGN.EQ.0) GO TO 3**
** CONTINUE**
** WRITE(6,107)W(I,J),J=1,II,N**

** COMPUTATION OF ((A-DS*C)**IZ(I-1))*W(.I)**

** CALL MAIN9A(AOSC,IZ,WSBUFF,W)**

** COMPUTATION OF TO**

** DO 7 I=SUM(IZ(I))**
** 7 I=SUM(IZ(I))
C*** ISUW IS THE COLUMN DIMENSION OF W

**
Do 8 I=1,N

8    B**FF2(I,J)=W(I,J)
I=12
E$$S=0.0001FL0
CALL M**GR(W,IN=N,I8M,IRANK,IROW,ICOL,EP$$,IER)
WRITE(6,112) IER

112  FORMAT(1X,'ERROR CODE IN COMPUTATION OF RK OF W IS',IER)

113  FORMAT(1X,'RK OF ADSCW IS',IER)

114  FORMAT(1X,'I8M,IRow(1),ICOL(1),I=1,N)

115  FORMAT(1X,'TO=',IER)

116  FORMAT(1X,'TO=',IER)

C*** SELE$$ION OF IND COLUMNS OF W
CALL MAIN98(BUFF2,N,IRANK,IECR)

C** TEST TO CHECK THAT BUFF2 HAS N-NUS COLUMNS

108  IF (IECR.GT.1) WRITE(6,108) INUO

109  IF (IECR.GT.1) WRITE(6,109) INUS

110  IF (IECR.GT.1) WRITE(6,110) INUS

111  IF (I$$EQ.0) IRANK=0

112  IM=IRANK+INUS

13  IF (I$$EQ.0) IRANK=0

113  IF (I$$EQ.0) IRANK=0

114  IF (DIM.EQ.N) GO TO 14

115  FORMAT(4(1X,'TO=',IER))

116  RETURN
END

SUBROUTINE MAIN9A(ADSC,ZWN,BUFF)

C** Computes ((A-OS*C)**IZ(tr-I)*W() in AN16E was TETA, OF NO INTEREST FROM NOW ON

C*** Computes ((A-OS*C)**IZ(I-1)**W(I)

CMMOM/MAN**E/BUFF1

C** Use BUFF1 AS A WORKING AREA

DIMENSION ADSC(12,12),IW(12,12),W(12,12),BUFF(12,12),BUFF1(12,12)

117  FORMAT(1X,'RANK',IER)

118  FORMAT(1X,'I8M,IRow(1),ICOL(1),I=1,N)

119  FORMAT(1X,'TO=',IER)

120  FORMAT(1X,'TO=',IER)

121  RETURN
END

SUBROUTINE MAIN9A(ADSC,ZWN,BUFF)

C** Computes ((A-OS*C)**IZ(tr-I)*W() in AN16E was TETA, OF NO INTEREST FROM NOW ON

C*** Computes ((A-OS*C)**IZ(I-1)**W(I)

CMMOM/MAN**E/BUFF1

C** Use BUFF1 AS A WORKING AREA

DIMENSION ADSC(12,12),IW(12,12),W(12,12),BUFF(12,12),BUFF1(12,12)

117  FORMAT(1X,'RANK',IER)

118  FORMAT(1X,'I8M,IRow(1),ICOL(1),I=1,N)

119  FORMAT(1X,'TO=',IER)

120  FORMAT(1X,'TO=',IER)

121  RETURN
END

SUBROUTINE MAIN9A(ADSC,ZWN,BUFF)

C** Computes ((A-OS*C)**IZ(tr-I)*W() in AN16E was TETA, OF NO INTEREST FROM NOW ON

C*** Computes ((A-OS*C)**IZ(I-1)**W(I)

CMMOM/MAN**E/BUFF1

C** Use BUFF1 AS A WORKING AREA

DIMENSION ADSC(12,12),IW(12,12),W(12,12),BUFF(12,12),BUFF1(12,12)

117  FORMAT(1X,'RANK',IER)

118  FORMAT(1X,'I8M,IRow(1),ICOL(1),I=1,N)

119  FORMAT(1X,'TO=',IER)

120  FORMAT(1X,'TO=',IER)

121  RETURN
END

SUBROUTINE MAIN9A(ADSC,ZWN,BUFF)

C** Computes ((A-OS*C)**IZ(tr-I)*W() in AN16E was TETA, OF NO INTEREST FROM NOW ON

C*** Computes ((A-OS*C)**IZ(I-1)**W(I)

CMMOM/MAN**E/BUFF1

C** Use BUFF1 AS A WORKING AREA

DIMENSION ADSC(12,12),IW(12,12),W(12,12),BUFF(12,12),BUFF1(12,12)

117  FORMAT(1X,'RANK',IER)

118  FORMAT(1X,'I8M,IRow(1),ICOL(1),I=1,N)

119  FORMAT(1X,'TO=',IER)

120  FORMAT(1X,'TO=',IER)

121  RETURN
END

SUBROUTINE MAIN9A(ADSC,ZWN,BUFF)

C** Computes ((A-OS*C)**IZ(tr-I)*W() in AN16E was TETA, OF NO INTEREST FROM NOW ON

C*** Computes ((A-OS*C)**IZ(I-1)**W(I)

CMMOM/MAN**E/BUFF1

C** Use BUFF1 AS A WORKING AREA

DIMENSION ADSC(12,12),IW(12,12),W(12,12),BUFF(12,12),BUFF1(12,12)

117  FORMAT(1X,'RANK',IER)

118  FORMAT(1X,'I8M,IRow(1),ICOL(1),I=1,N)

119  FORMAT(1X,'TO=',IER)

120  FORMAT(1X,'TO=',IER)

121  RETURN
END

SUBROUTINE MAIN9A(ADSC,ZWN,BUFF)

C** Computes ((A-OS*C)**IZ(tr-I)*W() in AN16E was TETA, OF NO INTEREST FROM NOW ON

C*** Computes ((A-OS*C)**IZ(I-1)**W(I)

CMMOM/MAN**E/BUFF1
A-40

1. BiFF(I, INC) = W(I, IND)
2. BiFF(I, IND) = W(I, IND)
3. Inc = INC + 1
4. K% = IZ(IND) - 1
5. If (K% < 0) GO TO 4
6. Do 4 K = 1, INC
7. W(I, IND) = 0, E0
8. Do 2 J = 1, N
10. K = K% + K%
11. If (K% < 0) GO TO 4
12. Do 4 INC = INC + 1
13. Do 3 I = 1, N
15. Do 3 J = 1, INC
17. Do 5 I = 1, N
18. Do 5 J = 1, INC
20. C = C + 1
21. Continue
22. Do 6 I = 1, N
23. Do 6 J = 1, INC
24. W(I, IND) = BUFE(I, J)
25. If (IEEI*3 > 5) WRITE(6, 102)((W(I, J), J = 1, INC), I = 1, N)
26. Write(6, 102) ((W(I, J), J = 1, INC), I = 1, N)
27. RETURN

SUBROUTINE MAIN9B(BUFF2, IRANK, IECD)

C** SELEClON OF INDEPENDENT COLUMNS OF W

1. COMMON/MANT3/ICOL
2. COMMON/MANT6E/BUFF
3. INTEGER P, Q, R
4. If (IEEIC > 0) WRITE(6, 100)
5. WRITE(6, 101)
6. IF DATA ARE INCORRECTLY ENTERED, TYPE 00, OTHERWISE TIE 11, T* 104, ISIGQ
7. IF (ISIGQ * EQ. 0) GO TO 1
8. Do 1 I = 1, N
9. Do 3 J = 1, IRANK
10. BiFF(I, J) = BUFF(12, 12)
11. BiFF(I, J) = BUFF(I, J)
12. BiFF(I, J) = BUFF(I, J)
13. WRITE(6, 105)((BUFF2(I, J), J = 1, IRANK), I = 1, N)
14. RETURN

END
 SUBROUTINE MAIN9C

C** COMPUTES T AND T-1 (STEP 63)
C
COMMON/MAN9C/ A+B,C+BI,CBI+N,P0,R,IECR,EPS,BUFF
C
COMMON/MAN9C/T0,IRANK,II
C
COMMON/MAN9C/A/GG
C
COMMON/MAN9C/IEX INUS,INJO
C
COMMON/TRASH2/BUFF1,BUFF2,BUFF3,BUFF4
C
COMMON/MAN9C/OMEGC,DS,CS

DIMENSION (12,12),R(12,12),C(12,12),I(312,12),CBI(12,12)

DIMENSION BUFF(I2,12),TO(I2),GG(i2,12),BUFF(I2,12)

DIMENSION aUFF2(12,12),TU(12),OMEGC(12,12),CS(12,12)

DIMENSION )S(12,12),TM(12,12),TTM(12,12)

DIMENSION T(12,12),TT(12,12),BUFF4(12,12)

INTEGER P, R

IR(IEX.GT.2) WRITE(6,100)

100 FORMAT(1X,'SUBROUTINE MAIN9C')
C** COMPUTES GI,...,A**(INU(T)-1) FOR ALL GI
C CALL MA9CA(A,INU,N,IECR,GG,BUFF1,BUFF2,BUFF3,R)
C** BUF1=(G1,AG1,...,A**(INU(T)))*G1,G2,A*G2
C** BUF2=(A**(INU(I)-1)*G1,A**(INU(I)-1)*G2,...)

IEUM=0

DO 1 I=1,IECR

1 WRITE(6,101) ISUM

101 FORMAT(1X,'ISUM=',I3)

DO 2 J=1,ISUM

2 WRITE(6,102) (T(I,J),J=1,IN)

20 FORMAT(1X,'T=',EIO.4)

C** TEST TO CHECK THAT T HAS N COLUMNS

INU=INU-INUA

IF(INU.EQ.INUA) GO TO 5

5 WRITE(6,103) (T(I,J),J=1,IN)

103 FORMAT(1X,'T DOES NOT HAVE N COLUMNS',/,'HERE IS A MISTAKE,CHECK THE USER GUIDE')

STOP

C** COMPUTATION OF T-1=TT

IF(IECR.EQ.5) WRITE(6,104) (T(I,J),J=1,N),I=1,N)

104 FORMAT(1X,'T-1=TT',/,'HERE IS A MISTAKE,CHECK THE USER GUIDE')

STOP

IF(IECR.GE.6) WRITE(6,105) (T(I,J),J=1,N),I=1,N)

105 FORMAT(1X,'T-1=TT',/,'HERE IS A MISTAKE,CHECK THE USER GUIDE')

STOP
CALL LINV2F(T*N,T*TT,TT*IDG,TW*KAREA,IER)

Dn 67 I=1,N
Dn 67 J=1,N

67 T(I,J)=BUFF4(I,J)
I=IECR*(T,1) WRITE(6,105) IER, IDG

105 FORMAT(I*ERROR CODE IN COMPUTATION OF T-1,13,13,13)

I=IECR*(T,5) WRITE(6,106) (TT(I,J),J=1,N),I=1,N)

106 FORMAT(4(I*TT=,10.4))
IOT=II*R
IF(IOT.T.EQ.P) GO TO 7
WRITE(6,107)

107 FORMAT(1X,'ERROR CODE FOR T-1 COMPUTATION, I3, I3)

Ir(IECR.GT.1) WRITE(6,108) (TTM(I,J),J=1,N),I=1,N)

108 FORMAT(4(I*X=,10.4))
RETURN
END

C* COMUTATION OF TM

CALL MA9CB(C,R,N,BUFF1,BUFF2,BUFF3,TM,IECR)

C* COMPUTATION OF TM-1

Do 78 I=1,P
Do 78 J=1,P

78 T4(I,J)=BUFF4(I,J)
CALL LINV2F(TM,P,TTM,TTM,IDG,KAREA,IER)

Do 78 I=1,P
Do 78 J=1,P

78 T4(I,J)=BUFF4(I,J)
IF(IECR.GT.1) WRITE(6,107) IER, IDG

107 FORMAT(1X,'ERROR CODE FOR T-1 COMPUTATION, I3, I3)

Ir(IECR.GT.1) WRITE(6,108) (TTM(I,J),J=1,P),I=1,P)

108 FORMAT(4(I*X=,10.4))
RETURN
END

SUBROUTINE MA9CA(A,INU,GEGG,BUFF1,BUFF2,BUFF3,R)

C* COMPUTES Gi,...,A**(INU(I)-1) FOR ALL Gi

DIMENSION A(12,12),INU(12),GG(12,12)
DIMENSION BUFF1(12,12),BUFF2(12,12)
DIMENSION BUFF3(12,12)
INTEGER P,Q,R
I=IECR*(G,1) WRITE(6,100)

100 FORMAT(1X,'SUBROUTINE MA9CA')
IF(IECR.GT.12) WRITE(6,101) (INU(I),I=1,R)

Do 10 II=1,N
Do 10 I=1,N

10 BiFF1(I,II)=GG(I,II)
BiFF2(I,II)=GG(I,II)
IND=1
K=INU(II)-1
IF(K.EQ.0) GO TO 5
Do 2 K=1,K
Do 2 I=1,N

2 BiFF3(I,II)=BUFF2(I,II)
Do 3 I=1,N
BiFF2(I,II)=0.E0
Do 3 J=1,N

3 BiFF2(I,II)=BUFF2(I,II)+A(I,J)*BUFF3(J,II)
Do 4 I=1,N
BiFF1(I,IND)=BUFF2(I,II)
IND=IND+1

4 BiFF1(I,IND)=BUFF2(I,II)
Do 5 IND=IND+1

5 CONTINUE
CONTINUE

6 CONTINUE

101 FORMAT(8(I*INU=,13))
IF(IECR.GT.5) WRITE(6,102) ((BUFF2(I,J),J=1,R),I=1,N)

102 FORMAT(4(I*X=,10.4))
IN=IND-1
IF (IECR.GT.5) WRITE(6,103)((BUFF(I,J)+J=1,IND),I=1,N) 
103 FORMAT(4(1X,'BUFF=',F10.4)) 
C** BUF2=(GS,GI,...,NU(J)-1)*GS,GI,...,NU(I)-1)*GS 
C** BUF2=(A**(I-NU(I)-1)*GI,A**(I-NU(I)-1)*S2 
RETURN 
END 
SUBROUTINE MACS(C,P,N,BUFF,BUFF1,BUFF2,TM,IECR) 
C** COMPUTES TM 
DIMENSION C(1,N),BUFF1(1,N),BUFF(1,N),BUFF2(1,N) 
DIMENSION TM(1,N),P(1,N) 
IF (IECR.GT.2) WRITE(6,100) 
100 FORMAT(1X,'SUBROUTINE MACS*) 
C** BUF2=(A**(I-NU(I)-1)*GI,A**(I-NU(I)-1)*S2 
C** BUF2=(A**(I-NU(I)-1)*GI,A**(I-NU(I)-1)*S2 
IF (IECR.GT.12) WRITE(6,101)((BUFF2(TJ),J=1,R),I=1,N) 
101 FORMAT(4(1X,'BUFF2=','F10.4)) 
IF (IECR.GT.12) WRITE(6,102)((BUFF(IJ),J=1,R),I=1,N) 
102 FORMAT(4(1X,'BUFF=','F10.4)) 
IF (IECR.GT.12) WRITE(6,103)((CS(IJ),J=1,N),I=1,P) 
103 FORMAT(4(1X,'CS=','F10.4)) 
C** T=C*BUFF1+CS*BUFF 
DN 1 I=1,P 
DN 1 J=1,R 
TM(I,J)=0.E0 
DN 1 K=1,N 
1 TM(I,J)=TM(I,J)+C(K)*BUFF2(K,J) 
IF (I.EQ.0) GO TO 4 
DN 2 I=1,P 
DN 2 J=1,N 
BUFF1(I,J)=0.E0 
DN 2 K=1,N 
2 BUFF1(I,J)=BUFF1(I,J)+CS(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=','F10.4)) 
IF (IECR.GT.9) WRITE(6,105)((TM(I,J),J=1,R),I=1,P) 
105 FORMAT(4(1X,'TM2=','F10.4)) 
DN 3 I=1,P 
DN 3 J=1,N 
TM(I,J)=BUFF1(I,J) 
3 CnNTINUE 
4 CN=NU 
J'=II+J 
IF (IECR.GT.5) WRITE(6,106)((TM(I,J),I=1,NN),J=II,JJ) 
106 FORMAT(4(1X,'TM=',F10.4))
SUBROUTINE MAIN9D
C** COMPUTES AH, BH, CH, DH (STEP 60)
COMMON/MAN17/A, B, C, BI, CB1, N, P, Q, R, IE, EPS, BUFF
COMMON/MAN19A/T, TT, TTM, T4
COMMON/MAN14/AH, BH, CH
C** IN WAVE WERE OMEGCA, DSA, CS, OF NO INTEREST FROM NOW ON
DIMENSION A(12,12), B(12,12), C(12,12), BI(12,12), CB1(12,12)
DIMENSION PUF, T(12,12), TT(12,12), TTM(12,12)
DIMENSION AH(12,12), BH(12,12), CH(12,12)
INTEGER P, Q, R
C** COMPUTATION OF AH
IF (IE<CR.GT.2) WRITE(6,100)
100 FORMAT(1X, 'SUBROUTINE MAIN9D')
101 FORMAT(12, 12 E10.4)
102 FORMAT(12, 12 E10.4)
103 FORMAT(12, 12 E10.4)
104 FORMAT(1X, 'A HAT = ')
105 FORMAT(1X, 'B HAT = ')
106 FORMAT(1X, 'C HAT = ')
C** COMPUTATION OF BH
IF (IE<CR.GT.12) WRITE(6,105)
105 FORMAT(12, 12 E10.4)
106 FORMAT(12, 12 E10.4)
C** COMPUTATION OF CH
I=(IE<CR.GT.12) WRITE(6,107)
107 FORMAT(12, 12 E10.4)
108 FORMAT(12, 12 E10.4)
RETURN
END
IF (IECR.GT.0) WRITE (6,109)
109 FORMAT (1X,C30)
I=IP
IF (IECR.GT.0) WRITE (6,600) (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,0,R,IECR,EPS,BUFF
COMMON/MANI5/INU,INUS,INUJ
COMMON/MANI5A/DFR
C** IN 4ANI6A WAS ADSC OF NO INTEREST FROM NOW ON
DIMENSION A(12,12),R(12,12),C(12,12),BI(12,12),CBI(12,12)
DIMENSION RUFF(12,12),INU(12)
DIMENSION AH(12,12)
INTEGER IP
I=IECR.GT.0) WRITE(6,100)
100 FORMAT (1X,'SUBROUTINE MAIN9E')
I=IECR.GT.0) WRITE(6,601) (AH(I,J),J=1,N),I=1,N)
CALL MAIN9F
RETURN
END
SUBROUTINE MAIN9F
C** Computes THE POL COEFF
COMMON/MANI2/A,B,C,BI,CBI,N,P,0,R,IECR,EPS,BUFF
COMMON/MANI5/INU,INUS,INUJ
COMMON/MANI5A/XP
COMMON/TRASH2/BUFF
C** IN 4ANI6A WAS ADSC OF NO INTEREST FROM NOW ON

*DIMENSION A(12,12),B(12,12),C(12,12),3I(12,12),CBI(12,12)  
*DIMENSION UFF(12,12),INJ(12),XP(12,12),BUFFI(12)  
INTEGER P,R  

C** REAdS THE EIGENVALUES DESIRED FOR EACH DETECTION SPACE  
IF(IECR.GT.2)WRITE(6,100)  
100 FORMAT(1X,*SUBROUTINE MAIN9*)  
D= 8 II=1,R  
N=INU(II)  
WRITE(6,101)N  
   100 FORMAT(1X,'TYPE THE DESIRED, EIGENVALUES FOR THE',/1X,  
   101 FORMAT(5,13,E11.6),BUFFI(I),I=NI)  
   102 FORMAT(1X,'IF DATA ARE CORRECTLY ENTERED TYPE 11. OTHERWISE TYPE0')  
   103 FORMAT(12)  
   I=SIGN  
   IF(I.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')  
   111/*SEE SOURCESUBROUTINE MAIN9*  
   2 CONTINUE  
   INI=NI+1  
   D=2 INI=N1  
   B=UFF(I)=0,E0  
   X0(1,II)=0,E0  
   ON 4 I=1,INI  
   4 X0(1,II)=XP(1,II)+BUFF(I)  
   X0(2,II)=0.0  
   ON 5 I=2,4  
   5 X0(2,II)=X0(2,II)*BUFFI(I)  
   ON 6 I=3,4  
   6 X0(2,II)=X0(2,II)*BUFFI(2)  
   X0(2,II)=X0(2,II)*BUFFI(3)  
   X0=BUFFI(4)  
   Y=BUFFI(3)*BUFFI(4)  
   Y'=A=Y*BUFFI(1)  
   Y'=B=Y*BUFFI(2)  
   X0(3,II)=XXA*XXB  
   X0(3,II)=X0(3,II)+YX0  
   X0(3,II)=X0(3,II)+YY0  
   X0(3,II)=X0(3,II)  
   X0(4,II)=1.0  
   ON 7 I=1,4  
   7 X0(4,II)=X0(4,II)*BUFFI(I)  
   ON 9 I=1,NI  
   ON 10 I=1,NI  
   9 CONTINUE  
   10 CONTINUE  
   IF(IECR.GT.5)WRITE(6,105)  
   105 FORMAT('POL COEF ARE')  

ID=R
I=1(I<CR.*GT.*5)WRITE(6,*)((XP(I,J)+J=1+IR),I=1,N)
CALL MAIN9G
RETURN
END
SiBROUTINE MAIN9G
C** COMPUTES DPSI
COMMON/MANI2/A,B,C,BI,CBI,N,P,Q,R,IECR,EPS,RAFF
COMMON/MANI4/AB,BA,CH
COMMON/MANI6A/XP
COMMON/MANI6B/DPSI
COMMON/MANI5/INU,INJS,INJO
C** IN MAIN9A WERE XMTETA, OF NO INTEREST FROM NOW ON
DIMENSION A(12,12),R(12,12),C(12,12),BI(12,12)
DIMENSION RUFF(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,N
DO 1 I=I+N
1 DCSI(IJJ)=0.E0
1A=0
JJ=JJ-1
DO 2 I=I+JJ
2 IA=IA+INU(I)
IT=IA+1
IF(JJJ.EQ.O)IA=1
C** IA IS THE ROW WHERE PI REGINS(STEP 6E)
IN=IN+INU(JJ)
IN=IN+B-1
IA=IA-1
II=II+1
3 DOSSI(I,JJ)=XP(II,JJ)*AH(II,IB)
4 CONTINUE
J=J+1
DO 5 J=JJ,P
5 DSSI(I,J)=0.E0
IF(IECR.GT.5)WRITE(6,101)
101 FORMAT(1X,'DPSI=(')
DO=P
IF(IECR.GT.5)WRITE(6,101)
CALL MAIN94
RETURN
END
SiBROUTINE 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,RAFF
COMMON/MANI4/AB,BA,CH
COMMON/MANI5A/DFR
COMMON/MANI6A/DPSI
COMMON/TRASH2/DO,DH,BUFF1
DIMENSION A(12,12),B(12,12),BI(12,12),C(12,12),CBI(12,12)
DIMENSION RUFF(12,12),TT(12,12),T4(12,12),T4(12,12),T4(12,12)
DIMENSION FRR(12,12),DPSI(12,12),O(12,12),DH(12,12),BUFF1(12,12)
DIMENSION DH(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,N
1 DDFI(I,J)=DFR(I,J)+DPSI(I,J)
I=1:ECR,GT,0) WRITE(6,110)  
101 FORMAT(1X,D'HAT=')  
102 WRITE(6,103) MAI00620  
C** CHEC: COMPUTES AH=CH*DH  
IF(I=ICR,LE,3) GO TO 4  
   DON 2 I=1,N  
   DON 2 J=1,N  
2 BiFF(I,J)=0.E0  
   DON 2 K=1,P  
3 BiFF(I,J)=BUFF(I,J)+DH(I,J)*CH(K,J)  
   DON 3 I=1,N  
   DON 3 J=1,N  
4 CONTINUE  
C** COMPUTES D=T*DH*TTM  
   DON 5 I=1,N  
   DON 5 J=1,P  
   BiFF(I,J)=0.E0  
   DON 5 K=1,P  
5 BiFF(I,J)=BUFF(I,J)+DH(I,J)*TTM(K,J)  
   DON 6 I=1,N  
   DON 6 J=1,P  
6 BiFF(I,J)=D(I,J)+T(I,J)*BUFF(I,J)  
   WRITE(6,103)  
103 FORMAT(1X,D=')  
   WRITE(6,104) MAI00630  
C** CHEC: COMPUTES A=D*C  
IF(I=ICR,LE,3)RETURN  
   DON 7 I=1,N  
   DON 7 J=1,N  
7 BiFF(I,J)=0.E0  
   DON 7 K=1,P  
8 BiFF(I,J)=BUFF(I,J)+D(I,J)*C(K,J)  
   DON 8 I=1,N  
   DON 8 J=1,N  
104 FORMAT(1X,A=D*C*)  
   WRITE(6,105) MAI00640  
RETURN  
E=0  
&SUBROUTINE MFGR(A,IA,4,N,IRANK,IROW,IROL,EPS,IER)  
IMENSION A(1),IROW(1),ICOL(1)  
DOUBLE PRECISION SAVE,HOLD,WORK,F1,F2  
NDER=IER  
IEP=0  
1 IF(M) 2,2,1  
2 IF(N) 2,2,3  
3 IEP=1000  
4 CO TO 44  
5 IEP=IA  
6 CA=1  
7 CO TO 6  
A-49
(1) \( \text{CA} = \text{IA} \)
(2) \( \text{IV} = 0.0 \cdot \text{E0} \)
(3) \( \text{C} = 1 \)
(4) \( \text{O} 9 \quad \text{J} = 1 \cdot \text{N} \)
(5) \( \text{COL} \cdot (\text{J}) = \text{J} \)
(6) \( \text{P} = \text{IC} \)
(7) \( \text{O} 8 \quad \text{I} = 1 \cdot \text{M} \)
(8) \( \text{EEK} = A \cdot (\text{IR}) \)
(9) \( \text{F} \cdot (\text{ABS} \cdot \text{SEEK} - \text{ABS} \cdot \text{PIV} \cdot 8 \cdot 8 \cdot 7) \)
(10) \( \text{IV} = \text{SEEK} \)
(11) \( \text{C} = \text{J} \)
(12) \( \text{R} = \text{IR} + \text{IRA} \)
(13) \( \text{C} = \text{IC} + \text{ICA} \)
(14) \( \text{O} 10 \quad \text{I} = 1 \cdot \text{M} \)
(15) \( \text{F} \cdot (\text{M} \cdot \text{N}) = 12 \cdot 12 \cdot 11 \)
(16) \( \text{I} = \text{N} \)
(17) \( \text{O} 27 \quad \text{J} = 1 \cdot \text{LIM} \)
(18) \( \text{F} \cdot (\text{ABS} \cdot \text{PIV} - \text{TOL}) = 28 \cdot 28 \cdot 13 \)
(19) \( \text{RANK} = \text{J} \)
(20) \( \text{F} \cdot (\text{NR} - \text{IRANK}) = 16 \cdot 16 \cdot 14 \)
(21) \( \text{M} = \text{JR} \)
(22) \( \text{R} = \text{IR} \cdot \text{IRA} \)
(23) \( \text{O} 15 \quad \text{I} = 1 \cdot \text{N} \)
(24) \( \text{EEK} = A \cdot (\text{IM}) \)
(25) \( \text{A} \cdot (\text{IM}) = A \cdot (\text{MC}) \)
(26) \( \text{A} \cdot (\text{MC}) = \text{SEEK} \)
(27) \( \text{M} = \text{IM} + \text{IRA} \)
(28) \( \text{F} \cdot (\text{NC} - \text{IRANK}) = 19 \cdot 19 \cdot 17 \)
(29) \( \text{M} = \text{JC} \)
(30) \( \text{NC} = \text{ICA} \cdot (\text{NC} - 1) + 1 \)
(31) \( \text{C} = \text{INC} \)
(32) \( \text{O} 18 \quad \text{I} = 1 \cdot \text{M} \)
(33) \( \text{EEK} = A \cdot (\text{IM}) \)
(34) \( \text{A} \cdot (\text{IM}) = A \cdot (\text{MC}) \)
(35) \( \text{A} \cdot (\text{MC}) = \text{SEEK} \)
(36) \( \text{M} = \text{IM} \cdot \text{IRA} \)
(37) \( \text{C} = \text{MC} + \text{IRA} \)
(38) \( \text{N} = \text{ICOL} \cdot (\text{NC}) \)
(39) \( \text{ICOL} \cdot (\text{NC}) = \text{ICOL} \cdot (\text{IRANK}) \)
(40) \( \text{ICOL} \cdot (\text{IRANK}) = \text{IN} \)
(41) \( \text{AVE} = \text{PIV} \)
(42) \( \text{IV} = 0.0 \cdot \text{E0} \)
(43) \( \text{J} = \text{J} + 1 \)
(44) \( \text{R} = \text{JD} \)
(45) \( \text{C} = \text{JD} + \text{ICA} \)
(46) \( \text{J} = \text{J} \)
(47) \( \text{F} \cdot (\text{I} - \text{M}) = 21 \cdot 26 \cdot 26 \)
21 \*i = i+1
\*r = \*r + \*r
old = a(\*r)
old = old/save
\*a(\*r) = old
\*rc = \*r + \*ica
\*c = \*c
\*k = j
22 \*f(k-n) = 23, 20, 20
23 \*k = k+1
\*e1 = (krc)
\*e2 = a(kc)
\*work = f1 - f2*hold
\*a(krc) = work
\*f(abs(work) - abs(piv)) = 25, 25, 24
24 piv = work
\*r = i
\*c = k
25 \*rc = krc + \*ica
\*c = kc + \*ica
\*o to 22
26 \*j = jc + \*ica
\*r = jr + \*ica
27 \*d = jd + \*ida
28 if(rank-4) = 29, 34, 34
29 \*r1 = rank-1
\*r1 = rank+1
\*j = jd + \*ida
\*r = jr + \*ica
30 \*f(j) = 31, 34, 31
31 \*r = jr
\*c = jc + \*ica
\*j = jc + \*ira
\*j = j+1
\*o to 33
32 i = ir1+m
\*r = ir
\*c = jc
\*work = 0.0
\*o to 32
33 k = j1 + rank
\*work = work+a(kr)*a(kc)
\*r = kr + \*ica
34 \*c = kc + \*ira
\*a(j) = a(ij)*work
35 \*j = j1 + \*ira
\*j = j-1
\*o to 30
36 \*f(rank-4) = 35, 43, 43
37 \*c = jc
\*i = jr + \*ica
\*d = jd + \*ida
\*o = 2
\*i = ir1+n
\*r = jr
\*c = 1c
\*work = 0.0
\*k = j
39 IF(K=IRANK)40,41,41
40 F1=A(KR)
  F2=A(KC)
  WORK=WORK+F1*F2
  KC=KC-ICA
  JR=KR-IRA
  K=K+1
  GO TO 39
41 IC=IC+ICA
  A(JI)=-(A(JI)*WORK)/A(JJ)
42 I=JI+ICA
  JR=JR-IRA
  J=J-1
  GO TO 36
43 CONTINUE
44 CONTINUE
45 CONTINUE
46 RETURN
END
SUBROUTINE ORTRED(DEP1, ARR, NREAL, *DEP1, EPSII, IEPR)
DIMENSION DEP1(1), DEP(14, 12), ARR(12, 12), WRED(12)
COMMON/ TRASH2/ WW
=IDEP1*NREAL
*IF(IECR GT. 12) WRITE(6, 1004)(DEP1(I)*I=1,N)
+10 0 10 K=1, N
1 = (K-1)/IDEP1
2 = K-1*IDEP1
+104 0 IF(IECR GT. 12) WRITE(6, 1005) I, J
105 0 FORMAT(1X, *EP1=, E10, 4)
10 0 IF(IECR GT. 12) WRITE(6, 1006) DEP1(I)
1000 0 FORMAT(5X, *DEP1=, E10, 4)
101 0 IF(IECR GT. 12) WRITE(6, 1007) (DEP(I)*I=1,N)
1001 0 IF(IECR GT. 12) WRITE(6, 1008) (ARR(I)*I=1,N)
10000 0 IF(IECR GT. 12) WRITE(6, 1009) (WW(I)*I=1,N)
100000 0 IF(IECR GT. 12) WRITE(6, 1010)
1000000 0 IF(IECR GT. 12) WRITE(6, 1011) (WRED(I)*I=1,N)
10000000 0 IF(IECR GT. 12) WRITE(6, 1012) (APR(I)*I=1,N)
100000000 0 IF(IECR GT. 12) WRITE(6, 1013) (WW(I)*I=1,N)
1000000000 0 IF(IECR GT. 12) WRITE(6, 1014) (WRED(I)*I=1,N)
10000000000 0 IF(IECR GT. 12) WRITE(6, 1015) (ARR(I)*I=1,N)
100000000000 0 IF(IECR GT. 12) WRITE(6, 1016) (WW(I)*I=1,N)
1000000000000 0 IF(IECR GT. 12) WRITE(6, 1017) (WRED(I)*I=1,N)
10000000000000 0 IF(IECR GT. 12) WRITE(6, 1018) (ARR(I)*I=1,N)
100000000000000 0 IF(IECR GT. 12) WRITE(6, 1019) (WW(I)*I=1,N)
IF(ABS(ARR(I,J)).LE.EPSI1) ARR(I,J)=0.
CONTINUE
RETURN
END
APPENDIX B

LISTING OF LONGITUDINAL GUIDEWAY VEHICLE SIMULATION
SIMULATION PROGRAM - LCV DETECTION FILTER STUDY

IMPLICIT REAL(A-H,O-Z)
DIMENSION C(3,7),D(3,5)
DIMENSION X(T(7,1),VCM(9,1),C4T(9,1),VMAINQ(9,1)
DIMENSION AD1(7,1),AD2(7,1)
VAR(7,1),Y(7,1),YH(7)
VAR(3,1),G(3,1)
COMMON/A1/, A(7,7), B(7,5), U(5,1)
REAL PR, DT, TM
DIMENSION W(7,9), CC(24)

ALL .IVITS MUST BE INPUT IN FPS STANDARD SYSTEM
FT - LR - SEC

SPECIFY SYSTEM CONSTANTS

EXTERNAL FCT
NEN=7
TOL=0.1
IJD=1
WRITE(6,101)
READ(5,201) P1
ANVEST=350.0E0
G=32.2E0
TFAIL=10000.
ICOMD=0
A'V=373.0E0
A1=1500.0E0
A2=1000.0E0
A3=1250.0E0
A4=1.0E0
A5=1.0E0
A6=1250.0E0
A7=0.55.5E0
A8=1466.5E0
A9=12150.0E0
CV=100.0E0
A'MAX=6.44E0
A'MAX=8.05E0
IFL=U
I'LAG3=0
V9=0.0E0
INUMM=0
WT=7.333333E0

SEED FOR NOISE
IS00=314100
AFROC=0.03E0
5W=AFROC
DN=1.0E0

INITIAL CONDITIONS

WRITE(6,103)
READ (5,202) X
WRITE (6,104)
READ (5,202) VC
WRITE (6,105)
READ (5,201) ICHV
IF (ICHV.EQ.0) GO TO 12
WRITE (6,104)
DO 12 J=1,ICHV
READ (5,*) CT(J), VCM(J)
12 CONTINUE
WRITE (6,107)
READ (5,*) DT
IF (DT.EQ.0.) DT=0.1
WRITE (6,108)
READ (5,201) ICHW
IF (ICHW.EQ.0) GO TO 13
WRITE (6,109)
DO 13 J=1,ICHW
READ (5,*) CT(J), VWIN(J)
13 CONTINUE
WRITE (6,210)
READ (5,*) ICOMP
IF (ICOMP.EQ.0) GO TO 14
WRITE (6,211) ICOMP
READ (5,*) TF, FAILVL
14 CONTINUE
WRITE (6,212)
READ (5,*) IMP
PRINT BACK THIS INFORMATION
WRITE (6,213) XX
WRITE (6,211) VC
IF (ICHW.EQ.0) GO TO 15
DO 15 J=1,ICHW
WRITE (6,212) CT(J), VCM(J)
15 CONTINUE
WRITE (6,213) DT
IF (ICHW.EQ.0) GO TO 14
DO 14 J=1,ICHW
WRITE (6,214) CT(J), VWIN(J)
14 CONTINUE
X(1)=XX
X(2)=VC/A
X(3)=VC
VOL=0.
V2COMMM=100.E0
T=0.E0
T=TIM+DT

INITIALIZE DETECTION STATE XMEs(5,1)
X:1=VC/A
X:2=0.E0
X:3=0.E0
X:4=0.E0
X:5=0.E0
X:6=CV

LEAD VEHICLE DYNAMICS
FORM DETECTION MODEL DESIGN MATRICES

CALL SOLUT(6)
CALL MODEL(AMV,AMVEST,A2,A3,A4,AK1,AK2,AK3,AK4)

COMPUTE A MATRIX

DO 1 J=1,7
DO 1 K=1,7
A(J,K)=0.000
1 CONTINUE

CONTINUE

A(1,2)=1.000
A(2,3)=1.000/AMV
A(3,4)=1.000
A(4,5)=1.000
A(5,2)=AK1*AK4*AK2
A(5,3)=AK4
A(5,4)=AK2
A(5,5)=AK2
A(5,6)=AK2
A(5,7)=AK1
A(6,6)=AK4
A(6,7)=1.000

COMPUTE B MATRIX

DO 2 J=1,7
DO 2 K=1,5
B(J,K)=0.000
2 CONTINUE

B(2,1)=-CV/AMV
B(2,3)=-B1/AMV
B(2,4)=-1.000
B(5,2)=AMVEST*AK1
B(5,3)=AK1
B(5,4)=AK4
B(6,2)=DN

COMPUTE C MATRIX

DO 3 J=1,3
DO 3 K=1,7
C(J,K)=0.000
3 CONTINUE

C(1,2)=-AK1*AK4
C(1,3)=AK2
C(1,7)=AK1
C(2,2)=AK4
C(3,7)=1.E0

COMPUTE D MATRIX

DO 4 J=1,3
DO 4 K=1,5
D(J,K)=0.000
4 CONTINUE
Y(J+1)=0.02*0
CONTINUE
D(1,i)=4NWST
D(1,2)=AK1*A4
D(1,3)=AK4
D(2,5)=AK4
IF (IPF, EQ, 1) WRITE (6, 218)
IF (IPF, EQ, 1) CALL MUMP (A=7, 7)
IF (IPF, EQ, 1) WRITE (6, 219)
IF (IPF, EQ, 1) CALL MUMP (A=7, 5)
IF (IPF, EQ, 1) WRITE (4, 220)
IF (IPF, EQ, 1) CALL MUMP (A=3, 7)
IF (IPF, EQ, 1) WRITE (6, 221)
IF (IPF, EQ, 1) CALL MUMP (D=3, 5)

C C

NAVIGATION LOOP

CONTINUE

C

COMPUTE NEW INPUT TO NON-LINEAR SYSTEM

FOLLOWER MODE SENSORS COMPUTATIONS

DXP=DX
XV=M+V+9*VT
DX=V+X(1,1)

C

VARIABLE GAIA G1 AND FOLLOWER COMMAND VELOCITY

G1=0.2
VCOMM=G1*VX

C

VELOCITY PROFILER

IF (ICHV, EQ, 0, AND, TM, LT, DT*3) VCOMM=VC
IF (ICHV, EQ, 0, AND, TM, LT, DT*3) GO TO 22
DO 22 J=1, ICHV
IF (DANS, TM, CT(J, 1), LF, DT*5) VCOMM=VC(J, 1)
CONTINUE

C

TAKE SMALLER VELOCITY

IF (VCOMM, LE, VCOMM) VCOMM=VCOMM
IF (VCOMM, LE, VCOMM) VCOMM=VCOMM
IF (ABS(VCOMM-VCOMM), LE, 0.1) IFL=0
IF (VCOMM, LE, VCOMM)+IFL=2
IF (((I7M, IQM)*1)MOD, F1, 1) WRITE (5, 222) (XWES(J)+J, 1, A, +)
YM=FS(1), YM=FS(2), EPS1(1), EPS1(2)
CALL PROFL (VCOMM, AXMA, AXMA, DT, AC+VC+XX*IFL, IMD)
VOLD=VCOMM

C

WIND GUST MODEL

IF (ICHW, EQ, 0) VM=0, 0, 0
IF (ICHW, EQ, 0) GO TO 20
IF (ICHW, EQ, 1) VM=WIND(1, 1)
IF (ICHW, EQ, 1) GO TO 20
K=ICHW-1
DO 20 J=1, K

CONTINUE
IF(TM.GT.C T(J,1).AN .. fLT.CH(J+.1 ))V=V',IND(Jol) TES02440

TRAC' SL(PET MODL TES070TES070

CALL TSTTP<(X(I,1),GN,SL3PE9) TES050

PANO0'-NqR FTjrpTO-GuS144 DSTRIRJUTION TES02050

ZERO EAN -VAIANCE 1./100. TES0?50

WT=O.0OF( ISEED)/100. TES02530

INPUT VECTOR U

IF(X(2,1).EQ.0.0)U(1,1)=0.0E 0 TEF502 5 70

IF(X(2,1).GE.0.0)U(1,1)=X(2,1)/ABS(X(2,1))

U(2,1)=AC

U(3,1)=(X(2,1)+V*+2

U(4,1)=GN*SLPF TES02580

NOISE INPUT

U(5,1)=WT TES02590

VINDAN4 (X(2,1) +WT)

S)OLUTION TO STATE EQUATION

SOLUTION TO STATE EQUATION

CALL FINDIF(A7.39,6.XJ>J+1AD1,AD2)

THM=THM+ITTM TES02710

THM=THM+ITTM TES02720

FAILRE T~c)LEM;7*jTATI04TES03040

,EP1) TES02800

FAIL RE IMPLEMENTATION
B-7

C

IF (ICOMP.EQ.0) GO TO 30

TEMP=TM

IF (AH=TFAIL-TIE P) .GT. DT*5) GO TO 30

IF (ICOMP.EQ.1) VMEST=FAILVL

IF (ICOMP.EQ.2) AV=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) AK5=FAILVL

IF (ICOMP.EQ.8) AK6=FAILVL

IF (ICOMP.EQ.9) AK7=FAILVL

IF (ICOMP.EQ.10) AK8=FAILVL

IF (ICOMP.EQ.11) C1=FAILVL

IF (ICOMP.EQ.12) C2=FAILVL

IF (ICOMP.EQ.13) SD1=FAILVL

IF (ICOMP.EQ.14) SD2=FAILVL

IF (ICOMP.EQ.15) SD3=FAILVL

IF (ICOMP.EQ.16) SD4=FAILVL

IF (ICOMP.EQ.17) SD5=FAILVL

IF (ICOMP.EQ.18) SD6=FAILVL

IF (ICOMP.EQ.19) SD7=FAILVL

IF (ICOMP.EQ.20) SD8=FAILVL

IF (ICOMP.EQ.21) SD9=FAILVL

GO TO 17

30 CONTINUE

IF (TM.LT.TSTOP) GO TO 100

IF (IP.EQ.1) CALL MDUMP(A,7,7)

IF (IP.EQ.2) CALL MDUMP(B,7,7)

IF (IP.EQ.3) CALL MDUMP(C,7,7)

IF (IP.EQ.4) CALL MDUMP(D,7,7)

IF (IP.EQ.5) CALL MDUMP(E,7,7)

IF (IP.EQ.6) CALL MDUMP(F,7,7)

IF (IP.EQ.7) CALL MDUMP(G,7,7)

C

FORMAT STATEMENTS

C

102 FORMAT(1=DESIGN MATRIX, DUMP (1=YES))

103 FORMAT(1=INITIAL TRACK POSITION)

104 FORMAT(1=INITIAL VELOCITY (FT/SEC))

105 FORMAT(1=WAYSINE COMMAND VELOCITY CHANGES (II))

106 FORMAT(1=ENTER CHANGE TIME AND VCMPM)!

107 FORMAT(1=SAMPLING INCREMENT (OT=.1 DEFAULT))

108 FORMAT(1=WIND GUSTS (II))

109 FORMAT(1=ENTER CHANGE TIME AND WIND (+=HEADWIND))

110 FORMAT(1=INTEGER X.Y,II DUMP (NO. OF JIS))

111 FORMAT(1=INITIAL POSITION*,FI,12.3,*)

112 FORMAT(1=INITIAL VELOCITY*,FI,12.3,*)

113 FORMAT(1=VELOCITY CHANGE TIME, NEW VCOMMAND *,I3,2F8.2)

114 FORMAT(1=WIND LEVEL CHANGE TIME, NEW VELOCITY*,I3,2F8.2)

202 FORMAT(F12.5)

203 FORMAT(1)

204 FORMAT(1=FAILURE OF DEVICE, *13.2*,*NEW VALUE =*,

205 FORMAT(1=COMPONENT FAILURE NO. (ZERO IF NO FAIL))

206 FORMAT(1=TIME OF FAILURE AND NEW COMPONENT GAIN FOR NO. *

207 FORMAT(1=A MATRIX*)

208 FORMAT(1=B MATRIX*)

209 FORMAT(1=C MATRIX*)

210 FORMAT(1=D MATRIX*)

211 FORMAT(1=FILTER OUT,10G10.2)

STOP END

TES03050

TES03060

TES03070

TES03080

TES03090

TES03100

TES03110

TES03120

TES03130

TES03140

TES03150

TES03160

TES03170

TES03180

TES03190

TES03200

TES03210

TES03220

TES03230

TES03240

TES03250

TES03260

TES03270

TES03280

TES03290

TES03300

TES03310

TES03320

TES03330

TES03340

TES03350

TES03360

TES03370

TES03380

TES03390

TES03400

TES03410

TES03420

TES03430

TES03440

TES03450

TES03460

TES03470

TES03480

TES03490

TES03500

TES03510

TES03520

TES03530

TES03540

TES03550

TES03560

TES03570

TES03580

TES03590

TES03600

TES03610

TES03620

TES03630

TES03640
SUBROUTINE FCT(N,NX,NXPRIME)
COMMON/A(7,7),U(5,1)
DIMENSION X=NX, U(1,1), X2(1,1), X3(1,1)
CALL MMULD(A*U,X,U5,5)
DO 1 I=1,7
X2(I)=X(I)
1 CONTINUE
CALL MMULD(A*U,X2,NN,NN+1)
DO 2 J=1,7
X2(J)=X2(J)+X(I)*1(I+1)
2 CONTINUE
RETURN
END
FUNCTION SGN(X)
IF (X.LT.0)SGN=-1.E0
IF (X.GE.0)SGN=1.E0
RETURN
END
SUBROUTINE FINDIF(A,N,NX,X,U,DT,AD1,AD2)
IMPLICIT R*AL(A-H,0-Z)
DIMENSION X(N,NX),U(N,NX),AD1(N,NX),AD2(N,NX)
REAL DT,TCOUNT
CALL MMULD(A,X,AD1,N,NX)
CALL MMULD(X,AD2,N,NX+1)
DO 1 J=1,N
X(J)=X(J)+AD1(J,J)*AD2(J,J)*DT
1 CONTINUE
RETURN
END
SUBROUTINE TSTRK(X,G,SLOPE,G)
TSTRK - LONGITUDINAL PROFILE OF AGRT TEST TRACK, USED TO COMPUTE SLOPE AND GRAVITY INFORMATION
INPUT - X - LONGITUDINAL POSITION (FT)
G - NOMINAL GRAVITY (FT/SEC**2)
OUTPUT - SLOPE - TEST TRACK SLOPE AT X (RAD)
G - GRAVITY COMPONENT PARALLEL TO TRACK (FT/SEC**2)
LRV DETECTION FILTER SIMULATION - USOOGT
MARCH 10, 1978 ... MICHAEL J. DYMENT
IMPLICIT R*AL(A-H,0-Z)
TRACK CONSTANTS
TRACK LENGTH = TL
XL=6000.0E0
X1=1550.0E0
X2=1450.0E0
X3=2150.0E0
X4=2350.0E0
X5=2650.0E0
X6=2950.0E0
RESET POSITION IF VEHICLE HAS LAPPED TRACK
IF(X.GT.XL)X=X-XL
C VERTICAL CURVE 1

IF(X.LT.X1) GO TO 6
IF(X.GE.X2) GO TO 2
XT=X-X1
DY=0.0002E0*XT
GO TO 1
CONTINUE

C VERTICAL CURVE 2

IF(X.GE.X3) GO TO 5
XT=X-X3
DY=-0.0002E0*XT
GO TO 1
CONTINUE

C PLATEAU

IF(X.GE.X4) GO TO 4
DY=0.0E0
GO TO 1
CONTINUE

C VERTICAL CURVE 3

IF(X.GE.X5) GO TO 7
XT=X-X4
DY=-0.0002E0*XT
GO TO 1
CONTINUE

C VERTICAL CURVE 4

IF(X.GE.X6) GO TO 6
XT=X-X6
DY=0.0002E0*XT
GO TO 1
CONTINUE

C HORIZONTAL TRACK

DY=0.0E0
CONTINUE

C COMPUTE GRAVITY COMPONENT

G=G*N*DY
SLOPE=DY
RETURN
END

SUBROUTINE MULT(A,H,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)=C(I,J)*A(I,J)*B(J,K)
1 CONTINUE
RETURN
END
SUBROUTINE SUBROUTINE (A,R,C,M,N)
REAL A(M,N),R(M,N),C(M,N)
DO 1 I=1,M
DO 1 J=1,N
C(I,J)=A(I,J)+R(I,J)
1 CONTINUE
RETURN
END

SUBROUTINE SUBROUTINE (A,R,C,M,N)
REAL A(M,N),R(M,N),C(M,N)
DO 1 I=1,M
DO 1 J=1,N
C(I,J)=A(I,J)-R(I,J)
1 CONTINUE
RETURN
END

SUBROUTINE PROFL (VCOMM,AMAX,JMAX,DT,AA,VC,AX,IMD)
PROFILE:
CREATES A PROFILED VELOCITY COMMAND SUBJECT TO MAXIMUM
ACCELERATION AND JERK CRITERIA

INPUT
VCOMM - EXTERNAL COMMANDED VELOCITY (FT/SEC)
AMAX - MAXIMUM ACCELERATION CRITERIA (FT/SEC**2)
JMAX - JERK CRITERIA (FT/SEC**3)
DT - INTEGRATION INTERVAL
IMD - FLAG
0 - NEW VELOCITY COMMAND - SELECT NEW PROF
1 - RETAIN PRESENT PROF

OUTPUT
AA - COMMANDED ACCELERATION
VC - COMMANDER VELOCITY
AX - COMMANDER POSITION

IMPLICIT REAL*4(A-H,O-Z)
REAL*8 DT
REAL JMAX, JMAX

TEST FOR MODE SELECT
DV=VCOMM-VC
IF(IFL.FL.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(AA).GE.1.E-6)GO TO 10
IMD=2
10 CONTINUE
C MODE SELECT LOCATION
C
IF(1.ME.,Fl.)GO TO 100
IF(I.ME.,Eq.)GO TO 200
IF(I.ME.,Eq.)GO TO 300
IF(I.ME.,Eq.,4.)GO TO 400
C
C MODE 1 - STANDARD VELOCITY PROFILE
C
100 CONTINUE
IF(IFL.NE.1)GO TO 110
TCOUNT=0.00000
AA=0.00
T1=AMAX/JMAX
T2=(ABS(DV)-T1*AMAX)/AMAX
IT1=T1/DT+1
IT2=T2/DT+1
T2=FLOAT(IT2)*DT
JMX=ABS(DV)/(T1**2+T1*T2)
AMX=JMX*IT1
JMA=JMAX*SGN(DV)
AMX=AMX*SGN(DV)
T=T2*T1
T=FLOAT(IT2)*DT
IXX=ABS(DV)
/ (T1**2+T1*T2)
AX=JMX*T1
AMX=JMX*SGN(DV)
AMX=AMX*SGN(DV)
T=T2*T1
T=FLOAT(IT2)*DT
IFL=1
110 CONTINUE
TCOUNT=TCOUNT+DT
IF(TCOUNT.EQ.0.00000)AA=AA+JMX*DT
IF(TCOUNT.EQ.T1.0 AND .NOT.TCOUNT.T2)AA=AMX
IF(TCOUNT.EQ.T1.0 AND TCOUNT.LT.T2)AA=AA-JMX*DT
IF(TCOUNT.EQ.T2.00)AA=0.00
IF(ABS(AA).GT.AM)AA=AMX
VC=HC+AA*DT
AX=AA+VC*DT
RETURN
C
C MODE 2 - MODIFIED VELOCITY PROFILE
C
200 CONTINUE
IF(IFL.NE.1)GO TO 210
TCOUNT=0.00000
DV=ABS(DV)/2.00
AMAX=SQRT(2.00*DV*JMAX)
AMAX=JMX/JMAX
IT=IT1/DT+1
T1=FLOAT(IT1)*DT
AMAX=2.00*DV/T1
JMAX=AMAX*T1
AMX=ABS(AA+AMAX)*SGN(DV)
JMX=ABS(JM)+SGN(DV)
T2=2.00*T1
AA=0.00
IFL=1
210 CONTINUE
TCOUNT=TCOUNT+DT
IF(TCOUNT.EQ.0.00 AND .NOT.TCOUNT.T1)AA=AA+JMX*DT
IF(TCOUNT.EQ.5.T1 AND .NOT.TCOUNT.T2)AA=AA-JMX*DT
IF(TCOUNT.EQ.5.T1 AND .NOT.TCOUNT.T2)AA=0.00
VC=HC+AA*DT
AX=AA+VC*DT
RETURN
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.DO
T1=ABS(AA/JMAX)
IT1=T1/DT+1
T1=FLOAT(IT1)*DT
JMX=AA/T1
JMAX=SGN(AA)*ABS(JMX)
IFL=1
310 CONTINUE
TCOUNT=TCOUNT+DT
IF (TCOUNT.GT.T1) GO TO 320
AA=AA+JMAX*DT
VC=VC+AA*DT
AX=AA+VC*DT
RETURN
320 CONTINUE
AA=0.DO
AX=AA+VC*DT
TCOUNT=TCOUNT+DT
IFL=0
RETURN
330 CONTINUE
AA=DV/DT
A1T=AA-AA
A1=AJT/JT
IF(A1=AS(AJT)*ST,JMAX)AA=AA+JMAX*SGN(AJT)*DT
IF(A1=AS(AA)*ST,JMAX)AA=JMAX*SGN(AA)
VC=VC+AA*DT
AX=AA+VC*JT
RETURN
C MODE 4 - UNSPECIFIED
C
400 CONTINUE
RETURN
END
SUBROUTINE MOUTP(A,N,M)
DIMENSION A(M,N)
WHITE(*,100)
100 FORMAT(''),
DO 1 I=1,M
PRINT 10,1,(A(I,K)*K=1,N)
1 CONTINUE
10 FORMAT(12,1X,10(E9.3,1X))
RETURN
END
DO 1 I=1,N
1 XMFES(I)=XMV(I)
DO 2 I=1,l'
2 EPS(I)=0.
C2 EPS(I)=2.
1 CONTINUE
IFLAG=IFLAG+1
NW=WN+1
U(I)=UU(I)
U(2)=U(2)
V(I)=UU(I)
C COLDWALL FRICTION COMPONENT
C
V(2)=1.
V(2)=0.
V(3)=EPS(I)
V(4)=EPS(2)
IND=1
CALL FINDUF(ADFT,6,4,XMES,VTINT,AD1,AD2)
C CALL DVERK(4,ADFTC,YMES,TEND,IND,C,4,4,IFTR)
C** COMPUTATION OF YMES,EPS
DO 4 I=1,N
4 BUF(I)=0.
DO 4 J=1,N
4 BUF(I)=BUF(I)+COET(I,J)*XMES(J)
DO 5 I=1,N
5 YMES(I)=0.
DO 5 J=1,N
5 YMES(I)=YMES(I)+EDFT(I,J)*U(J)
DO 6 I=1,N
6 EPS(I)=EPS(I)-YMES(I)
DO 7 I=1,N
7 EPS(I)=EPS(I)-EPS(I)
C DO 8 I=1,N
C
CA EPS(I)=EPS(I)
EPS(I)=EPS(I))
C EPS(I)=EPS(I)+EPS(I)
EPS(I)=EPS(I)
RFUKN
END
SUBROUTINE DETFCN.T,YMES,XMES)
COMMON/DET/ADFT,COFT,VTINT,AD1,AD2,NMES,EPS
DIMENSION ADFT(6,6),XMES(6),YMES(6),EPS(6)
DO 1 I=1,N
1 RUF(I)=0.
DO 1 J=1,N
1 AUF1(I)=AUF1(I)*ADET(I,J)*XESP(J)
DO 2 I=1,N
AUF2(I)=0.
DO 2 J=1,N
2 AUF2(I)=AUF2(I)*ADET(I,J)*U(J)
DO 3 I=1,N
XESP(I)=0.
DO 3 J=1,N
3 XESP(I)=XESP(I)*ADET(I,J)*EPS(J)
DO 4 I=1,N
4 XESP(I)=XESP(I)+AUF1(I)+AUF2(I)
RETURN
END

SUBROUTINE MODEL(AMV,AMVEST,A3,A4,A5,A6,A7,A8)
C COMPUTES ADET,CDET,ODET,EDET,ET,AD
C COMMON ADET,ODET,CDET,EDET,ET,AMV,AMVEST,A3,A4,A5,A6,A7,A8
DIMENSION ADET(6,6),ODET(6,2),EDET(2,2),ET
DIMENSION EM(5,4),EPS(2)
DO=2
TO=2
IS=2
J=6
DO 1 I=1,N
ADET(I,J)=A1
ADET(I,2)=1./AMV
ADET(2,3)=1.
ADET(3,4)=1.
ADET(4,5)=A6
ADET(5,1)=A3
ADET(5,2)=A5*A6/AMV
ADET(5,6)=A2
DO 2 I=1,N
ADET(I,4)=A4*AMVEST
ADET(4,1)=A1
ADET(6,1)=1.
C COULOMS FRICTION PARAMETERS
C COMMON 9DET(1,2)=-100./AMV
9DET(5,2)=A1*A4*A5*100./AMV
DO 3 I=1,N
3 J=1,N
CDET(1,5)=1.
CDET(2,1)=A4
DO 4 I=1,N
4 J=1,N
CDET(1,1)=AMVEST
DO 5 I=1,N
5 J=1,N
5 END
FIL00610
FIL00620
FIL00630
FIL00640
FIL00650
FIL00660
FIL00670
FIL00680
FIL00690
FIL00700
FIL00710
FIL00720
FIL00730
FIL00740
FIL00750
FIL00760
FIL00770
FIL00780
FIL00790
FIL00800
FIL00810
FIL00820
FIL00830
FIL00840
FIL00850
FIL00860
FIL00870
FIL00880
FIL00890
FIL00900
FIL00910
FIL00920
FIL00930
FIL00940
FIL00950
FIL00960
FIL00970
FIL00980
FIL00990
FIL01000
FIL01010
FIL01020
FIL01030
FIL01040
FIL01050
FIL01060
FIL01070
FIL01080
FIL01090
FIL01100
FIL01110
FIL01120
FIL01130
FIL01140
FIL01150
FIL01160
FIL01170
FIL01180
FIL01190
FIL01200
FIL01210
C

SUBROUTINE SOLUT(N)

C READS DDET

C

COMMON/DDET,ADET,DET,ADET,EDET,EDET,EDET,IP,IS,U,EPS,BP
DIMENSION ADET(6,6),DET(6,2),CDET(2,6),ODET(6,2)
DIMENSION FDET(2,2),AP(6,4),U(1),EPS(2)

ID=2

WRITE(6,100)

100 FORMAT(1X,*TYPE ADET,DETECTION FILTER*)

FORMAT(1X,*ODET(1,J),J=1,N)*1,*IP,IP=1,N)

WRITE(6,101) (ADET(I,J),J=1,N)*1,I=1,N)

WRITE(6,102) (EDET(I,J),J=1,N)*1,I=1,N)

WRITE(6,103) (FDET(I,J),J=1,N)*1,I=1,N)

RETURN

END
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

\[
\begin{align*}
\dot{V}_{ind} &= \frac{K_4 T}{m} \\
(T) &= T \\
(\dot{T}) &= \dot{T} \\
(\ddot{T}) &= a_4 T + a_3 \dot{T} + a_2 \ddot{T} + K_3 T_c \\
T_e &= K_1 (V_c - V_{ind}) + K_2 (V_c - V_{ind}) \\
&= K_1 a_c - K_1 K_4 \frac{T}{m} + K_2 V_c - K_2 V_{ind} \\
\dot{V}_c &= a_c
\end{align*}
\]
Furthermore \( T_c = T_e + m_v a_c \)
\[
V_{\text{ind}} = V_{\text{ind}}
\]

In matrix form

\[
\begin{pmatrix}
V_{\text{ind}} \\
\dot{T} \\
\ddot{T} \\
\frac{\dot{T}}{T} \\
\frac{\ddot{T}}{T} \\
\frac{\dot{V}}{V_e} \\
\frac{\ddot{V}}{V_e}
\end{pmatrix} =
\begin{pmatrix}
0 & \frac{\kappa_4}{m} & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & a_4 & a_3 & a_2 & K_3 & 0 & 0 \\
-k_2 & -k_1 & k_4 & 0 & 0 & 0 & k_2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
V_{\text{ind}} \\
\dot{T} \\
\ddot{T} \\
\frac{\dot{T}}{T} \\
\frac{\ddot{T}}{T} \\
\frac{\dot{V}}{V_e} \\
\frac{\ddot{V}}{V_e}
\end{pmatrix} + 
\begin{pmatrix}
0 \\
0 \\
0 \\
K_3 mV \\
V_e \\
V_e
\end{pmatrix} a_c
\]

\[
\begin{pmatrix}
T_e \\
V_{\text{ind}}
\end{pmatrix} =
\begin{pmatrix}
0 & 0 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
V_{\text{ind}} \\
\dot{T} \\
\ddot{T} \\
\frac{\dot{T}}{T} \\
\frac{\ddot{T}}{T} \\
\frac{\dot{V}}{V_e} \\
\frac{\ddot{V}}{V_e}
\end{pmatrix} + 
\begin{pmatrix}
0 & 0 \\
0 & 0 \\
0 & 0 \\
0 & 0 \\
0 & 0 \\
0 & 0 \\
0 & 0
\end{pmatrix} m_v a_c
\]

Event associated with \( K_4 \):
\[
\begin{pmatrix}
\frac{1}{m} \\
0 \\
0 \\
-k_1/m \\
0
\end{pmatrix} = b_4
\]
\[
C b_4 = \begin{pmatrix}
-k_4/m \\
1/m
\end{pmatrix}
\]
Event associated with $K_1$: 
\[
\begin{pmatrix}
0 \\
0 \\
0 \\
1 \\
0
\end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\
0
\end{pmatrix} \Rightarrow C_{b_1} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}
\]

Event associated with $K_2$: 
\[
\begin{pmatrix}
0 \\
0 \\
0 \\
1 \\
0
\end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\
0
\end{pmatrix} \Rightarrow C_{b_2} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}
\]

($b_1 = b_2$)

Event associated with $K_3$: 
\[
\begin{pmatrix}
0 \\
0 \\
0 \\
1 \\
0
\end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\
0
\end{pmatrix}
\]

But $C_{b_3} = 0$. We compute $A_{b_3}, CA_{b_3} = 0$, we compute $A^2_{b_3}, CA^2_{b_3} = 0$.

We compute
\[
A^3_{b_3} = \begin{pmatrix}
\frac{k_4}{m} \\
\frac{a_2}{m} + \frac{a_2}{m} \\
\frac{a_4 + 2a_3}{m} + \frac{a_3}{m} \\
- \frac{k_4}{m} \\
0
\end{pmatrix}
\]

\[
CA^3_{b_3} = \begin{pmatrix} -\frac{k_4 k_4}{m} \\
\frac{k_4}{m}
\end{pmatrix}
\]

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 $b_3 \in \mathcal{R}_4$, $b_3$ and $b_4$ 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

$$T_c = T_e + m_v a_c$$

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

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

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

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

In matrix form

$$\begin{pmatrix}
\dot{X}_v \\
\dot{T}
\end{pmatrix}' =
\begin{pmatrix}
0 & 31 \\
0 & a_4 & a_3 & a_2 & K_3 & 0 \\
0 & 0 & 0 & 0 & K_2 & V_c \\
0 & \frac{K_4}{m} & 0 & 0 & 0 & K_1
\end{pmatrix}
\begin{pmatrix}
\dot{X}_v \\
\dot{T} \\
\dot{V}_c
\end{pmatrix}
+ \begin{pmatrix}
0 \\
0 \\
T_e \\
V_c
\end{pmatrix}
$$

$$\begin{pmatrix}
\dot{T}_c \\
V_{inc}
\end{pmatrix} =
\begin{pmatrix}
0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
K_4 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
\dot{X}_v \\
\dot{T}_c \\
\dot{V}_{inc}
\end{pmatrix}
+ \begin{pmatrix}
0 \\
0 \\
T_e \\
V_c
\end{pmatrix}
$$
Event associated with $K_1$: 

\[
\begin{bmatrix}
0 \\
0 \\
1 \\
0
\end{bmatrix} = b_1 \\
\begin{bmatrix}
1
\end{bmatrix} \quad \therefore b_1 = 1
\]

Event associated with $K_2$: 

\[
\begin{bmatrix}
0 \\
0 \\
0 \\
1
\end{bmatrix} = b_2 = b_1
\]

Event associated with $K_3$: 

\[
\begin{bmatrix}
0 \\
0 \\
0 \\
1
\end{bmatrix} = b_3
\]

$C_{b_3} = 0$ we compute 

$A_{b_3}$, $CA_{b_3} = 0$, $A^{2}_{b_3}$, $CA^{2}_{b_3} = 0$

$A^{3}_{b_3} = 
\begin{bmatrix}
1/m \\
a_2 \\
a_3 + a_2 \\
a_4 + 2a_3 + a_2 \\
-\kappa_1\kappa_4/m \\
0
\end{bmatrix} \\
\begin{bmatrix}
-\kappa_1\kappa_4/m \\
\kappa_4/m
\end{bmatrix}
$

Event associated with $K_4$: 

$\Delta C = \begin{bmatrix}
0 \\
1
\end{bmatrix}$ \\
$\begin{bmatrix}
0 \\
0 \\
1
\end{bmatrix} = b_4 = b_1 = \begin{bmatrix}
0 \\
0 \\
1
\end{bmatrix}$
As the vector $e_{61} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$ is such that $C e_{61} \parallel \begin{pmatrix} 0 \\ -1 \end{pmatrix}$ and $A e_{61} \parallel b_1$. The events associated with a failure in $K_4$ are $e_{61}$ and $b_1$. As $C e_{61}$ and $C b_1$ 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 $b_1$ and $A^3 b_3$. Its numerical value was

$$(A.5) \quad D = \begin{pmatrix} 15.5 \\ -231.75 \\ 556.951 \\ 12350 \\ -231.628 \\ 10 \\ 52.4335 \\ 0.1 \\ 149.98 \end{pmatrix}, \quad \zeta = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Running the simulation with this filter, it was discovered that a failure in $K_4$ generated a unidirectional output along $\xi_2$. 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 $b_1$ and $A^3 b_3$ 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 \( V_{\text{ind}}, T, \dot{T}, \ddot{T}, T_e, V_c \) or the states \( \dot{V}_v, T, \dot{T}, \ddot{T}, T_e, V_c \) are selected. It does then make sense that a failure in \( K_4 \) generates unidirectional output along \( \dot{\xi}_2 \) 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:

\[
\begin{align*}
\text{If } K_5 K_4' = 1, \text{ nothing is changed in the real system.}
\end{align*}
\]

This could be done physically by multiplying the real output \( V_{\text{ind}} \) 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
\[ T_c = T_e + m v a_c \]

\[ T_e = k_1 a_c - k_1 k_5 k_3 \frac{T}{m} + k_2 V_c - k_2 k_5 k_3 \dot{X}_v \]

\[ (X_v)' = \frac{T}{m} \]

\[ (T)' = T \]

\[ (T)' = k_3 T_e + k_3 m v a_c + a_2 \ddot{T} + a_3 \dot{T} + a_4 T \]

In matrix form

\[
\begin{pmatrix}
\dot{X}_v \\
\dot{T} \\
\ddot{T} \\
\dot{V}_c \\
\dddot{V}_c
\end{pmatrix} =
\begin{pmatrix}
0 & \frac{1}{m} & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 \\
-\frac{k_5 k_3}{m} & -\frac{k_5 k_3}{m} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
\dot{X}_v \\
\dot{T} \\
\ddot{T} \\
\dot{V}_c \\
\dddot{V}_c
\end{pmatrix} +
\begin{pmatrix}
k_3 \frac{T}{m} \\
k_3 m v \\
k_1 \\
0 \\
0
\end{pmatrix}
\]
Event associated with $K_1$: 
\[
\begin{pmatrix}
0 \\
0 \\
0 \\
1 \\
0
\end{pmatrix} \quad \begin{pmatrix}
0 \\
0 \\
0 \\
1 \\
0
\end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}
\]

Event associated with $K_2$: 
\[
\begin{pmatrix}
0 \\
0 \\
1 \\
0
\end{pmatrix}
\]

Event associated with $K_3$: 
\[
\begin{pmatrix}
0 \\
0 \\
1 \\
0
\end{pmatrix} \quad \begin{pmatrix}
0 \\
0 \\
1 \\
0 \\
0
\end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}
\]

$A^3 b_3$ is such that $CA^3 b_3 = 0$, as $CA^2 b_3 = 0$, the event associated with $K_3$ will be $A^3 b_3$

\[
A^3 b_3 = \begin{pmatrix}
1/m \\
a_2 \\
a_3 + a_2 \\
1/m
\end{pmatrix} \quad \begin{pmatrix}
-1/\gamma \gamma/\gamma \gamma/\gamma \gamma/\gamma \\
\gamma/\gamma \gamma/\gamma \gamma/\gamma \gamma/\gamma \\
\gamma/\gamma \gamma/\gamma \gamma/\gamma \gamma/\gamma \\
\gamma/\gamma \gamma/\gamma \gamma/\gamma \gamma/\gamma
\end{pmatrix}
\]

Events associated with $K_4'$:

a – variation in $A$ along $b_1$ =

\[
\begin{pmatrix}
0 \\
0 \\
0 \\
1 \\
0
\end{pmatrix}
\]
b\text{-} variation in C along \[
\begin{pmatrix}
0 \\
1
\end{pmatrix}
\]
A_5 \quad \mathbf{F} = \begin{pmatrix}
1 \\
0 \\
0 \\
0
\end{pmatrix}
is such that \[
\mathbf{c}_2 \mathbf{F} = \begin{pmatrix}
0 \\
1
\end{pmatrix}
\quad \text{and} \quad A \mathbf{F} \parallel b_1.
\]

The events associated with $K_4'$ are $b_1$ and $\mathbf{F}$. 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 $K_5 = 2$, $K_4' = .5$.

\[
(A-7) \quad \mathbf{D} = \begin{pmatrix}
0 & -3.1 \\
0 & -4662.75 \\
0 & 11135.904 \\
12150 & -463225657.6 \\
20 & 104486.687 \\
0.1 & 299.96
\end{pmatrix}
\quad \mathbf{c} = \begin{pmatrix}
0 & 0 & 0 & 0 & 1 & 0
\end{pmatrix}
\]

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 $\mathbf{\xi}$, whose differential equation is $\dot{\mathbf{\xi}} = (A_DC) \mathbf{\xi} + b_1 \mathbf{n}_1(t)$ 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 $\mathbf{\xi}_1$ and $\mathbf{\xi}_2$. 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 $K_4 V$. 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 pre-
scribe 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 $\xi_1$ and the
error due to propulsion failures to $\xi_2$. 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 $\xi_2$. 
ORTHOGONAL REDUCTION PROCEDURE

Orthogonal reduction is a procedure which determines the null space of a matrix \( V \); i.e., all independent solutions of \( Vw = 0 \). Suppose \( V \) is \( n \times n \)
\[
V = \begin{bmatrix}
V_1^T \\
\vdots \\
V_n^T
\end{bmatrix}
\]
The orthogonal reduction procedure is an iterative process which generates an \( n \times n \) 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 \( n \times n \) matrix \( J_1^{(1)} \). An auxiliary \( n \)-vector is defined by \( w_1 = J_1^{(1)} v_1 \). If \( v_1 \) is nonzero \( w_1 \) will be nonzero, since \( J_1^{(1)} \) is positive definite. Furthermore \( w_1^T v_1 \) will be nonzero. A new symmetric positive semi-definite matrix is defined by
\[
J_2^{(2)} = J_1^{(1)} - \frac{w_1 w_1^T}{w_1^T v_1}
\]
This matrix has the property that \( J_2^{(2)} v_1 = 0 \).

The procedure continues according to the following general iteration

(1) with \( J_1^{(i)} \) from the previous iteration, form the auxiliary vector \( w_1 = J_1^{(i)} v_1 \)
(2) if \( w_i \neq 0 \) set \( J_2(i+1) = J_2(i) - \frac{w_i w_i^T}{w_i^T v_i} \)

if \( w_i = 0 \) set \( J_2(i+1) = J_2(i) \) and return to (1).

The algorithm has the following important properties:

(1) If \( J_2(i) \) is positive semidefinite, \( w_i^T v_i = 0 \) if and only if \( w_i = 0 \). This follows from the definition of \( w_i \).

(2) If \( J_2(i) \) is positive semidefinite so is \( J_2(i+1) \). This is obviously true if \( w_i = 0 \). Assume \( w_i \neq 0 \). For any arbitrary \( n \)-vector \( z \) and any scalar \( \lambda' \)

\[
(z - \lambda' v_i)^T J_2(i) (z - \lambda' v_i) \geq 0 \quad (A-8)
\]

In particular this must be true for

\[
\lambda' = \frac{w_i^T z}{w_i^T v_i}
\]

Substituting this value of \( \lambda' \) in (A8) and expanding it, we get

\[
(z - \lambda' v_i)^T J_2(i) (z - \lambda' v_i) = z^T J_2(i) z - 2 \lambda' v_i^T J_2(i) z + \lambda'^2 v_i^T J_2(i) v_i
\]

\[
= z^T J_2(i) z - 2 \lambda' w_i^T z + \lambda'^2 w_i^T v_i
\]

\[
= z^T J_2(i) z - \left( \frac{w_i^T z}{w_i^T v_i} \right)^2 w_i^T v_i
\]

\[
= z^T J_2(i+1) z \geq 0 \quad (A-9)
\]

By induction, this shows that all \( J_2(i) \) are positive semidefinite if the starting matrix \( J_2(1) \) is at least positive semidefinite.

(3) If \( w_i \neq 0 \) then \( \text{rk} J_2(i+1) = \text{rk} J_2(i) - 1 \)

and the null space of \( J_2(i+1) \) is the subspace formed by \( v_i \) and the
null space of $\mathcal{J}^{(i)}$.

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

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

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

In step 5c of the detection filter design algorithm, a matrix $\mathcal{J}_i$ was found where columns span the space $\mathbf{R}_i$. In step 5d the orthogonal reduction procedure is applied to

$$\mathcal{M} = \begin{bmatrix} \mathbf{C} \\ \mathbf{C} \mathbf{A} \\ \vdots \\ \mathbf{C} \mathbf{A}^{n-1} \end{bmatrix}$$

starting with $\mathcal{J}_i$ (for $i = 1, \ldots, r$)

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

\[ w_i = \mathcal{R}^{(i)} (c_j A \overline{V}_1)^T + 0 \]

By construction \( w_i \) lies in the null space of \( M \) and satisfies

\[ \begin{bmatrix} c_A^T \\ c_A \overline{V}_1 \\ c_A \overline{V}_{1,2} \end{bmatrix} w_i = 0 \quad \text{and} \quad c_A \overline{V}_1^{-1} w_i \neq 0 \]

These are all the requirements for a detection generator, except for the magnitude. \( w_i \) is a multiple of the detection generator \( q_i \).

If \( \text{rk} (\overline{R}_i) = 1 \), \( \text{rk}(R_i) = 0 \) as \( \overline{R}_i = b_i \oplus R_i \) (if \( Cb_i \neq 0 \), otherwise, use \( Ab_i \) etc). Then \( \mathcal{J}_2 = 0 \), 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 \( b_i \).

**Intermediate turning points**

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

\[ M_D = \begin{bmatrix} c_S' \\ c_S' (A - \Theta_S c) \\ \vdots \\ c_S' (A - \Theta_S c)^{n-1} \end{bmatrix} \]

starting with a positive definite matrix (on option). The rows of \( M_D \) correspond to the \( v_i^T \) 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 $w_1 = \mathcal{S}_L(i) (C_j (A-D_S)_{\ell})^T = 0$, where $C_j$ is the $j$th row of $C_S'$, it is then known that $C_j (A-D_S C)$ 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.)