To the Graduate Council:

I am submitting herewith a thesis written by Frederik P. Wolvaardt entitled "An Integrated Approach for Sensor Failure Detection and Signal Reconstruction." I have examined the final copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Nuclear Engineering.

Belle R Upadhyaya

Belle R. Upadhyaya, Major Professor

We have read this thesis
and recommend its acceptance:

Jack F. Wasserman

B-7. Barqua

Accepted for the Council:

Vice Provost
and Dean of The Graduate School

STATEMENT OF PERMISSION TO USE

In presenting this thesis in partial fulfillment of the requirements for a Master's degree at The University of Tennessee, Knoxville, I agree that the Library shall make it available to borrowers under rules of the Library. Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgment of the source is made.

Permission of extensive quotation from or reproduction of this thesis may be granted by my major professor, or in his absence, by the Head of Interlibrary Services when, in the opinion of either, the proposed use of the material is for scholarly purposes. Any copying or use of the material in this thesis for financial gain shall not be allowed without my written permission.

Signature F. P. Wolvacult

Date 26 November 1986

# AN INTEGRATED APPROACH FOR SENSOR FAILURE DETECTION AND SIGNAL RECONSTRUCTION

A Thesis

Presented for the

Master of Science

Degree

The University of Tennessee, Knoxville

Frederik P. Wolvaardt

December 1986

This work is dedicated to my parents George and Alta Wolvaardt.

## ACKNOWLEDGMENTS

The author wishes to thank Dr. B. R. Upadhyaya for his continuous guidance and suggestions during the entire course of this project. He wishes to thank Dr. P. F. Pasqua for his support and advice throughout the five years spent at the University of Tennessee. The author is grateful to Dr. T. W. Kerlin for his suggestions and Dr. O. Glöckler for his help in recording the data.

The author extends his acknowledgment to the Measurement and Control Engineering Center at the University of Tennessee for the financial support through research funding. The author also wishes to thank Mrs. Barbara Kelmers and Mrs. Marilyn McCullough for their diligence in typing the manuscript.

## ABSTRACT

The objective of this study is the development of a signal validation system utilizing redundant arrays of sensors. The signal validation system must be able to detect, isolate, and identify sensor degradations as well as provide a reconstruction of the measurement.

This was accomplished by combining the Generalized Consistency Check and the modified Sequential Probability Ratio Test in a decision estimator module. The Generalized Consistency Check is a computationally efficient management system for redundant sensors, while the Sequential Probability Ratio Test provides the ability to make decisions based on the degradation history of a sensor. The methodology was also extended to the detection of noise degradations by separating the high and the low frequency components of signals.

To verify the signal validation system an experimental air flow loop was constructed. Signals containing bias and noise type degradations were recorded.

The signal validation system was able to detect and isolate bias and noise type failures under steady state and transient conditions. An estimate of the measured variable was successfully provided for all the failure situations considered.

TABLE OF CONTENTS


| CHAPTER | | PAGE |
|---|---|---|
| 1. | INTRODUCTION | 1 |
| 1.1 | Problem Definition | 1 |
| 1.2 | Development of Methodology | 3 |
| 1.3 | Application and Verification | 5 |
| 1.4 | Outline of the Thesis | 6 |
| 2. | GENERAL APPROACH | 8 |
| 2.1 | Introduction | 8 |
| 2.2 | Case Studies | 8 |
| 2.3 | General Approach | 11 |
| 2.4 | Contributions of This Thesis | 16 |
| 2.5 | Summary | 17 |
| 3. | METHODOLOGY | 18 |
| 3.1 | Introduction | 18 |
| 3.2 | Generalized Consistency Check | 19 |
| 3.3 | Sequential Probability Ratio Test | 25 |
| 3.4 | The Modified Sequential Probability Ratio Test | 36 |
| 3.5 | Detection of Bias Failures | 43 |
| 3.6 | Detection of Noise Failures | 45 |
| 3.7 | Summary | 49 |
| 4. | APPLICATION OF THE METHODOLOGY | 51 |
| 4.1 | Introduction | 51 |
| 4.2 | Software Development | 52 |
| 4.3 | Hardware Development | 55 |
| 4.4 | Experimental Result | 58 |
| 4.4.1 | Steady State Bias Failure | 58 |
| 4.4.2 | Bias Degradation During a Transient | 67 |
| 4.4.3 | Noise Degradation | 70 |
| 4.4.4 | Simultaneous Bias and Noise Degradation | 77 |
| 4.4.5 | Repeated Pulse Type Failure | 84 |
| 4.5 | Redundancy from Independent Sensors | 89 |
| 4.6 | Summary | 97 |
| 5. | CONCLUDING REMARKS | 98 |
| 5.1 | Conclusions | 98 |
| 5.2 | Recommendations | 99 |
| LIST OF REFERENCES | | 101 |

CHAPTER PAGE

APPENDIXES .....105

APPENDIX A. DECISION ESTIMATOR MODULE .....106
APPENDIX B. LISTING FOR "DATAGET.FOR" .....140
APPENDIX C. LISTING FOR "GDATRAN.FOR" .....143
APPENDIX D. LISTING FOR "PLOT 1.FOR" .....147

VITA .....151

LIST OF FIGURES


| FIGURE | | PAGE |
|---|---|---|
| 2.2.1 | Illustration of a Decision Estimation Module with Three Input Signals | 15 |
| 2.2.2 | Example of a Signal Validation Flow Diagram | 15 |
| 3.2.1 | The Uniform Sensor Noise Probability Density Function | 21 |
| 3.3.1 | The Gaussian Density Functions Representing Normal and Degraded Sensor Ouputs | 29 |
| 3.3.2 | The Trajectory of the Likelihood Ratio Function, $\lambda$ | 33 |
| 3.3.3 | The Definition of False and Missed Alarm Probabilities | 35 |
| 3.4.1 | Illustration of the Time Delay, t, in the Wald System | 37 |
| 3.4.2 | Illustration of the Modified SPRT | 39 |
| 3.4.3 | The Relationship Between Chien's Mean Time Between False Alarms and Wald's Missed and False Alarm Probabilities | 42 |
| 3.5.1 | Flow Chart Illustrating the Implementation of the Signal Validation Methodology | 46 |
| 4.3.1 | Schematic Diagram of the Air Flow Loop Illustrating the Sensor Locations | 56 |
| 4.4.1 | Bias Degradation in Second Thermocouple at Steady State Conditions | 60 |
| 4.4.2 | The LLR for the Second Thermocouple (Bl = 9.9) | 60 |
| 4.4.3 | The Weighted Average of the Three Thermocouples | 63 |
| 4.4.4 | The LLR for the First Thermocouple (Bl = 9.9) | 63 |
| 4.4.5 | The LLR for the Third Thermocouple (Bl = 9.9) | 64 |
| 4.4.6 | Bias Degradation in Second Thermocouple at Transient Conditions | 67 |

| FIGURE | | PAGE |
|---|---|---|
| 4.4.7 | The LLR for the Second Thermocouple (Bl = 9.9) | 67 |
| 4.4.8 | The Weighted Average of the Three Thermocouples | 68 |
| 4.4.9 | The LLR for the First Thermocouple (Bl = 9.9) | 68 |
| 4.4.10 | The LLR for the Third Thermocouple (Bl = 9.9) | 69 |
| 4.4.11 | The LLR for the Second Thermocouple (Bl = 9.9) | 71 |
| 4.4.12 | The LLR for the First Thermocouple (Bl = 9.9) | 71 |
| 4.4.13 | The LLR for the Third Thermocouple (Bl = 9.9) | 72 |
| 4.4.14 | The Weighted Average of the Three Thermocouples | 72 |
| 4.4.15 | Periodic Noise Degradation in the Second Thermocouple | 74 |
| 4.4.16 | The Weighted Average of the Three Thermocouples | 74 |
| 4.4.17 | Bias LLR for the Second Thermocouple (Bl = 11.3) | 75 |
| 4.4.18 | Bias LLR for the First Thermocouple (Bl = 11.3) | 75 |
| 4.4.19 | Bias LLR for the Third Thermocouple (Bl = 11.3) | 76 |
| 4.4.20 | Variance LLR for all Three Thermocouples (A = -6.9, B = 6.9) | 76 |
| 4.4.21 | Bias and Noise Degradations in Second Signal | 79 |
| 4.4.22 | Normal Noise Components of Signals | 79 |
| 4.4.23 | Degraded Noise Component in Signal 2 | 80 |
| 4.4.24 | Bias LLR for the Second Signal (Bl = 11.3) | 80 |
| 4.4.25 | Bias LLR for the First Signal (Bl = 11.3) | 82 |

| FIGURE | | PAGE |
|---|---|---|
| 4.4.26 | Bias LLR for the Third Signal (Bl = 11.3) | 82 |
| 4.4.27 | Noise LLRs for all Three Signals (A = -6.91, B = 6.91) | 83 |
| 4.4.28 | Comparison of Normal Signal 1, Degraded Signal 2, and Estimate | 83 |
| 4.4.29 | The First Thermocouple Signal Containing Pulses | 86 |
| 4.4.30 | The Bias LLR for the First Thermocouple (Bl = 12.1) | 86 |
| 4.4.31 | The Bias LLR for the Second Thermocouple (Bl = 12.1) | 87 |
| 4.4.32 | The Bias LLR for the Third Thermocouple (Bl = 12.1) | 87 |
| 4.4.33 | The Weighted Average of the Three Thermocouples | 88 |
| 4.4.34 | The Variance LLR for the First Thermocouple (A = -6.91, B = 6.91) | 88 |
| 4.4.35 | The Variance LLRs for the Second and Third Thermocouples (A = -6.91, B = 6.91) | 90 |
| 4.5.1 | Amplitude Probability Density Function for Outlet Thermocouple Noise Confirming the Gaussian Nature of the Signal (Variance = 1.38 x $10^{-7}mV^2$) | 93 |
| 4.5.2 | Cross Correlation Function of the First and Seventh Thermocouples (case 1) | 94 |
| 4.5.3 | Cross Correlation Function of the First and Seventh Thermocouples (case 2) | 94 |
| 4.5.4 | Cross Correlation Function of the First and Seventh Thermocouples (case 3) | 95 |
| A.3.1 | Example of Information Displayed to the Operator | 115 |
| A.4.1 | Flow Chart Illustrating the Generalized Consistency Checking Procedure | 119 |

## LIST OF ABBREVIATIONS

| | |
|---|---|
| A/D | Analog to Digital |
| DFBW | Digital Fly By Wire |
| FDI | Fault Detection and Isolation |
| GCC | Generalized Consistency Check |
| IMU | Inertial Measurement Unit |
| LLR | Log Likelihood Ratio |
| PWR | Pressurized Water Reactor |
| SPRT | Sequential Probability Ratio Test |

# CHAPTER 1

# INTRODUCTION

## 1.1 Problem Definition

In a large dynamic system such as a nuclear power plant, process signals are used for plant control, safety systems, and plant monitoring systems. Sensor outputs are also critical during post-accident diagnostics and transient recovery. Aerospace systems, chemical industry processes, manufacturing processes, etc., are examples of systems where sensors are used to measure temperature, pressure, mass flow rate, velocity, force (thrust), tension, rotational speed, thickness, chemical concentrations, etc. The monitoring of sensor anomalies is just as important as the detection of any other mechanical, electrical, or chemical malfunction of a component or process. The proper operation of sensors such as those in the emergency systems of a nuclear power plant is critical, and the consequences could be very severe if anomalies are left undetected.

With the advent of digital computers several approaches[1] have been developed to detect, isolate and identify sensor malfunctions. Some of the methods were never developed past the experimental stage, while others have been applied successfully in an industrial setting[2-6]. In industry, the detection and identification of faulty measurements, and (in some cases) the reconstruction of measurements, have been applied to avoid or reduce catastrophic consequences as well as to improve product quality and therefore improve productivity and

profit. The detection, isolation, and identification of sensor failures, and the reconstruction of signals is commonly known as signal validation. Examples of sensor malfunctions in a variety of applications are described in Chap. 2.

The current study is part of a larger, continuing study underway at the University of Tennessee that addresses the questions of sensor degradations. The overall objective is to develop a signal validation system that employs both traditional analytical tools as well as newer artificial intelligence methods. The specific objective is to employ the concept of a redundant array of measurements and obtain a practical analytical signal validation tool. Redundancy can be achieved through the use of multiple like or unlike sensors, or through the use of theoretical or empirical analytical models. The main consideration is that the signal validation system must be general, in the sense that it can be applied to any type of sensor. Analytical models are system specific and cannot be generalized.

Two of the previously used signal validation methods are combined in this work. The first, known as the generalized consistency approach, is used because of its ability to handle redundant sensors and to provide an estimate of the signal based on readings from all of the sensors. The second, known as the sequential probability ratio test is used because of its ability to sequentially check for sensor degradations, and to record the sensor degradation history. More details about these methods are presented in Sect. 1.2.

## 1.2 Development of Methodology

The concept of redundancy along with the sequential probability ratio test (SPRT) has been applied to detect sensor failures[1]. Redundancy has been achieved using Kalman filtering and other theoretical modeling approaches. The SPRT is then used to compare the actual signal and the analytical estimate. One drawback with this approach is that it is generally not an easy task to obtain a good analytical signal. The complexity of the model needs to be restricted if the signal validation system is to be used on-line.

In many industrial settings, hardware redundancy in the form of more than one sensor measuring the same state variable is available. This is commonly the case in nuclear power plants. The generalized consistency approach for redundancy management system[2] has been developed to deal with redundant arrays of sensors. This approach was successfully applied to problems in the aerospace and nuclear industry[3,4,5,6]. An algorithm for the detection of scalar measurement anomalies has been developed under a project sponsored by the Electric Power Research Institute[6]. This algorithm will be referred to as the Generalized Consistency Check (GCC) in the rest of this study.

The GCC is an on-line method for checking for inconsistencies among redundant sensors. Any two like measurements are defined to be inconsistent if the difference between their values is greater than a specified value. The redundant sensors are then used to obtain a weighted estimate of the signal at each sampling instant. It is important to note that for the GCC to be effective, at least three

signals must be available, measuring the same state variable. Where only two sensors are available, the required level of redundancy sometimes may be achieved using analytical modeling. The GCC does not require that all the signals be direct measurements. It is good practice to keep the analytical model as simple as possible for on-line applications. The models must be active in the sense that they should use the outputs from other validated sensors measuring other unlike process variables, to obtain the estimate. An analytical model may also be helpful in detecting common mode failures. In some applications the cost of redundancy may be a serious problem. In many cases, the increased product quality or the increased safety obtained using redundant arrays of sensors can offset the cost of installing and maintaining these sensors. This depends strongly on the location and purpose of a particular measurement. Recent advances in sensor manufacturing techniques allow the installation of several measuring elements in the same device[7]. This may also reduce the cost of redundancy. The cost of redundancy has not been a problem in nuclear and aerospace applications. In fact, in many cases redundant arrays of sensors are required by law[8].

The generalized consistency checking methodology has been successfully applied[5] to the detection of bias, drift, spikes, and abnormal noise type errors of fluid level sensors in a pressurized water reactor (PWR) pressurizer. It was possible to isolate the degraded sensor, but the kind of degradation could not be identified. Another problem with the GCC is that a decision at each sample is based on information available at that instant only. The degradation

history of sensors as represented by their states at previous samples is not used. A sensor may be inconsistent at a particular sampling instant due to a spurious noise spike. In a study by EPRI[6] this problem is overcome by assigning a cumulative index to each sensor. Everytime a sensor is inconsistent the index is increased by one, and whenever a sensor is consistent the index is decreased by one. When the index reaches a specified value a decision is made that the sensor is in the degraded state.

A statistical method developed by Wald[9] is ideal for solving this type of problem. This method is known as the Sequential Probability Ratio Test (SPRT), and it allows the optimum sequential checking of the state of a sensor based on specified missed and false alarm probabilities. A decision at a sample is based on information at that sample as well as information existing at previous samples. The behavior of the SPRT is also sensitive to different types of sensor failures. The combination of the generalized consistency check and the sequential probability ratio test forms the basis for the signal validation system used in this study.

### 1.3 Application and Verification

The basic algorithm developed by EPRI[6] is used to apply the GCC. This algorithm is extended to include the SPRT. The resultant algorithm is verified using simulated and experimental signals. A computer simulation may be used to generate transient signals containing constant bias and bias drift errors. A random number generator is used to generate normal and excessive Gaussian noise levels.

The method is also fully validated using data from an experimental facility constructed at the University of Tennessee. Bias and noise type errors are introduced in sensors installed at various locations in the facility (described in Sect. 4.3). Both steady state and transient situations are considered. This procedure requires signal conditioning, signal digitization and signal recording. An IBM PC is the main tool used for this study. Software was also developed to perform the digitization and processing of the signals.

The final verification of any signal validation system must be made at an existing industrial process. Work is currently underway to apply the system developed in this study to the Hot Line Facility (rolling mill) at the Aluminum Company of America Plant in Alcoa, Tennessee.

## 1.4 Outline of the Thesis

The purpose of Chap. 1 is to provide the problem definition, the objective and the general approach taken in this study. Chapter 2 contains the discussion of several case studies to familiarize the reader with different kinds of sensor-related problems encountered in different industries. A detailed discussion of how the approach to signal validation performed in this study differs from previous applications is also presented.

Chapter 3 is entirely devoted to the detailed development of the methodology used in this study. The practical application of the methodology used in the main algorithm is also described.

The verification and application of the signal validation system

is presented in Chap. 4. Several examples, including bias and noise errors for steady state and transient situations, are presented. The behavior of the transit time of air flowing through a section of piping is investigated using a signal correlation technique. This is one of the many features that can be used in a signal validation system that also uses artificial intelligence tools. The main approach is to obtain a data base that contains relationships between sensor outputs and other properties of a particular process.

Conclusions and suggestions for future work are given in Chap. 5.

# CHAPTER 2

# GENERAL APPROACH

## 2.1 Introduction

This chapter is devoted to providing examples of sensor failures encountered in industry, examples of applied signal validation systems, a description of the general approach taken in this study, and a discussion of the contributions of this thesis. The section discussing the general approach contains an explanation of the decision estimator and the signal flow path concepts. The contributions of this study include the development and verification of the methodology, and the development of a signal validation system at the University of Tennessee.

## 2.2 Case Studies

In the period between 1979 and mid-1984 Westinghouse nuclear power plants worldwide experienced 59 trips due to a faulty signal to the feedwater regulating valve operator[10]. Twelve of these trips were caused by malfunctions in sensor and sensor wiring. The average down time per trip is about 24 hours and the cost for replacement power is approximately $500,000 per day. It is clear that if a single trip can be avoided a considerable amount of money can be saved. A signal validation system provides a means for avoiding or reducing trips due to sensor failures and may provide a strong economic benefit. Examples of failure of temperature sensors may be found in

Ref. [11, 12, 13, 14, 15] and include calibration drift, improper installation, etc.

A current study[15] concerning the recognition of anomalous behavior of the Space Shuttle main engine points out that three types of sensor failures occur. The first is a jump in a pressure sensor output due to a sensor purge port failure. The second type involves noise and erratic behavior. The third type involves drift and bias. A bias failure could be caused by a drift in the fuel flow meter calibration.

Signal validation techniques were developed originally by the aerospace industry. Many of the techniques have been adapted for use in the nuclear industry. Increased safety concerns in the chemical industry have recently opened up the way for many possible applications of signal validation.

An example of the application of a signal validation system in the aerospace industry is an application of the Sequential Probability Ratio Test (SPRT) to dual flight control sensors aboard the F-8 digital fly-by-wire (DFBW) aircraft. As applied to the F-8 DFBW aircraft, the technique utilizes analytical redundancies obtained from relationships among variables measured with dissimilar instruments. The analytical redundancy is based on a model utilizing rotational kinematics, altitude kinematics and translational dynamics. The models are kept as simple as possible to allow for on-line application. An important feature of the SPRT is that it minimizes the actual number of observations necessary to make decisions based on specified missed and false alarm probabilities. The technique was

applied to actual flight test data and illustrated the reliable on-board detection and identification of failures in dual aircraft control system sensors.

A failure detection and isolation algorithm has also been developed for the Space Shuttle Inertial Measurement Units[18]. This algorithm is based on an improved version of the SPRT. (A detailed discussion of the modified SPRT will be given in Chap. 3.) The space shuttle contains three Inertial Measurement Units (IMU's). An analytical model and a Kalman filter is used to generate an independent measurement that can be used to check actual sensor outputs.

The signal validation system based on the SPRT allows the detection of "soft" or gradual degradation of sensors since it utilizes an accumulation of the sensor's measurement history. This can not be accomplished using a failure isolation system based on one sample only. In over 300 runs[18], no false isolations were made by the sequential technique. Another advantage is the small computer memory requirement, which makes the on-board implementation of the system practical.

A signal validation technique, based on the Parity Space Methodology, has been developed by the Charles Stark Draper Laboratory [19]. This technique has been applied successfully to the power-related instrumentation of the MIT-II reactor, [4], the instrumentation of a Boiling Water Reactor Suppression Pool [6], the Safety Parameter Display System of a Pressurized Water Reactor [20], and several other cases. A similar application of the Parity Space Methodology has been made to the measurements in the pressurizer of a

PWR[5,21].

The Parity Space implementation relies on the availability of redundant arrays of sensors. Simple physical models are often used to achieve the required levels of redundancy. The Fault Detection and Isolation (FDI) algorithm used in this study is based on the parity space methodology (also referred to here as the Generalized Consistency Check) as applied by the Charles Stark Draper Laboratory. A detailed description of this methodology, is presented in Chap. 3.

## 2.3 General Approach

Above we mentioned two signal validation methods: the Sequential Probability Ratio Test (SPRT) and the Generalized Consistency Check (GCC). These two approaches are combined to form the basis for the signal validation system of this thesis. To apply the SPRT to a particular sensor the output of the sensor must be compared to an estimate of the sensor output. This estimate is usually obtained using a Kalman filter. If a redundant array of sensors measuring the same variable is available, the readings of one or more of these sensors may be used to obtain an estimate. The GCC provides a computationally efficient procedure for managing redundant sensors and also provides an estimate of the signal. The estimate is a weighted average of all the sensor outputs based on their mutual inconsistencies. As will be discussed in Chap. 3, the SPRT and GCC complement each other.

The availability of redundancy is therefore an important feature of the signal validation approach. If only one signal is available

the signal validation is limited to the observation of unusual behavior by checking for instrument loop integrity or changes in the sensor time constant. When duplex redundancy exists, a signal validation process is capable of detecting a sensor failure, but no isolation or identification of the failure itself. Triplex redundancy provides the capability of detecting, isolating, and identifying the sensor failure. The signal can also be reconstructed in the form of a weighted average of all the sensor outputs. A further increase in the number of sensors improves the reliability of the failure detection, isolation, identification, and signal reconstruction process. The detection of common mode failure requires analytical measurement and other information previous to the failure.

Another important feature of the approach taken in this study is the extension of the signal validation procedure to the detection of increased noise levels in the signal. By filtering out the low frequency components of the signal, the fluctuating components of the signal are retained. The resultant signal is a random signal with a zero mean value. The SPRT can directly be applied to this signal. Since the mean is known to be zero, the GCC is not required to provide an estimate. As will become clear in Chap. 3, redundancy is also not essential for the detection of unacceptable signal fluctuation levels. However, more than one sensor is necessary to distinguish between process induced noise and sensor induced noise.

An important requirement in applying the SPRT to both bias and noise degradation is that the signal errors must form an uncorrelated sequence as a function of time. In the case of a sensor measuring a

changing process variable each sample depends on the sample taken at the previous time instant, and the signal is not uncorrelated with itself. The sensor degradation is a random event that can occur at any sample. To overcome the correlation problem, the sensor degradation is characterized by the error between the sensor value and an estimate of the sensor value. This error is an uncorrelated time sequence and the SPRT can be applied to detect unacceptable error magnitudes.

A similar problem exists in the case of random signals. A signal may be periodic as is often the case with 60 Hz contamination. This concern with correlation may be avoided by increasing the sampling interval [22], or subtracting one noise signal from another where redundant signals are available.

Redundancy may be achieved in several ways. First, a redundant array of like sensors may be used. This is generally referred to as direct redundancy. In many industrial environments it would be expensive to install and maintain hardware redundancy. Analytical modeling is a tool that has often been used to achieve required levels of redundancy for signal validation. Both physical and empirical analytical models are used. The models are kept as simple as possible to allow for on-line application. Another important requirement is that the models must be driven by validated sensor outputs. This is necessary to insure a continuous and realistic relationship between the model and the actual process. An example of an empirical model is the relationship between steam pressure and temperature at saturated conditions. By measuring temperature continuously using a reliable

sensor, the pressure may be predicted. Analytical models based on mass, energy, and momentum balances are also often used.

The arrangement of sensors is always plant and component specific. For a sensor validation system to be useful, it must be as general as possible to allow application to different plants or components. To achieve this, the approach taken by the Charles Stark Draper Laboratory [6] is adopted in this study. The approach is to design the Failure Detection and Isolation (FDI) algorithm as an integrated unit known as the Decision Estimation Module. The Decision Estimation Module has a variable number of inputs. These inputs may be outputs of different types of sensors or of analytical models. An illustration of a Decision Estimation Module is given in Fig. 2.2.1. The symbol for an "OR" gate is used since the number and types of inputs are left unspecified. The outputs from the Decision Estimation Module are the information to the operator about the detection of failures, and an estimate of the signal based on the inputs. As illustrated in Fig. 2.2.2, the Decision Estimator Module may be used in combination with signals, analytical models, and other Decision Estimation Modules to obtain a plant or component specific FDI system.

For any particular plant or component, a signal flow diagram illustrates the interconnections between sensors, analytical models, and Decision Estimation Modules. Fig. 2.2.2 represents an example of a signal flow diagram. Note that the analytical model is represented by an "AND" gate. This is done since all the inputs are necessary for a reliable output. The "OR" gate of the Decision Estimation Module illustrates that an output is possible even after one or more of the

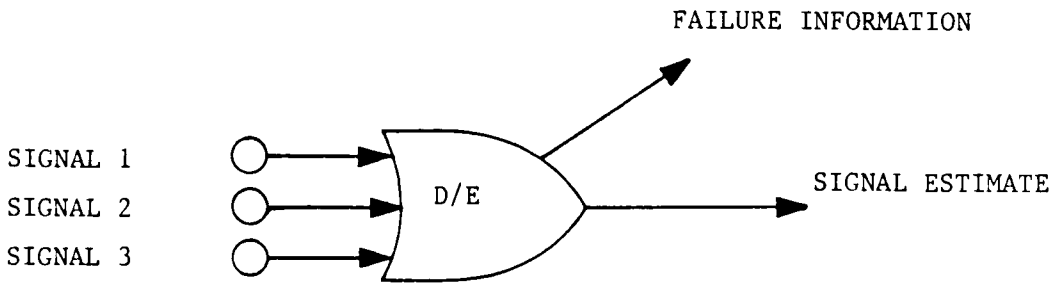



Figure 2.2.1 Illustration of a Decision Estimation Module with Three Input Signals.

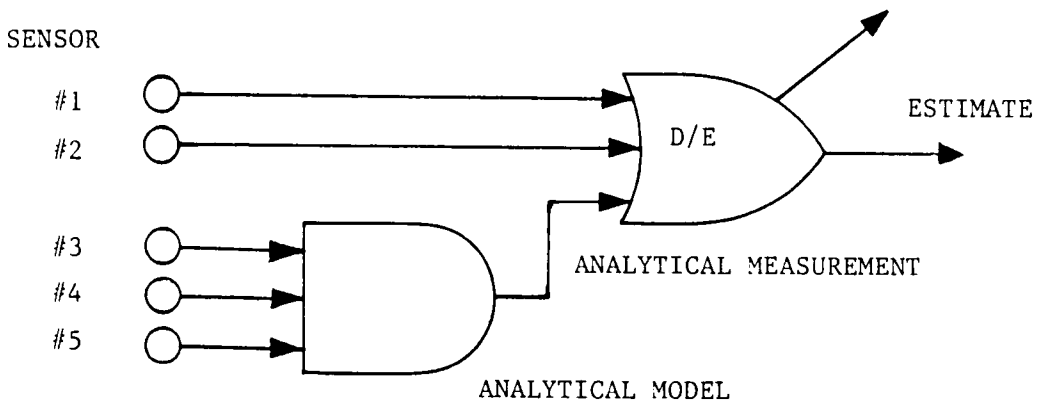



Figure 2.2.2 Example of a Signal Validation Flow Diagram.

inputs are degraded. It is also important that the inputs to the analytical model are from validated signals, or from reliable sensors.

There is no set procedure for designing a signal validation flow diagram. The design is an iterative procedure that is very plant or component specific. The signal validation flow diagram is influenced by the availability of hardware redundancy, analytical redundancy, sensor reliability, and the importance of the sensor function. Analytical redundancy is also helpful for the detection of common mode failures, and failures of components other than sensors.

## 2.4 Contributions of This Thesis

This study presents an improvement of the basic signal validation methodology as used in the past. Two methods, the SPRT and GCC, that were previously used independently are combined to provide a more efficient and complete system. The methodology is also extended such that both bias type and noise type degradations may be detected. This is best achieved by the separation of the fluctuation and mean value components of the signals.

To verify the extended methodology an experimental facility was constructed. The experimental facility consists of a closed air loop containing adjustable electrical heaters, an adjustable electrical fan, several thermocouples, and a pitot tube. The air loop is described in more detail in Sect. 4.3.

To record signals from the experimental arrangement a personal computer based digital data acquisition system has been developed.

Software has also been developed to process the signals and to obtain actual temperature readings in degrees F.

One of the main features of this study is the development of the decision estimator algorithm based on the modified methodology. The algorithm is presented in Appendix A, where operating instructions are also provided. The algorithm features appropriate operator display capabilities. To facilitate understanding of the signal validation procedure, plotting routines have been developed to represent changing signals and other important quantities such as the sequential probability ratio and the reconstructed signal.

Several different failure situations have been simulated using the experimental facility, and the results are presented in Chap. 4. Finally, the use of sensor diversity is illustrated through the use of temperature fluctuations from two thermocouples to monitor fluid transit time.

## 2.5 Summary

The first part of this chapter is devoted to the discussion of examples of sensor failures. Past applications of the SPRT and the GCC are also described. The general approach of this study contains the concepts:

- analytical and direct redundancy,
- the decision estimator module, and
- the signal validation flow path design.

The contributions of this study pertaining to the development and experimental verification of the methodology are outlined.

# CHAPTER 3

# METHODOLOGY

## 3.1 Introduction

In this chapter the basic sensor failure detection methodology will be developed. As discussed before, the Generalized Consistency Check and the Sequential Probability Ratio Test form the basis for signal validation. These two methods and their combination will be described in detail.

The Sequential Probability Ratio Test (SPRT) has previously been used along with a Kalman filter[1] to check for sensor failures. The Kalman filter provides an estimate of the signal. This estimate is based on an analytical model and previous values of the state variables. The question is whether this procedure can be improved by using a redundant array of sensors. Thus, if the estimate of the signal is based on redundant measurements or models it is expected to be better than an estimate based on a single model. In this study the Kalman filter is replaced by an estimate from the Generalized Consistency Check (GCC). The GCC provides an estimate that may be used with the SPRT. The particular sensor output is compared to the estimate of the output using the SPRT. When the sensor degrades the SPRT detects the difference between the measurement and the estimate that is based on other measurements.

As will be described later in this chapter, the noise is recorded separately by filtering the mean signal value by a high pass filter.

To check for noise degradations only the SPRT is required. No signal estimate is required, since it can be assumed that the noise signal has a constant zero mean. Only the SPRT is used for detecting noise degradations, and the SPRT and GCC are used to check for bias type degradations.

## 3.2 Generalized Consistency Check

The basis of the parity space, or generalized consistency check failure detection and isolation method, is the detection of inconsistencies among redundant sensors measuring the same quantity. Using the basic parity space methodology[2], Desai and Ray[23] developed a systematic failure detection algorithm that compares all possible pairs of signals. Specified error thresholds are used to check for an inconsistency between the two signals. This algorithm is powerful for fault isolation because of its simplicity and low memory requirements. One of the important factors is that no multiplicative operations are required for its operation. The only limitations are

- at least three signals must be available as inputs,
- a limited number of inputs may fail at the same time, and
- the magnitude of a failed input must exceed the assigned threshold.

The algorithm, as developed by Desai and Ray, does not require a detailed knowledge of the noise characteristics of a sensor because they did not apply the SPRT. In an application by EPRI[6], the signals were modeled as having uniform noise probability density functions. The symmetric error thresholds used to check for

inconsistencies between any two signals are selected to be equal to the sum of the noise and bias errors and is based on the sensor tolerance provided by the sensor manufacturer. The uniform density function representation is illustrated in Fig. 3.2.1. The inconsistency thresholds are represented by ± b. These thresholds may also be adjusted depending on the fault tolerance limitation for the specific application.

A detailed description of the Parity Space Methodology can be found in Ref. [2]. The basic procedure for determining whether a pair of scalar signals is inconsistent is now described. Suppose an error bound, $b_i$, has been assigned to each signal. If $m_i$ and $m_j$ form a pair of scalar measurements, the absolute value of their difference at time instant k, $|m_i(k) - m_j(k)|$, is compared to the sum of their error bounds, $|b_i + b_j|$. The pair of measurements is consistent if the following condition is satisfied

$$|m_i(k) - m_j(k)| \leq b_i + b_j, \qquad (3.2.1)$$

and inconsistent if the condition is not satisfied. Thus, if the difference between the two measurements is larger or equal to the sum of their error bounds, the pair is inconsistent.

If a total of $\ell$ different scalar measurements of the same variable exist it is necessary to check $\ell(\ell - 1)/2$ pairs of measurements. An index, $I_i$, is assigned to every measurement. This index is increased by one every time the particular measurement is inconsistent with another signal, i.e., every time Eq. (3.2.1) is not satisfied. In the mathematical form the procedure is given by

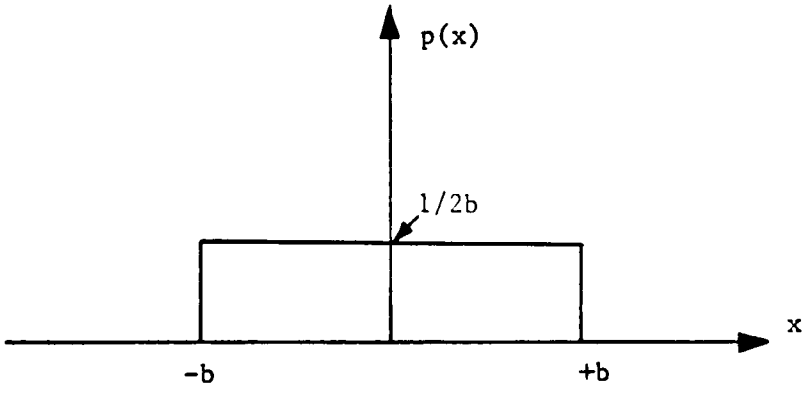



Figure 3.2.1 The Uniform Sensor Noise Probability Density Function.

$$I_i = \sum_{j=1}^{\ell} D[\,|\, m_i - m_j \,| < (b_i + b_j)\,], \qquad (3.2.2)$$

where $i = 1, 2, 3, \cdots, \ell$,

and $D[\theta] = \begin{cases} 0 & \text{if } \theta \text{ is true} \\ 1 & \text{if } \theta \text{ is false.} \end{cases}$

The value of this index varies between 0 and $\ell$-1 and provides an indication of the degree of inconsistency of each sensor. There are $\ell$ levels of inconsistency . The sensors with high indices are more likely to be degraded than those with lower indices. A measurement with $I_i = \max_j I_j$ is isolated as degraded. The management and isolation of the sensors when inconsistencies exist is a fairly complex procedure and is best understood by referring to the flow chart, Fig. A.4.1, given in Appendix A.

As examples of the assignment of the index, $I_i$, to the signals, consider the following two cases. (1) Assume that three sensors measuring the same process variable are available. In the first case assume the first sensor is inconsistent with both the second and the third sensors. Then the index, $I_i$, will have the value 2, while $I_2$ and $I_3$ are both equal to unity. The first sensor can then be isolated as inconsistent, and is weighted by zero in the weighting procedure described below. (2) Assume that three sensors measuring the same process variable are available. In this case assume that the first sensor is inconsistent with the second sensor only, while the second and third sensors are consistent with respect to each other. In this case, $I_1 = 1$, $I_2 = 1$, and $I_3 = 0$. The first and second sensors are

now considered partially inconsistent and all three sensors are used in the weighting procedure as described below.

The next important step is to estimate the value of the measured variable. To accomplish this, the index $I_j$ (defined above) of each sensor as well as a weighting function for each signal is required. The weighting function, $w_i$, may be taken to be inversely proportional to a power of the error bound. The estimate of the measurement, $\hat{x}$, is computed at every sampling instant. Examples of the estimate under different conditions are presented in Chap. 4. If $(\ell-k)$ measurements are identified as degraded, the estimate $\hat{x}$ is given by

$$\hat{x} = \frac{\sum_{i=1}^{k} w_i \, (\ell - 1 - I_i) \, m_i}{\sum_{i=1}^{k} w_i \, (\ell - 1 - I_i)} \qquad (3.2.3)$$

where $k$ = remaining number of consistent measurements,

$\ell$ = total number of measurements, and

$\ell - k$ = number of measurements with $I_i = \underset{j}{\text{Max}} \, I_j$.

Notice that measurements that are identified as degraded ($I_i = \underset{j}{\text{Max}} \, I_j$) are not used in the estimate, while all other measurements, $m_i$, with $I_i < \underset{j}{\text{Max}} \, I_j$ are used to obtain the weighted average. The weighting is according to the weighting factor, $w_i$, and to the index of each measurement, $(\ell - 1 - I_i)$. Thus, the larger the index $I_i$ the less the measurement , $m_i$, is weighted in the estimate.

When a large number of failures occur, $\left[ > \frac{\ell - 1}{2} \right]$, or when common mode failures occur, the algorithm checks the current measurements against the past value of the estimate, $\hat{x}$. The j values for which

$$| m_i - \hat{x} | \leq b_i, \quad i = 1,2,\cdots,j \tag{3.2.4}$$

is satisfied are then used to calculate a new estimate of the measurement using

$$\hat{x} = \frac{\sum_{i=1}^{j} w_i m_i}{\sum_{i=1}^{j} w_i} \tag{3.2.5}$$

If no such values of $m_i$ exist ($j = 0$) no estimate can be made and the algorithm terminates.

The algorithm also indicates whether the degraded sensors are reading high or low as compared to the estimate $\hat{x}$. Several examples are provided in the EPRI[6] report illustrating the working of the algorithm.

Up to this point the algorithm provides information about measurement inconsistencies at a single sampling instant. Since degradation might be caused by spurious noise spikes, more than one sample is required. Another index, Q, is assigned to each measurement. At each sample, this index is increased (decreased) by one every time a measurement is inconsistent (consistent). When the index reaches some predetermined value a decision can be made. This is not an optimum

procedure, since the maximum value of Q is fixed. The magnitude of a measurement degradation is not used. An optimum procedure, based on specified missed and false alarm probabilities, is discussed in Sect. 3.3. Several other advantages of this approach are also noted.

### 3.3 Sequential Probability Ratio Test

An often encountered problem in sensor failure detection is whether one judges a sensor as failed after only one sample or whether more than one observation is required. The question arises as to how many observations are needed. It is clear that a decision based on just one observation yields too high a risk for an erroneous decision. Several measurements are required to obtain a reliable result. At every new sample a mean and variance may be computed. These quantities may then be used to check whether the sensor is degraded or not. However, this procedure requires too many samples to obtain a meaningful mean and a meaningful variance. During the collection of these samples, a significant degradation of the process monitored may occur. A method is required to detect a failure using the degree of failure and the continuous behavior of the sensor, without having to calculate a new mean and variance at each sample. The Sequential Probability Ratio Test (SPRT) satisfies these requirements.

The problems discussed above exist because the failure of a sensor is often random in nature. The time and degree of failure is not known and cannot be predicted. The objective is to detect the failure as soon as possible with a small probability of making a wrong decision. The statistical nature of the SPRT makes it ideal for this

application. The SPRT is also optimum in the sense that a minimum number of samples is required to make a decision based on specified missed and false alarm probabilities.

The SPRT was originally developed by Wald[9]. The procedure consists of testing whether a sensor is more likely to be in the normal mode $H_0$, or in the degraded mode $H_1$. The decision is based on specified missed and false alarm probabilities. These quantities provide a measure of confidence for the decision.

This test requires that the sequential measurements with the same sensor be independent of each other. In the case where sensors measure dynamic process variables, the subsequent measurements are not independent. The value of each measurement depends on the value of the measurement at the previous sampling instant. This difficulty can be overcome by applying the SPRT to the difference between the sensor indicated value and the estimated value of the measured process variable. This difference is independent of the change in the actual value from sample to sample. The difference depends only on the degradation condition of the sensor and the usual noise present in the signal. This assumption may not be strictly satisfied in the presence of a sinusoidal component because the sinusoidal signal is a correlated noise. The actual value of the variable will be approximated by the analytically calculated estimate of the variable obtained as a result of the application of the GCC as discussed in Sect. 3.2. The advantages of combining the GCC and the SPRT will be discussed in Sect. 3.5. The SPRT is described below in detail.

To model the sensor outputs, a Gaussian (normal) probability

density function is used. This assumption is true in most cases because the signal fluctuations have a normal distribution. Let x be a process variable. Let the measured value of x be given by $x(t_m)$ at any time instant $t_m$, and let $\hat{x}(t_m)$ be the analytically computed estimate of x. Define the measurement residual between $x(t_m)$ and $\hat{x}(t_m)$ as:

$$s(t_m) = x(t_m) - \hat{x}(t_m). \tag{3.3.1}$$

The SPRT is applied to this difference. When the index of this test reaches a certain magnitude the sensor giving the measurement $x(t_m)$ will be judged as degraded. The residual $s(t_m)$ is computed at each sample during normal sensor operation. A mean $m_o$ and a variance $\sigma_o^2$ is computed for $s(t_m)$. This defines the Gaussian density function modeling the normal mode signal error:

$$p_o(s, m_o, \sigma_o^2) = \frac{1}{\sqrt{2\pi\sigma_o^2}} \exp\left[ - \frac{(s - m_o)^2}{2\sigma_o^2} \right] \tag{3.3.2}$$

The sensor failure can be stated in terms of a change in the mean m, or a change in the variance $\sigma^2$, or changes in both the mean and the variance.

When the mean reaches a degraded value $m_1$, the hypothesis $H_1$ that the sensor is degraded will be accepted. When the variance reaches a degraded value $\sigma_2^2$, the hypothesis $H_2$ that the sensor is degraded is accepted. If the mean and variance remain normal, the normal sensor hypothesis $H_o$ is accepted. Failure thresholds in terms of the Gaussian density functions $p_1(s, m_1, \sigma_1^2)$ are defined. Different

thresholds consisting of different degraded values of m and $\sigma^2$ are also possible. These density functions model the outputs from degraded sensors. A different density function is specified for constant bias, bias drift, excessive noise level and other kinds of failures. Bias and noise degradations will be of interest in this study. At this stage Gaussian density functions are available describing output errors of normal and degraded sensors. These density functions are illustrated in Fig. 3.3.1. The density functions may be obtained through experimentation, from design criteria, or from product quality criteria.

The basis for the SPRT lies in the likelihood ratio. To describe the computational procedure for obtaining the likelihood ratio a general sensor degradation will be considered. At each sampling instant the sensor error $s(t_m)$ is computed. This value is then used to calculate the value of the normal density function $p_o(s_m, m_o, \sigma_o^2)$ and the degraded density function $p_1(s_m, m_1, \sigma_1^2)$.
The likelihood ratio is then computed from

$$\lambda^1 = \frac{p_1(s, m_1, \sigma_1^2)}{p_o(s, m_o, \sigma_o^2)} . \qquad (3.3.3)$$

This ratio is updated at every sampling instant. The joint density function, modeling a sequence of independent samples, is obtained by multiplying the individual density functions together. For example, if a set of samples $x_i$, $i = 1, 2, \cdots, x_m$ is collected with a density function $p_i$ describing each sample in the set, the joint density

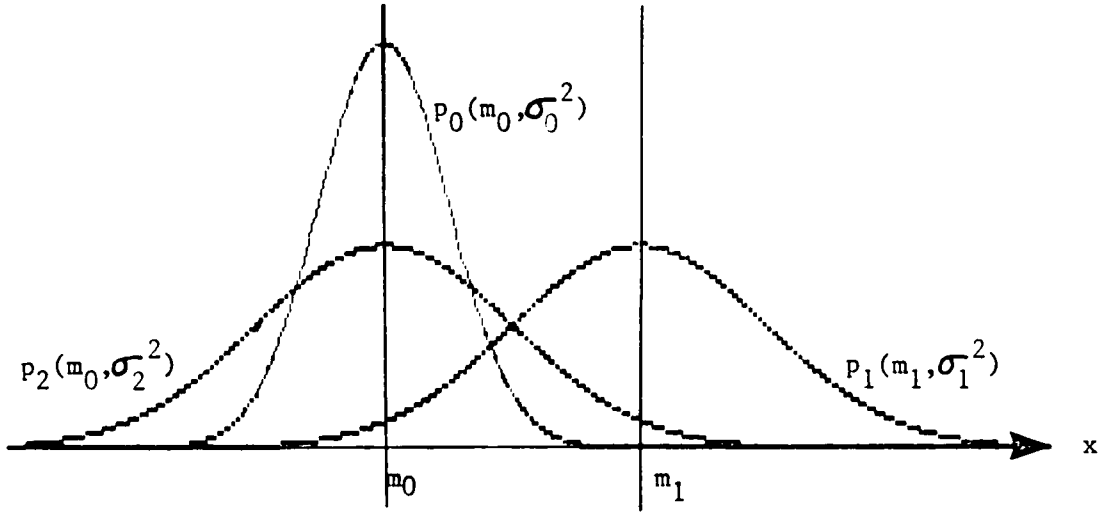



Figure 3.3.1 The Gaussian Density Functions Representing Normal and Degraded Sensor Outputs.

function modeling the occurrence of the set of samples, $x_1$, $x_2$, ..., $x_m$, is given by $p_1 \cdot p_2 \cdot p_3 \cdots p_n$. An overall likelihood ratio that is updated at each sample is given by

$$\lambda_m^1 = \frac{p_1(s_1/H_1)\ p_1(s_2/H_1)\cdots p_1(s_m/H_1)}{p_o(s_1/H_o)\ p_o(s_2/H_o)\cdots p_o(s_m/H_o)} \qquad (3.3.4)$$

where $H_o$ and $H_1$ represent the normal and degraded means and variances. By taking the natural logarithm on both sides of the equation, the multiplication procedure becomes additive.

This is given by

$$\lambda_m = \ln(\lambda_m^1) = \ln \left[ \prod_{i=1}^{m} \frac{p_1(s_i/H_1)}{p_o(s_i/H_o)} \right]$$

$$\lambda_m = \sum_{i=1}^{m} \ln \left[ \frac{p_1(s_i/H_o)}{p_o(s_i/H_o)} \right] . \qquad (3.3.5)$$

Rewriting Eq. (3.3.5) the following equation is obtained

$$\lambda_m = \lambda_{m-1} + \ln \left[ \frac{p_1(s_m/H_1)}{p_o(s_m/H_o)} \right] . \qquad (3.3.6)$$

Equation (3.3.6) is the form that will be used in this study.

Since the sensor error, $s_i$, is assumed to be independent from sample to sample and is modeled by a Gaussian density function, the density functions $p_1(s_m, m_1, \sigma_1^2)$ and $p_o(s_m, m_o, \sigma_o^2)$ of Eq. (3.3.6) may be replaced by their equivalent Gaussian density function expressions. The following equation is derived in a general form considering both

bias and noise degradations using $p_o\ (s_m,\ m_o,\ \sigma_o^2)$ and $p_1(s_m,\ m_1,\ \sigma_1^2)$:

$$\lambda_m = \lambda_{m-1} + \frac{s_m^2}{2}\left[\frac{1}{\sigma_o^2} - \frac{1}{\sigma_1^2}\right] + \left[\frac{m_1}{\sigma_1^2} - \frac{m_o}{\sigma_o^2}\right] s_m + \frac{1}{2}\left[\frac{m_o^2}{\sigma_o^2} - \frac{m_1^2}{\sigma_1^2}\right] + \ln\left[\frac{\sigma_o}{\sigma_1}\right] \quad (3.3.7)$$

Equation (3.3.7) may be simplified by distinguishing between mean and variance changes. To check for only bias or mean degradations, $\sigma_1^2$ is set equal to $\sigma_o^2$. For a normal sensor the mean of the sensor error should theoretically be equal to zero, $m_o = 0$. Using these simplifications Eq. (3.3.8) is derived for bias degradation detection:

$$\lambda_m = \lambda_{m-1} + \frac{m_1}{\sigma^2}\left(s_m - \frac{m_1}{2}\right) \quad . \quad (3.3.8)$$

Equation (3.3.8) is used to detect both bias drifts and constant bias errors in this study.

To detect excessive noise levels in the signals Eq. (3.3.7) can be reduced to Eq. (3.3.9) by setting $m_1$ equal to $m_o$. To detect noise degradations only, $m_1$ and $m_o$ are set equal to zero. This gives

$$\lambda_m = \lambda_{m-1} + \frac{1}{2}\left[\frac{1}{\sigma_o^2} - \frac{1}{\sigma_1^2}\right] s_m^2 + \ln\left[\frac{\sigma_o}{\sigma_1}\right] . \quad (3.3.9)$$

Equations (3.3.8) and (3.3.9) can be used to detect bias and noise degradations respectively.

The question of how to use Eqs. (3.3.8) and (3.3.9) to detect sensor failures is now considered. The likelihood ratios defined by Eqs. (3.3.8) and (3.3.9) are updated at each sampling instant. For a normal sensor the ratio would decrease and eventually reach a specified bound, A, smaller than zero. When the ratio reaches this bound the decision is made that the sensor is normal, and the ratio is initialized by setting it equal to zero. For a degraded sensor the ratio would increase and eventually reach a specified bound B. When the ratio is equal to B the decision can be made that the sensor is degraded. At this point the ratio is reinitialized to zero. The increase in the ratio for a degraded sensor and the decrease in the ratio for a normal sensor can be understood qualitatively by referring to Fig. 3.3.1. For the case of bias degradations the measurement error, s, would fall closer to the degraded mean, $m_1$, than to the normal mean, $m_0$. The value of the density function $p_1$ at s would be larger than the value of the density function $p_0$ at s. Since $p_1$ appears in the numerator of the likelihood ratio, an increase in this ratio occurs. This behavior of the ratio is illustrated in Fig. 3.3.2. A similar argument can explain the increase in the likelihood ratio for a variance degradation, or the decrease in the ratio for normal sensor behavior.

The bounds A and B used to make the decision whether a sensor is normal or degraded, are calculated using the following equations as derived by Wald[9]:

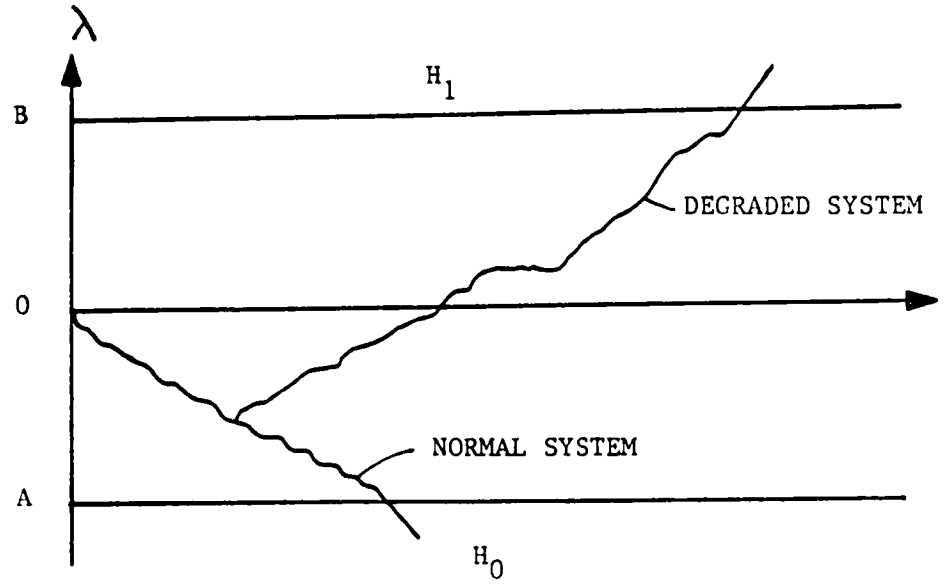



Figure 3.3.2 The Trajectory of the Likelihood Ratio Function, $\lambda$.

$$A = \ell n \left(\frac{\beta}{1 - \alpha}\right) \tag{3.3.10}$$

$$B = \ell n \left(\frac{1 - \beta}{\alpha}\right) \tag{3.3.11}$$

where, $\alpha$ = specified probability of false alarms,

$\beta$ = specified error probability for missed alarms.

The false and missed alarm probabilities are defined below with reference to Fig. 3.3.3.

- False-alarm probability = $\int_{x_o}^{\infty} p_o(x)dx$,

- Missed-alarm probability = $\int_{-\infty}^{x_o} p_1(x)dx$

where $x_o$ = true measurement,

$p_o(x)$ = normal mode ($H_o$) density function, and

$p_1(x)$ = degraded mode ($H_1$) density function.

The false and missed alarm probabilities are specified depending on the specific application. They are also an indication of the confidence in the decision.

The SPRT as developed by Wald is inefficient since decisions are made about the failure as well as the normal mode. If an a priori assumption is made that all the sensors are initially in normal

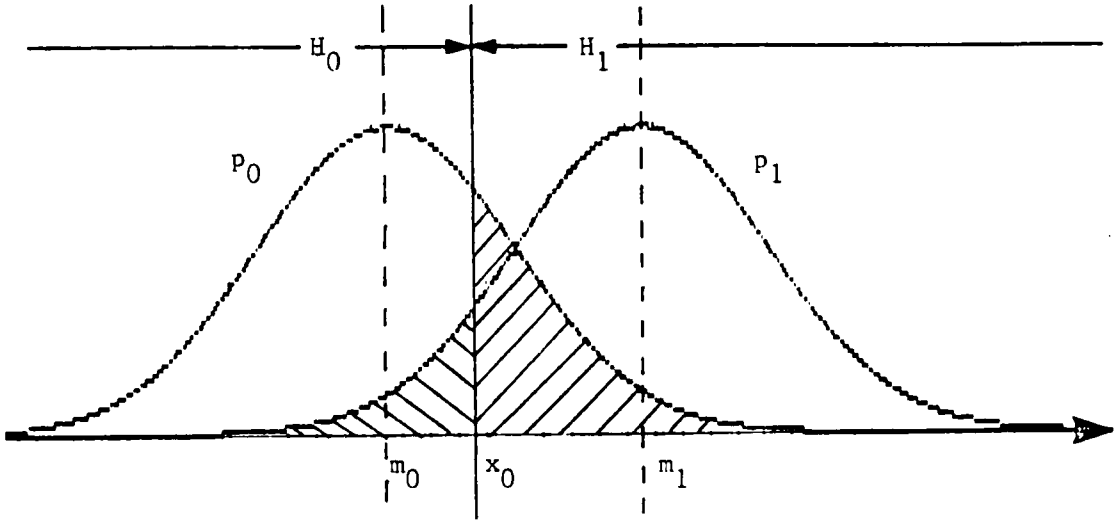



Figure 3.3.3 The Definition of False and Missed Alarm Probabilities.

operating condition, it is not necessary to check for the normal mode. A modified version of the SPRT that checks only for the failure of sensors is discussed in Sect. 3.4. The modified SPRT has several advantages over the conventional test. These are also discussed in Sect. 3.4.

## 3.4 The Modified Sequential Probability Ratio Test

A study by Chien[18] showed that the SPRT as developed by Wald[9] may be improved for the specific application of sensor failure detection. The original test checks for both the normal mode and the degraded mode of a sensor. Chien pointed out that if an a priori assumption is made that a sensor is operating in the normal mode at the start of the SPRT, it is only necessary to check for the degraded sensor mode. This one-sided approach improves the original SPRT in several ways. First, in the original SPRT the Log Likelihood Ratio (LLR) experiences a negative drift during normal operation. When a degradation occurs at a random instant the LLR starts to drift in the positive direction. The negative drift must first be cancelled and this is responsible for a time delay, t, as illustrated in Fig. 3.4.1. Second, since decisions are made concerning both the normal and degraded states of the sensors, additional observation time is required to make a reliable decision about either mode. If only the degradation check is performed, less observation time is required. Third, if a sensor is assumed to operate in the normal state at the beginning of the test, the concept of a missed alarm is not defined. It can be shown that if a sensor degradation does occur the

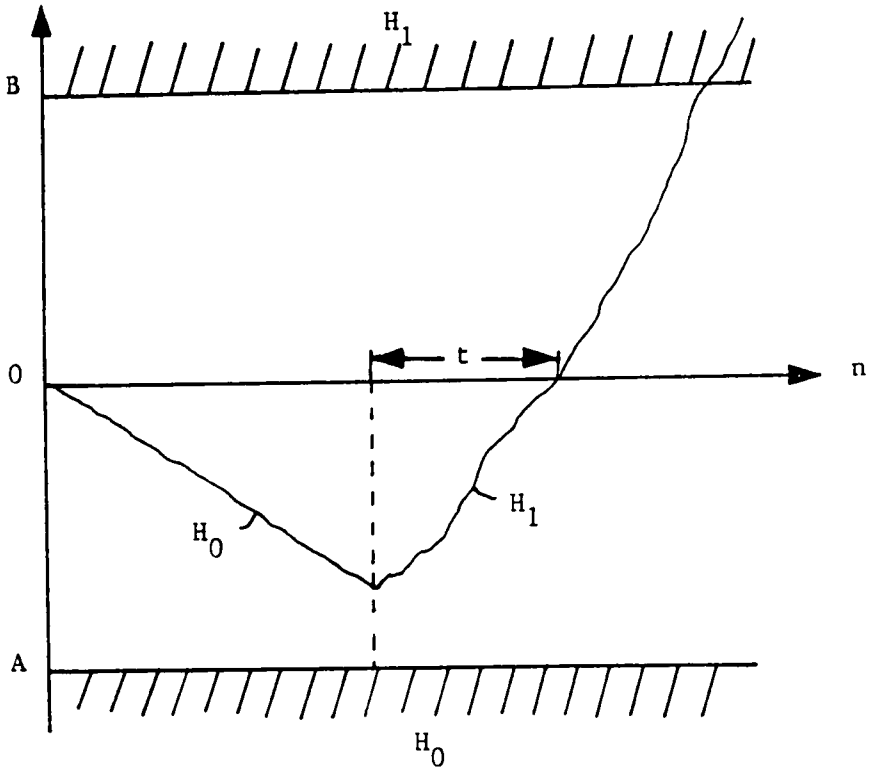



Figure 3.4.1 Illustration of the Time Delay, t, in the Wald System.

probability of its ultimate detection is unity[18]. The implementation of the one-sided failure isolation system is described in this section.

The LLR, $\lambda_n$, represents the degradation information of a sensor based on n samples. If the a priori assumption is made that the sensor is in the normal state, a drift of $\lambda_n$ in the negative direction does not provide any useful information, since this can be expected from a sensor in the normal state. The modification of Wald's test is accomplished through a stochastic control action. The control action is the restoration of $\lambda_n$ to zero every time $\lambda_n$ satisfies,

$$\lambda_n < 0. \tag{3.4.1}$$

If the LLR drifts to values larger than zero the sensor is more likely to operate in the degraded state and more information is required. No control action is applied in this case. This is illustrated in Fig. 3.4.2. The modified test becomes a stochastic control and decision problem. Chien shows that the optimum control action is indeed the restoration of the LLR to zero every time it is smaller than zero.

The bounds A and B as used by Wald are no longer valid. A new bound $B_1$ is derived, and unlike A or B it is not based on a missed alarm probability. If the bound B is used the control action would increase the false alarm probability considerably. Due to the nature of the modified SPRT it is more convenient to specify a mean time, T, between false alarms as the false alarm criterion. It has been shown[18] that the relation between the new bound $B_1$, and the false

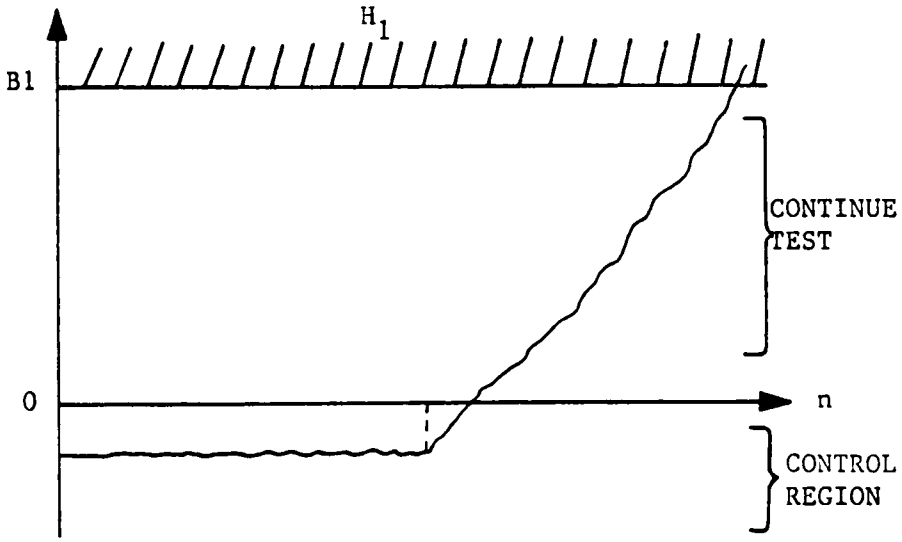



Figure 3.4.2 Illustration of the Modified SPRT.

alarm criterion, T, is given as

$$T = \frac{2\sigma_1^2}{\theta_1^2} (e^{B_1} - B_1 - 1). \qquad (3.4.2)$$

This expression is obtained by minimizing the mean time to detection after a degradation has occurred.

An equation may also be derived that describes the relationship between Wald's two-sided and Chien's one-sided system.

$$e^{B_1} - B_1 - 1 = -\left[ B + A \frac{e^B - 1}{1 - e^A} \right] . \qquad (3.4.3)$$

where $A = \ln \left(\frac{\beta}{1 - \alpha}\right)$ ,

$B = \ln \left(\frac{1 - \beta}{\alpha}\right)$,

$\alpha$ = false alarm probability,

$\beta$ = missed alarm probability.

Since it is intuitively more convenient to use $\alpha$ and $\beta$ as opposed to the expected time between false alarms, T, it is informative to understand the relationship between $\alpha$, $\beta$ and T. Substituting Eq. (3.4.3) into Eq. (3.4.2) the following equation relating $\alpha$, $\beta$, and T results

$$T = - \left(\frac{2\sigma^2}{\theta^2_1}\right) \left[ B + A \frac{e^B - 1}{1 - e^A} \right] . \qquad (3.4.4)$$

This relationship is presented graphically in Fig. 3.4.3. The false alarm probability $\alpha$ is varied from $\alpha = 10^{-4}$ to $\alpha = 4 \times 10^{-2}$. Since the missed alarm probability is zero for Chien's test, three cases of small $\beta$ are presented ($\beta = 0.001$, $\beta = 0.0001$, and $\beta = 0.00001$).

It is important to maintain a low false alarm probability ($\alpha \leq 1\%$) in any failure detection system. It can be seen from Fig. 3.4.3 that a large T ($T \approx 10^4$) corresponds to a small $\alpha$. For large values of T Eq. 3.4.2 may be simplified to

$$B_1 = \ln \left[ \frac{\theta_1^2}{2\sigma_1^2} \cdot T \right]. \tag{3.4.5}$$

This equation allows the easy calculation of $B_1$ for practical applications. Notice that $B_1$ is based on two considerations. First, the accepted tolerance on the sensor that considers normal fluctuations depending on the application. This is given by the presence of the factor $\frac{\theta_1^2}{2\sigma_1^2}$. The tolerance on mean fluctuations is given by $\theta_1^2$, and the tolerance on the noise is given by $\sigma_1^2$. The expected time between false alarms, T, represents a trade-off between the time it takes to detect a degradation and the frequency of false alarms. A large value of T will reduce $\alpha$, but it increases the mean time to detection. The selection of a value of T depends on the specific application.

If $\lambda_n$ is computed for a time greater than T and $\lambda_n$ does not exceed the bound $B_1$, then $\lambda_n$ is reinitialized to zero.

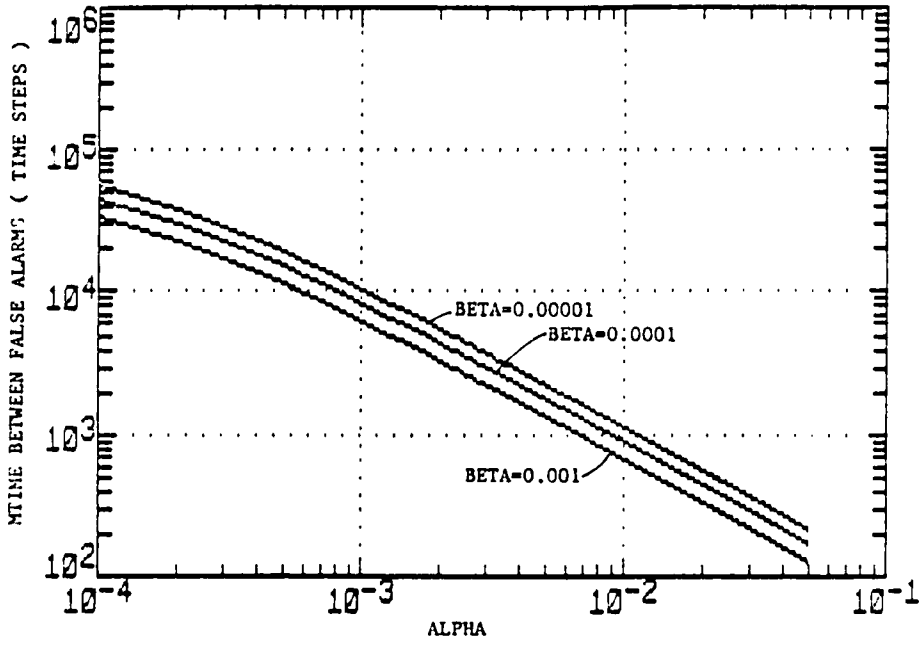



Figure 3.4.3 The Relationship Between Chien's Mean Time Between False Alarms and Wald's Missed and False Alarm Probabilities.

## 3.5 Detection of Bias Failures

To detect bias drift and constant bias type degradations a combination of the GCC and the modified SPRT is implemented. It has been shown that both types of failures may be detected using the same failure detection system[22]. The GCC provides the estimate of the signal, $\hat{x}_m$, while the modified SPRT uses $\hat{x}_m$ to isolate and identify failures. The GCC also provides information about the relative consistencies among sensors at each sampling instant. The extention of the SPRT to the detection of excessive signal noise levels is discussed in Sect. 3.6.

The GCC and modified SPRT are incorporated in the algorithm "CONSIST.EXE". This algorithm forms the decision estimation module for the signal validation system discussed in Chap. II. The inputs and outputs as well as a listing of the algorithm are provided in Appendix A. The GCC is contained in the algorithm according to the procedure discussed in Sect. 3.2 and the flow chart presented in Fig. A.4.1. in Appendix A. The GCC is implemented at every sample. The SPRT is included in the algorithm as a subroutine that is called at every sample as soon as the GCC is able to yield an estimate of the signal.

At every sample the GCC calculates a weighted average, $\hat{x}_m$, of the sensor outputs according to Eq. (3.2.3). The GCC also provides the operator with "consistency" messages. If all the sensors are used in the estimate this is indicated by a corresponding message. If a subset of the sensors is used to obtain the estimate a "consistency" message indicates which sensors are inconsistent. Consistency

messages also indicate partially inconsistent sensors and their particular weighting in the estimate. Another important situation is when a majority of the sensors are inconsistent. In this situation, the algorithm compares each sensor against the previous value of the estimate, $\hat{x}_{m-1}$, for consistency and calculates a new estimate based on Eq. (3.2.5). A "consistency" message indicating this situation is also given. When all sensors are inconsistent with respect to each other as well as the previous estimate, no new estimate is calculated. The algorithm provides a "consistency" message pointing out that all sensors are mutually inconsistent or that a common mode failure may exist and continues to the next sample. Since the GCC is applied independently from sample to sample, the "consistency" messages should be considered as warnings only. Failures are indicated by the SPRT, as will be discussed below. The messages generated by the GCC serve to keep the operator up to date from sample to sample. The estimate of the signal is also displayed to the operator.

After the estimate, $\hat{x}_m$, is computed for a particular sample, the subroutine that applies the SPRT is initiated. First, the sensor error, $s_m$, is computed for each sensor. The likelihood ratio, $\lambda_m$, is computed according to Eq. (3.3.8) for each sensor. If $\lambda_m$ is less than zero the control rule of the modified SPRT is applied and $\lambda_m$ is set equal to zero. If $\lambda_m$ is greater than zero it is compared to the bound, $B_1$, as determined by Eq. (3.4.5). If $\lambda_m$ exceeds $B_1$ the decision is made that the sensor corresponding to the particular $\lambda_m$ experienced a bias degradation. A failure message indicating the sensor number and identifying the bias error is displayed to the operator.

At this point, the algorithm reinitializes and the procedure is repeated. The algorithm also reinitializes after a time, $t > T$, has elapsed without any failures. The time, T, is the specified mean time between false alarms.

It must be pointed out that the SPRT has the ability to detect slow degradations before the GCC may detect any inconsistencies. This allows the detection of incipient failures. This is possible since the likelihood ratio is based on the sensor history and not on a single sample. On the other hand, if a sensor failure with a large magnitude suddenly appears the SPRT will detect the failure in the minimum number of samples. For extreme failures a single sample is sufficient.

The procedure followed during the signal validation process is illustrated by the flow chart presented in Fig. 3.5.1. This flow chart also includes the procedure for the detection of noise failures as discussed in Sect. 3.6.

## 3.6 Detection of Noise Failures

To detect noise type degradations the SPRT as developed by Wald, [9] is implemented. The modified SPRT is not used since an appropriate expression for the missed alarm criterion does not yet exist for the case of noise degradations[22]. On the other hand, Wald's original SPRT may be applied without any modification to the detection of changes in the noise levels[24]. The first step in checking for noise or variance degradations is to separate the fluctuating and mean value components of a signal using a high pass

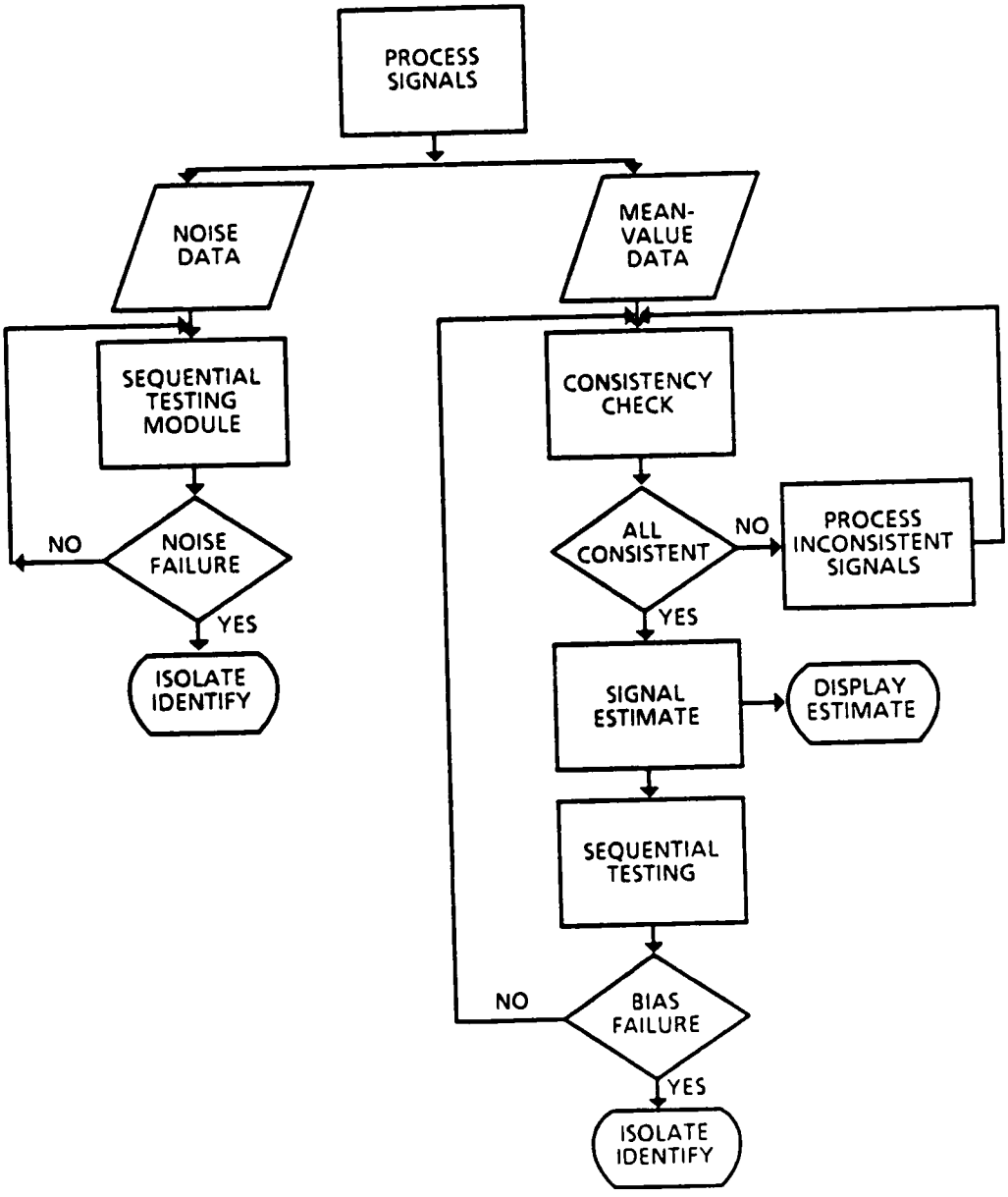



Figure 3.5.1 Flow Chart Illustrating the Implementation of the Signal Validation Methodology.

filter. The advantages of separating the fluctuations from the mean value are discussed below. This is a new feature of the present study. The implementation of the SPRT to the noise signals in the algorithm is also discussed.

To enable the distinction between bias and noise type degradations the noise and mean value signals are separated and then recorded. If the SPRT for noise degradation is applied to the original signal, any bias degradation is also detected as a noise degradation. This makes the distinction between the two types of failures difficult. After the separation is accomplished, bias and noise errors are detected independently. The separation may be achieved digitally, or using conventional high pass filters.

There are several advantages of recording the fluctuating components separately. A high resolution in the recording of the fluctuating component is possible by amplifying this portion of the signal to the full range of the analog to digital converter. The SPRT may readily be applied to check for noise degradations in transient cases. The noise signals may be used to provide a large amount of additional information about the sensors and the system. Any periodic type of failure can also be detected even if its amplitude is too small for it to be a bias failure.

The disadvantage of separating the noise and bias failure checking is the need to record additional signals. In the algorithm used in this study, the detection of noise failures is provided as an option depending on the operator's needs.

The detection of noise degradations using the SPRT is provided as

a subroutine that is called at every sample in the algorithm "CONSIST." The SPRT is implemented in the form described in Sect. 3.3. Since the mean value of the noise signal is known to be zero, no estimate, $\hat{x}_m$, is required. The SPRT is applied independently of the GCC. Since the estimate, $\hat{x}_m$, is by definition equal to zero the sensor error defined by Eq. (3.3.1) is equal to the noise signal $x(t_m)$. The expression for the log likelihood ratio, as defined by Eq. (3.3.9) for noise degradations becomes

$$\lambda_m = \lambda_{m-1} + \frac{1}{2}\left(\frac{1}{\sigma_o^2} - \frac{1}{\sigma_1^2}\right) x_m^2 + \ln\left(\frac{\sigma_o}{\sigma_1}\right) .$$

This is the expression that is used to compute the log likelihood ratio at each sample for each sensor. This ratio is then compared with the bounds A and B, as given by Eq. (3.3.10) and (3.3.11), respectively. If the bound B is exceeded by $\lambda_m$ the decision is made that the sensor has experienced a noise degradation.

It was stated that since the mean value of the signal is known to be zero, no estimate of the signal and therefore the GCC is required. This implies that redundancy is not necessary for the detection of excessive noise levels in the signals. Redundancy is necessary to detect whether the increase in the noise is due to the sensor or the process. The SPRT may be applied to the difference between two noise signals, or independently to different noise signals. Increases in noise levels in a particular sensor and not in others can be detected and will indicate that the excessive noise is generated in the sensor instead of the process. If excessive noise levels exist in all

sensors the noise is likely to be generated in the process.

The SPRT is defined for a sequence of samples that is uncorrelated. In real systems completely uncorrelated noise signals do not exist. One example of a correlated signal is the commonly found periodic 60 Hz contamination. The correlation concern may be reduced by increasing the sampling interval[22]. This is an effective solution for weakly correlated signals. For the more common cases of gradual and small increases in the variance the increase will not provide a problem. Also, since the fluctuation signals are considered separately from the mean value signals, the detection of bias degradation is not affected by the increase in the time step. Another option that can be used in the case of sequences that are strongly correlated, is to subtract one noise signal from another and apply the SPRT to the resultant measurement residual. It must be pointed out that the SPRT is capable of detecting increases in the signal noise levels that are strongly correlated. This is illustrated by the example containing a periodic noise contamination presented in Chap. 4. It must however be remembered that there is some approximation involved in applying the SPRT to such a case. The correlation problem is considerably reduced in the application of the SPRT to the bias degradation case, since the SPRT is applied to the measurement residual, $s(t_m)$.

## 3.7 Summary

The methodology used in the signal validation system is described in this chapter. First, the GCC is described in detail. The procedure for detecting inconsistencies among sensors measuring the same

variable, and the reconstruction of the signal using a weighted average is presented. The SPRT, as applied to both bias type and noise type degradations, is described in detail. Improvement to the original two sided SPRT is also pointed out. The implementation of the combined SPRT and GCC to bias degradations, and the SPRT to noise degradations is developed in the algorithm "CONSIST". The two most important new features of the methodology developed in this study are the combination of the GCC and the SPRT, and the extension of the SPRT to variance degradations. The problems that may occur in applying the methodology to correlated sequences are also discussed.

# CHAPTER 4

# APPLICATION OF THE METHODOLOGY

## 4.1 Introduction

This study is not complete without the verification of the methodology through its application to actual signals. Since signals from actual industrial processes are not readily available, we considered two options. (1) Signals can be generated using computer simulation. Any analytical model of a real process or system can be used to generate signals containing a variety of different failures. A Gaussian random number generator may be used to add noise to the simulated signals. (2) Signals can be obtained from an experimental facility. Such a facility should yield real signals, and also be flexible enough to allow the simulation of a variety of different failures. Since signals recorded using an experimental facility are more realistic than those generated by a computer, emphasis is placed on the application of the methodology to experimental signals.

To obtain experimental signals an air flow loop was constructed as part of this study. The flow loop contains three adjustable heaters, a variable speed fan, flow sensors, and several temperature sensors. For data acquisition and analysis, a signal processing and signal digitization system has been developed using a microcomputer. The signal recording system consists of both hardware and software components. The hardware components include amplifiers, filters, and an analog to digital (A/D) conversion board for the microcomputer. The

software includes an algorithm for performing the actual digitization, an algorithm converting the stored integer values to actual voltages and temperatures, and an algorithm to display the signals in a desired format.

This chapter is devoted to a description of the experimental facility, the software development, and detailed discussion of the application of the signal validation system to several different failure situations.

## 4.2 Software Development

During the course of this study several algorithms have been developed. These algorithms are outlined in this section. The computer algorithms are used to collect and record data, process the recorded data, display the processed data, and perform the signal validation.

The most important of all the algorithms is the program "CONSIST". This user-friendly algorithm forms the central part of the signal validation system. The algorithm performs the generalized consistency check (GCC) on a redundant array of signals according to the procedure described in Sect. 3.2 and illustrated in Fig. 3.2.2. The algorithm also utilizes the weighted average of the signals as provided by the GCC and applies the sequential probability ratio test (SPRT) to each sensor. The option also exists to detect variance type degradations in addition to the standard mean degradation detection. Appendix A serves as a user's guide for the algorithm "CONSIST.FOR." A detailed description of the required inputs, and the generated output of the algorithm is presented in the appendix. The file structure and

format of the input signals are also described. Appendix A also contains a complete listing of the algorithm "CONSIST.FOR" as well as a sample output table as generated by the algorithm. A detailed description of results obtained by the application of the algorithm "CONSIST.FOR" to several different failure situations is presented in Sect. 4.4.

The first algorithm that must be executed in the signal validation procedure is "DATAGET". This program controls the analog to digital conversion procedure. Using this algorithm up to sixteen signals may be digitized. The minimum digitization frequency is 13 Hz, while the maximum digitization frequency is 27.5 KHz (for one signal). The analog to digital conversion is performed using a 12 bit data translation A/D converter. The algorithm has the capability of specifying the resolution of the signals. This is accomplished by specifying a gain value. The gain value specifies a maximum voltage to cover the full range of the A/D converter. The possible gain values are 1 (10V), 2 (5V), 4 (2.5V), and 8 (1.25V). The number in the parentheses represent the maximum voltage (signal range) that can be digitized corresponding to each gain setting. The data is first stored in the central computer memory and is then written to a diskette by the algorithm. A listing of the program "DATAGET.FOR" is given in Appendix B.

The next step in the signal validation procedure is the conversion of the signals recorded in the integer form to actual temperatures in degrees Fahrenheit. This is accomplished through the implementation of the algorithm "GDATRAN". To obtain actual temperatures the integer

values are first converted into actual voltages. For this conversion a gain divider must be specified for each signal. The gain divider depends on the amplification setting of the amplifiers and on the gain specification in the algorithm "DATAGET." "GDATRAN" then computes the actual voltage signals from the reference junction sensor and each thermocouple. The reference junction and thermocouple voltages are used along with a thermocouple calibration table to compute the temperature reading of the thermocouples in deg. F. The temperature values are then stored on diskettes in the correct format as required by the program "CONSIST". A listing of "GDATRAN.FOR" is given in Appendix C.

During this study it has been necessary to display signals to confirm the performance of the data acquisition and conversion algorithms. It is also necessary to display the output variables of the program "CONSIST". For this purpose the algorithm "PLOT1" has been developed. The algorithm provides the capability to plot as many as three signals on the same graph. A maximum number of 2000 samples per signal may be plotted as a function of time. Graphs can be made on the video screen or the printer. The file structure of the input files are the same as those for "GDATRAN" and "CONSIST". The listing of "PLOT1.FOR" is given in Appendix D.

For the purpose of integrating the data collection, data conversion, and signal validation algorithms, consistent data file structures are used in all the algorithms. The file structure is described in Appendix A.

## 4.3 Hardware Development

In order to obtain real signals to validate the signal failure detection system, an experimental closed air loop was constructed as part of this study. The air loop consists of a 55 gallon steel drum with an inlet at the lower section and an outlet at the upper section. The inlet and outlet are connected by a 24 ft loop of aluminum piping of six inches inner diameter. A variable speed air-blowing fan is connected to the inlet section directing air flow into the drum. The drum contains three adjustable electrical heaters. The heaters are used to heat the air to different temperatures. Thermocouples have been installed in several locations along the piping. The most important of which are three thermocouples located at the outlet from the drum. A pitot tube measuring the air flow velocity has also been installed. The piping contains a test section that can be replaced if the loop is to be used for other experiments, such as temperature sensor response time testing. The whole arrangement including the drum, the piping, and the fan is insulated to reduce heat loss from the system. A schematic diagram of the air flow loop illustrating the locations of the sensors is shown in Fig. 4.3.1.

Since thermocouples are used to measure the air temperature, a reference junction was constructed so that the reference junction temperature may be used to determine the actual temperature values. The reference junction may be used with up to eight thermocouples and consists of an insulated box containing two copper blocks for each thermocouple. The copper blocks act as heat sinks providing a stable junction temperature. The reference junction temperature is measured

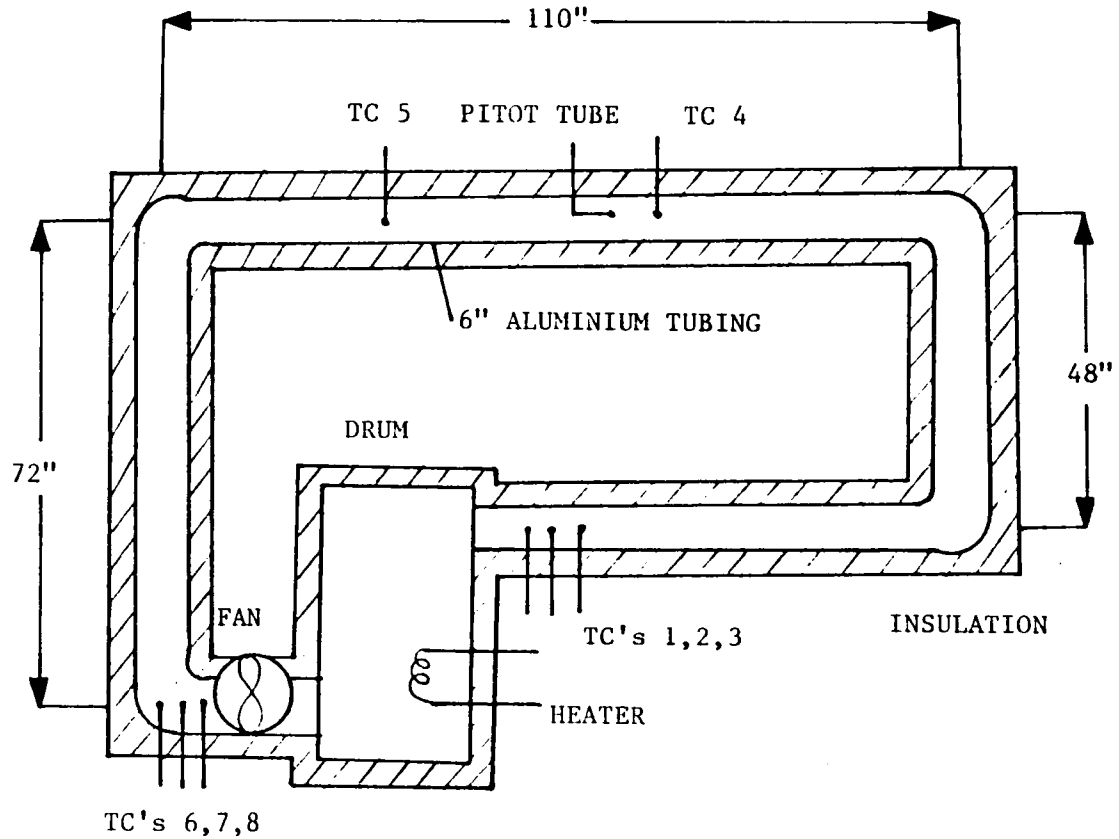



Figure 4.3.1 Schematic Diagram of the Air Flow Loop illustrating the Sensor Locations.

using an integrated circuit temperature detector (Analog Devices Model AD590). The integrated circuit temperature detector yields an electrical current that is directly proportional to the temperature. This current is easily converted to a voltage proportional to the temperature by measuring the voltage drop across a resistor.

Since the output signals from the thermocouples are in the millivolt range, these signals have to be amplified such that sufficient resolution exists during digitization. A set of eight voltage amplifiers is used to amplify the thermocouple signals. An amplification factor of 200 is used for all the signals. The amplifiers contain built-in electronic low pass filters which are used to eliminate undesirable high frequency components. External high pass filters are also used while recording the fluctuating components of the signals to check for noise type degradations.

An analog to digital conversion board is installed in the microcomputer to allow the digitization of the conditioned signals. A maximum of 16 signals may be digitized simultaneously using this facility. The frequency and the resolution of the analog to digital conversion may be specified by the user. The recorded data are stored on diskettes after digitization. An algorithm that controls the digitization process has been developed and is discussed in Sec. 4.2.

Different types of failures were introduced in the signals during data acquisition. Bias failures can be obtained by removing one or more of the thermocouples from the piping. Excessive noise levels can be obtained by removing the grounding of one or more of the amplifiers, or by recording signals without proper low-pass filtering.

## 4.4 Experimental Result

This section contains a discussion of the application of the signal validation system to several failure situations. The sensor failure types discussed include:

- bias degradation at steady state condition,
- bias degradation during transient condition,
- noise degradation,
- a simultaneous bias and noise degradation, and
- a repeated pulse type failure.

Each case discussion contains a description of the experimental arrangement, a description of the failure, a discussion of the input to the algorithm, and a discussion of all the output variables generated by the algorithm. Plots of the signal estimate and the log likelihood ratio as functions of time, are provided.

### 4.4.1 Steady State Bias Failure

The bias failure of one sensor at steady state conditions is the simplest type of failure considered. To obtain the experimental data the air loop was operated at full heater power and air flow conditions. Three K-type thermocouples were used to measure the temperature of the air leaving the heater. To simulate a bias drift in one of the signals, the second thermocouple was gradually removed from the air loop. This caused a decrease in the temperature as indicated by the second thermocouple. The degradation in the second thermocouple is introduced after about 20 seconds. Until about 20 seconds

all three temperatures agree closely. After about 20 seconds the first and third thermocouples continue to be consistent with respect to each other. The temperatures as measured by all the thermocouples are presented in Fig. 4.4.1.

The tolerance for the consistency check is specified to be b = ± 2 °F for each sensor. For quicker detection of the failure, a smaller error bound may be used. The magnitude selected allows a clear understanding of the signal validation procedure during the detection and isolation of a failure. The corresponding tolerance for the SPRT is taken to be $\Theta_1$ = 4 °F. In general, the bound $\Theta_1$ is taken to be twice the bound, b. This selection is made to ensure that the GCC and the SPRT are consistent with respect to each other. The normal degraded variances as required by the SPRT are specified to be

$$\sigma_o^2 = 4\ °F^2 \text{ and}$$

$$\sigma_1^2 = 16\ °F^2\ .$$

For the calculation of the maximum accepted value of the log likelihood ratio, a mean time between false alarms of 10,000 time steps is used. This corresponds to a false alarm probability of approximately 0.1% and results in the maximum value of the log likelihood ratio of 9.90. The details of the program input are described in Appendix A.

The behavior of the Log Likelihood Ratio (LLR) for each signal is now discussed. The ratio as computed for the second thermocouple to check for bias degradations is discussed first. The reader is referred to Fig. 4.4.2 for this discussion. The LLR remains at -2 from time zero until about 20 seconds. As the second thermocouple

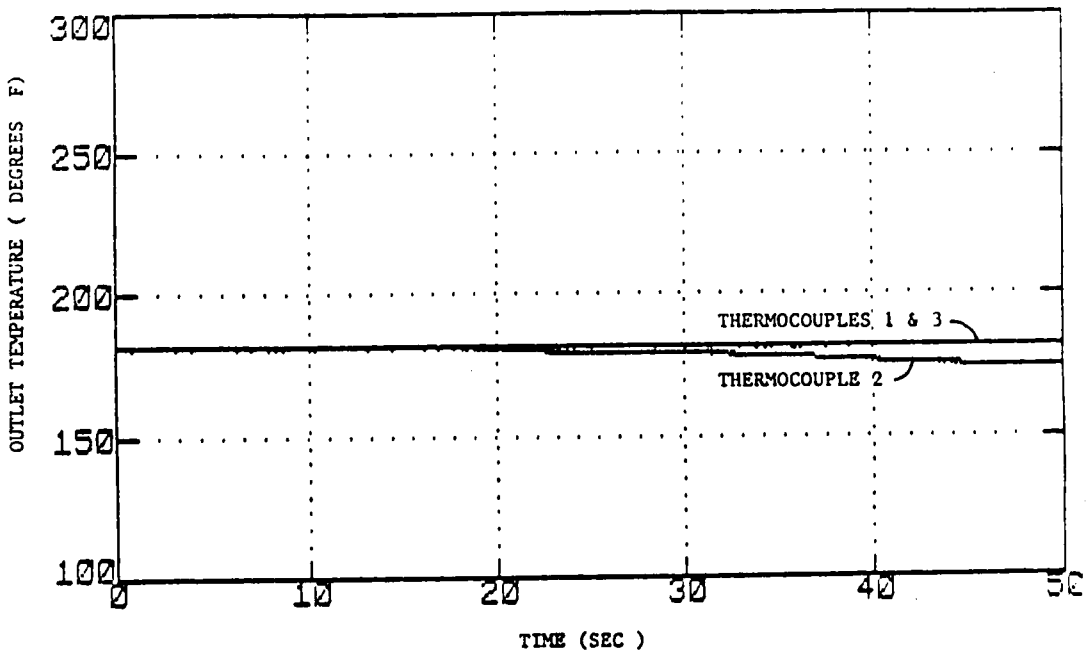



Figure 4.4.1 Bias Degradation in Second Thermocouple at Steady State Conditions.

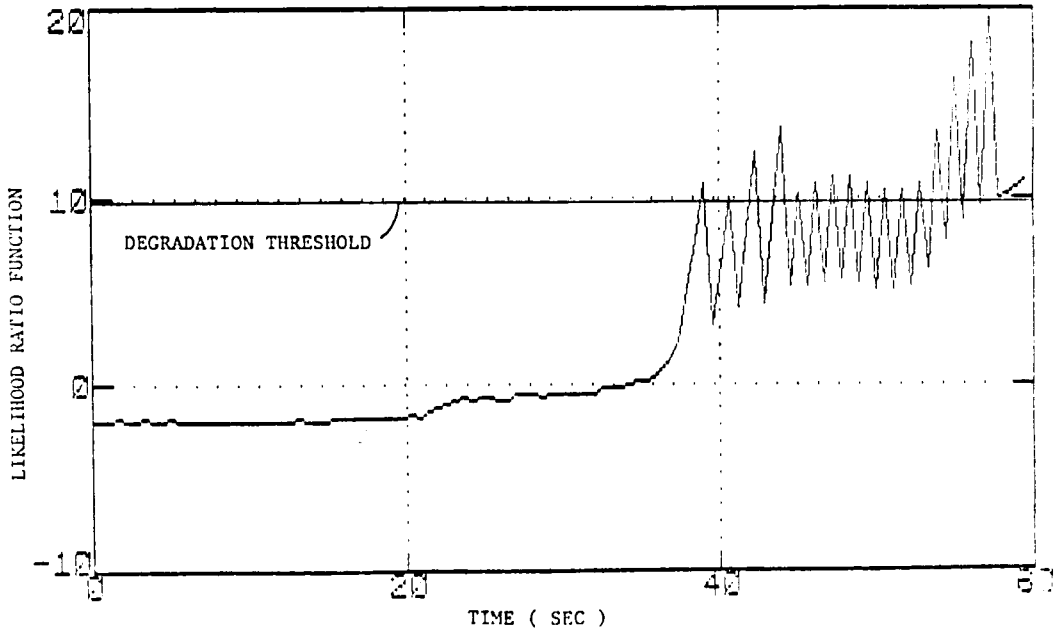



Figure 4.4.2 The LLR for the Second Thermocouple ( B1= 9.9 ).

starts to degrade at about 20 seconds, the LLR starts to increase in the positive direction. Recall that an increase in the LLR indicates a tendency for the sensor to be in the degraded condition. The LLR continues to increase and becomes positive at about 35 seconds. After 38.5 seconds (771 samples) the LLR exceeds the calculated maximum value of 9.90, and the second thermocouple is identified to have undergone a bias degradation. Notice that up to 38.5 seconds the magnitude of the degradation is less than the value of $\Theta_1$ = 4 °F, and therefore the measurement is acceptable. After this point the failure magnitude exceeds 4 °F and the LLR increases at an accelerated rate. Also at 38.5 seconds the GCC isolates the second thermocouple as inconsistent and it is no longer used in the computation of the signal estimate. In other words, the second signal has zero weighing in the weighted average as obtained by the GCC procedure.

It must be pointed out that the LLR is plotted after the control action is applied. The LLR is set equal to zero whenever it is less than zero. The LLR is then recomputed for the next sample. These recomputed values are plotted at every sample. A similar situation exists after the LLR exceeds its maximum. Every time a failure is detected the LLR is reinitialized to zero and a new LLR is computed at the following sample. Again the recomputed values are plotted. This discussion is applicable to all of the LLR plots in this section.

At this point the estimate of the temperature as computed by the GCC is discussed. Up to about 20 seconds, all three of the measurements as well as the estimate are in close agreement with each other. The estimate is just an equally weighted average of all the three

signals. From 20 seconds up to 38.5 seconds no inconsistencies exist and the estimate continues to be an equally weighted average. The second thermocouple has started to degrade, and this degradation is reflected in the estimate. At 38.5 seconds the second thermocouple is isolated as inconsistent and the estimate is based on the two normal thermocouples only. The weighted average of the measurements is presented in Fig. 4.4.3. Notice that the estimate is affected by the degradation of the second thermocouple, but that the effect never exceeds the specified tolerance of 4 °F.

The LLRs as computed for the two normal thermocouples and presented in Figs. 4.4.4 and 4.4.5 behave very much alike. Up to about 20 seconds both ratios remain at about -2 indicating that both thermocouples are normal. After 20 seconds both LLRs start to increase in value. This is due to the effect of the degraded sensor on the estimate and the calculation of the LLR based on the measurement residual, $S_i$, containing the estimate. Refer to section 3.3 for a detailed description. At 38.5 seconds the degraded sensor is no longer used in the estimate and the LLRs of the normal sensors return to -2. Notice that neither of the two ratios reached the maximum of 9.90 and no false alarms are given.

This discussion illustrates the advantages of combining the GCC and the SPRT. The GCC provides the estimate of the signal that can be used in the SPRT to achieve the optimum detection time.

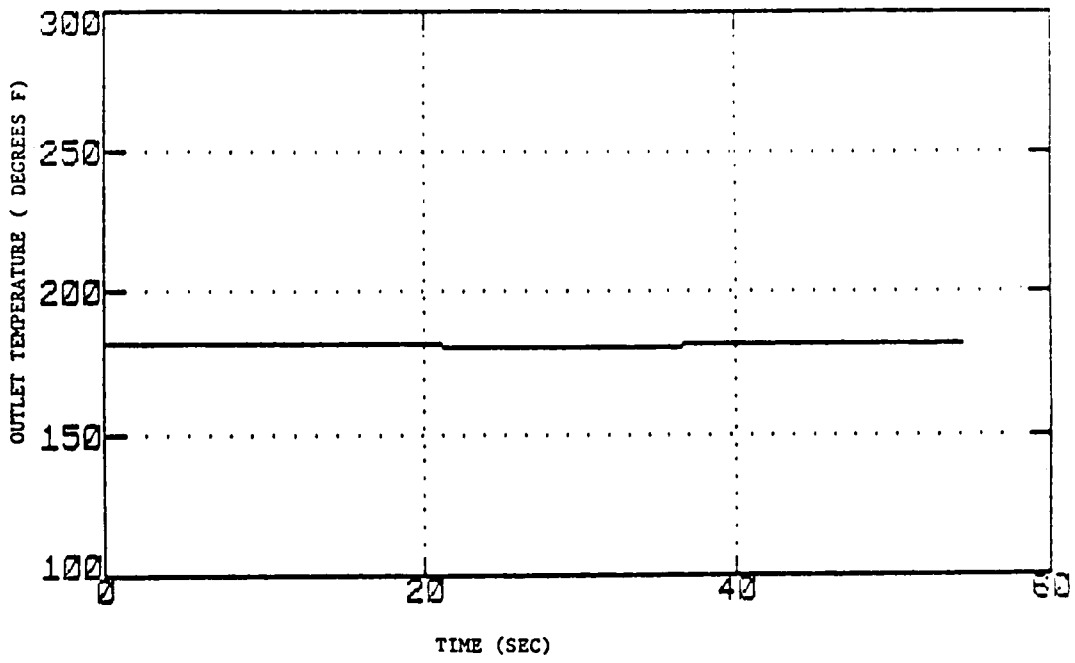



Figure 4.4.3 The Weighted Average of the Three Thermocouples.

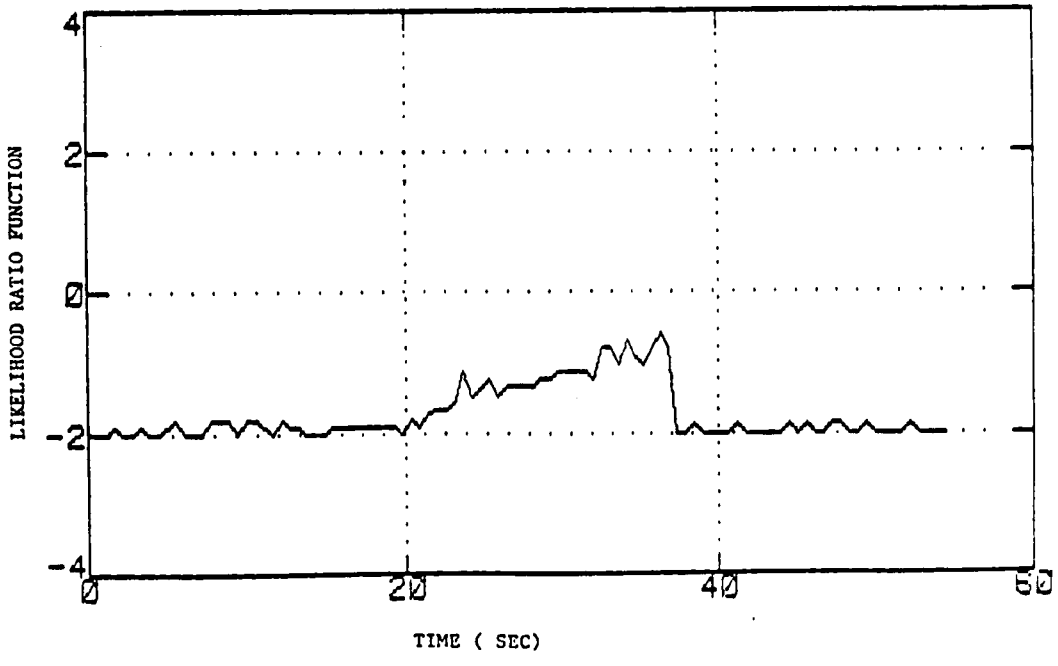



Figure 4.4.4 The LLR for the First Thermocouple ( B1= 9.9 ).

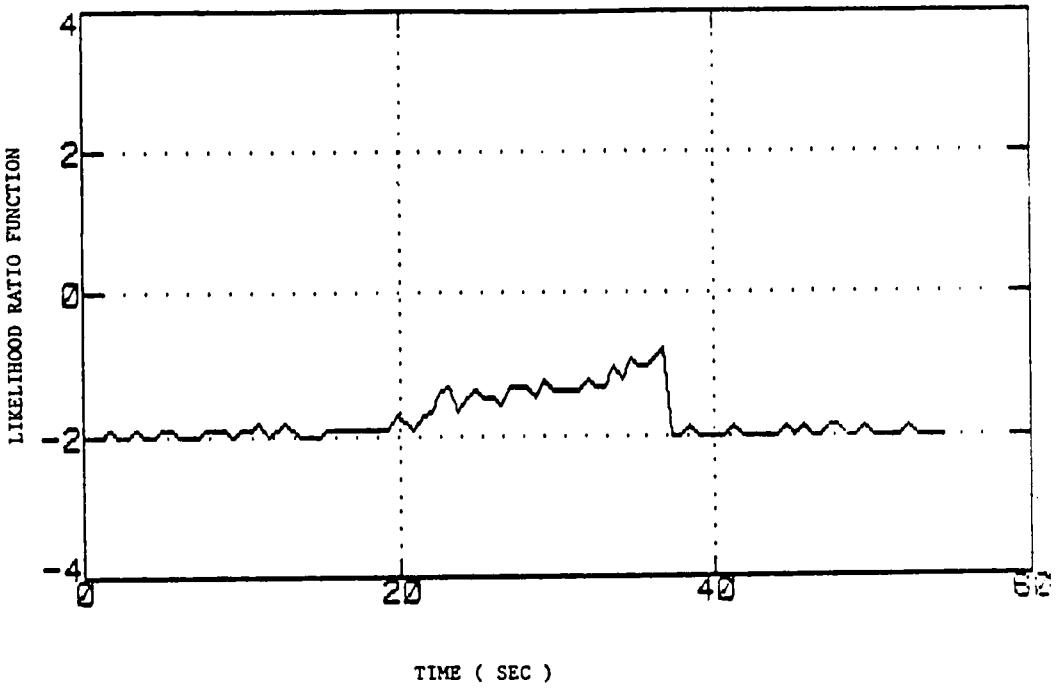



Figure 4.4.5 The LLR for the Third Thermocouple ( B1=9.9 ).

### 4.4.2 Bias Degradation During a Transient

The second case considered is the application of the signal validation system to sensors measuring a changing process variable. It is important to check the applicability of the signal validation system to transient situations since this is one of the main motivations for combining the GCC and the SPRT. To obtain transient experimental data the air loop is operated at full fan power and zero heater power at room temperature. The heaters are then turned on to full power and the outputs of three thermocouples measuring the outlet temperature of the heater section are recorded. After 1100 seconds the second thermocouple is partially removed from the air flow. The thermocouple is reinserted after 1380 seconds. This allows the investigation of the behavior of the LLRs in the case where a sensor only partially degrades and then returns to the normal state. If a smaller maximum allowable failure magnitude is selected the degradation may be considered as complete instead of just partial. The behavior of the LLRs in this situation may be investigated.

To check the behavior of the LLRs under partial degradation conditions relatively large error tolerances are selected. The GCC tolerance is specified to be ±4 °F. The corresponding $\Theta_1$ for the SPRT is set at 8 °F with $\sigma_0^2 = 16\ °F^2$ and $\sigma_1^2 = 64\ °F^2$. The mean time between false alarms is specified as 10,000 samples with a time step of 0.5 second. The maximum allowable value of the LLR is calculated to be 9.90. The maximum degradation in the second thermocouple does not exceed the specified 8 °F tolerance. The temperatures as indicated by all three thermocouples are shown in Fig.

4.4.6. The LLR for the second thermocouple calculated for bias is shown in Fig. 4.4.7. Up to about 1100 seconds the LLR remains negative between -1 and -2. Notice that the LLR is less smooth as a function of time compared to the case described in Sect. 4.4.1, reflecting the noisy nature of the original signals. After 1100 seconds the LLR starts to increase and reaches a maximum of 0.9 at 1400 seconds. After this point the ratio decreases reflecting the reinsertion of the thermocouple into the air loop. The LLR does not exceed the maximum value of 9.90 since the failure never exceeds the specified $\theta_1$ of 8°F.

The estimate of the signal as computed by the GCC is presented in Fig. 4.4.8. The degradation in the second sensor does have a limited effect on the estimate. The magnitude of the influence is small compared to the magnitude of the original degradation. This is due to the reduced weighting of the second signal by the GCC. It is important to realize that the weighting procedure continues even when no degradations are detected. In the case where a bias degradation is detected the weighting is zero, while in partial degradations the weighting varies between zero and unity ($o < w < 1$).

The LLRs for the two normal thermocouples are presented in Figs. 4.4.9 and 4.4.10. Both ratios remain between -1 and -2. At 1100 seconds the degradation of the second thermocouple starts to influence the weighted average, which in turn causes the LLRs for the two normal sensors to increase. This increase is much smaller than the increase in the ratio for the second sensor and remains much smaller than the upper limit of 9.90. After the second sensor returns to the normal mode the ratios for the two normal sensors decrease to the original

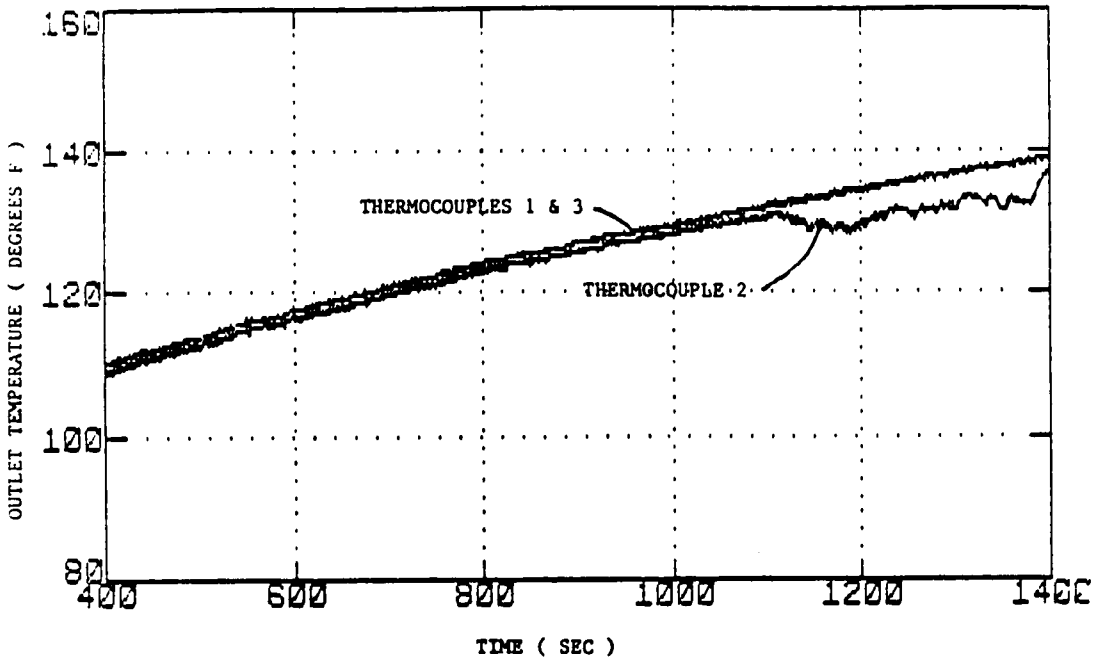



Figure 4.4.6 Bias Degradation in Second Thermocouple at Transient Conditions.

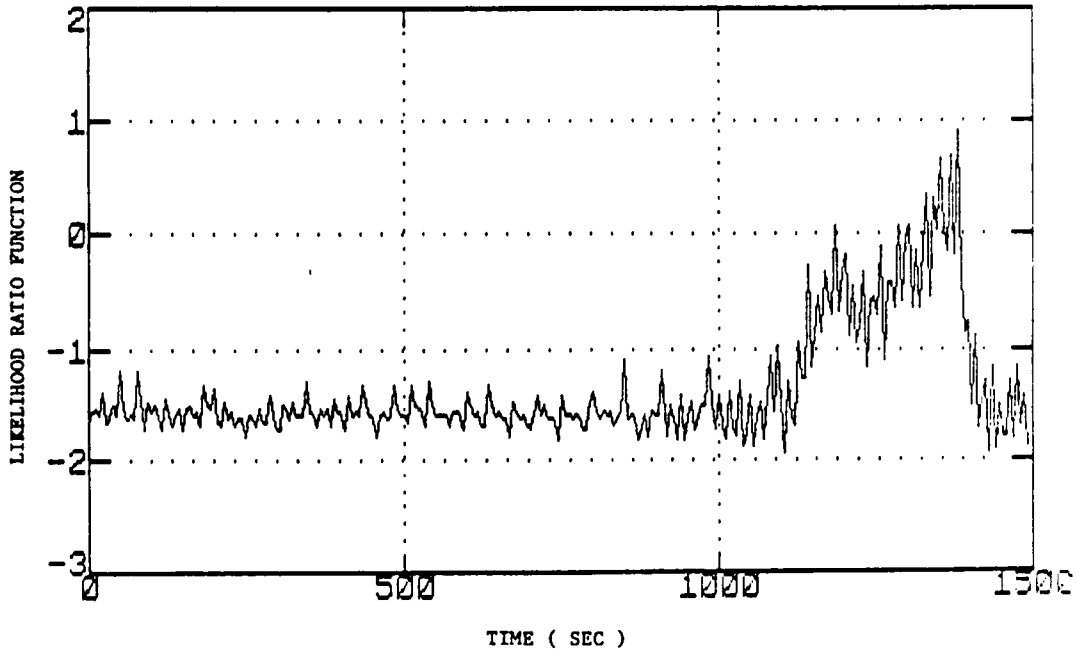



Figure 4.4.7 The LLR for the Second Thermocouple ( B1=9.9 ).

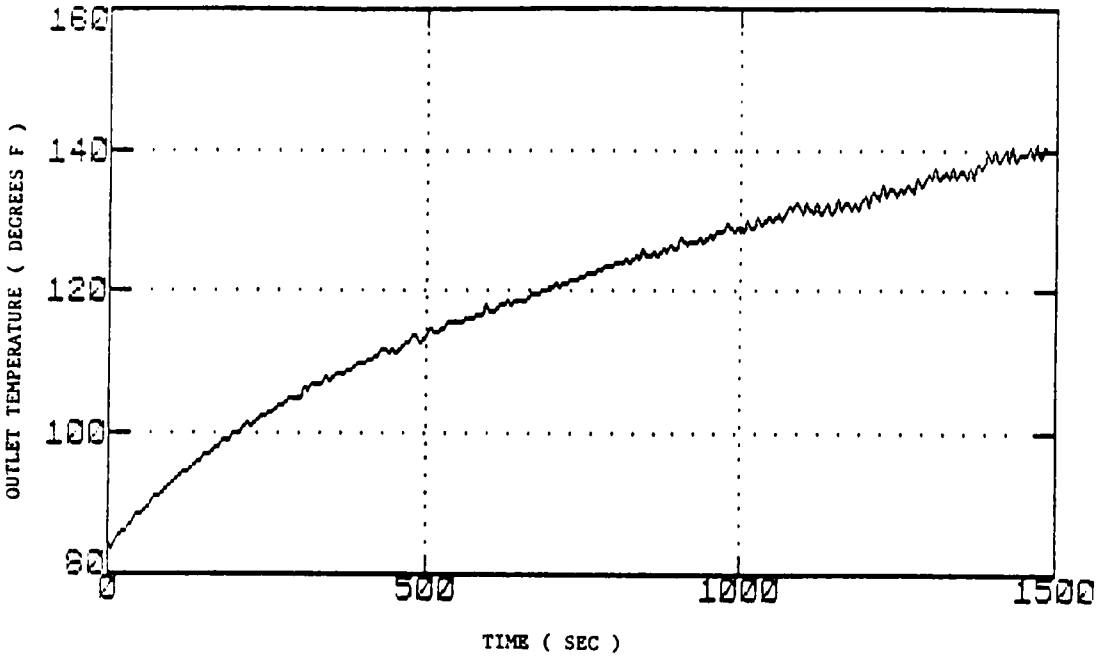



Figure 4.4.8 The Weighted Average of the Three Thermocouples.

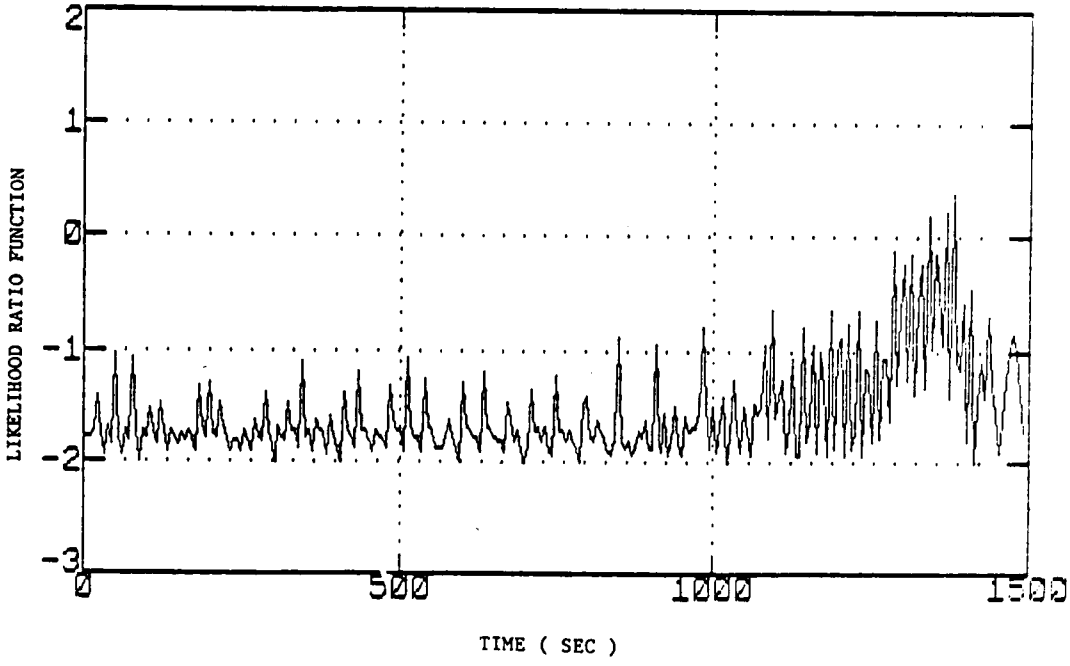



Figure 4.4.9 The LLR for the First Thermocouple ( B1=9.9 ).

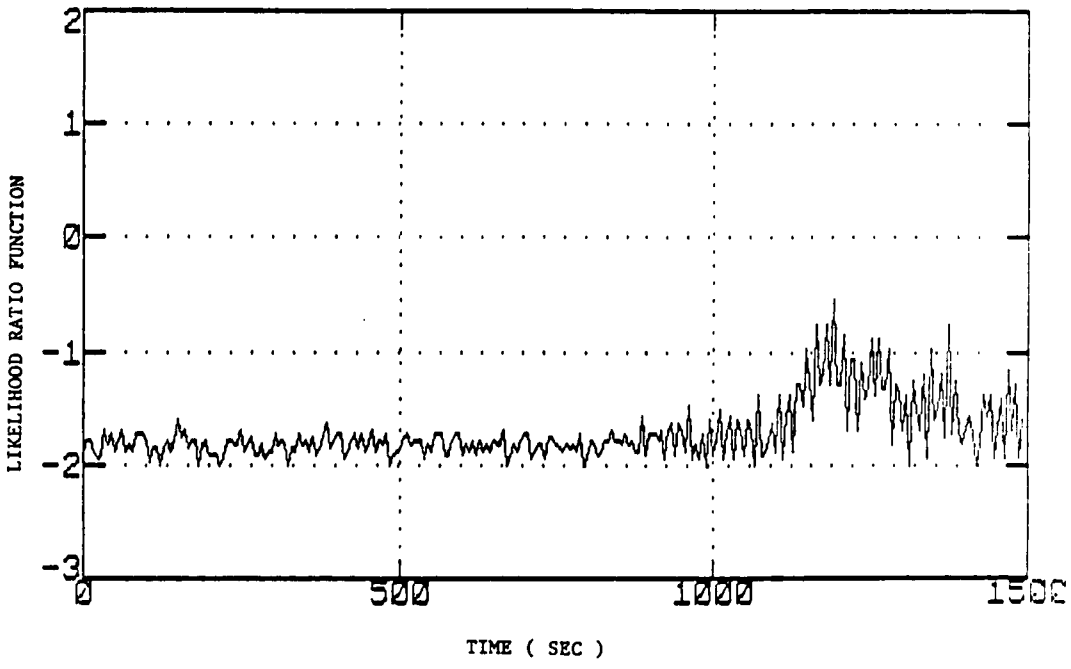



Figure 4.4.10 The LLR for the Third Thermocouple ( B1=9.9 ).

levels. If the GCC did not compute a weighted average the LLRs for the normal sensors would be affected to the same extent as the LLR for the degraded sensor.

The same data are next used to check the effect of a complete bias degradation on the performance of the signal validation system. To accomplish this a smaller maximum acceptable degradation is specified. The specified inputs are: ±2 °F for the bound b, 4 °F for $\Theta_1$, 4 $°F^2$ for $\sigma_o^2$, and 16 $°F^2$ for $\sigma_1^2$.

The LLRs for the three sensors are presented in Figs. 4.4.11, 4.4.12, and 4.4.13. The LLR for the second sensor exceeds the maximum value of 9.90 and the bias degradation of the second thermocouple is detected. All LLRs return to about -1.5 after the second thermocouple returns to normal. The estimate of the signal as computed by the GCC is presented in Fig. 4.4.14.

### 4.4.3 Noise Degradation

One of the important features of this study is the extention of the SPRT to the detection of unacceptable signal variance levels. To simulate high noise levels the air loop is operated at full heater and fan power. Large electronic noise levels exist in the signals and are filtered out during the bias degradation tests. In the noise detection test the filtering is applied to the first and third outlet thermocouples only. The second thermocouple signal is left unfiltered and contains large noise levels. Since the noise is periodic in nature, this test also illustrates the applicability of the signal validation system to signals that have nondecreasing autocorrelations. To check

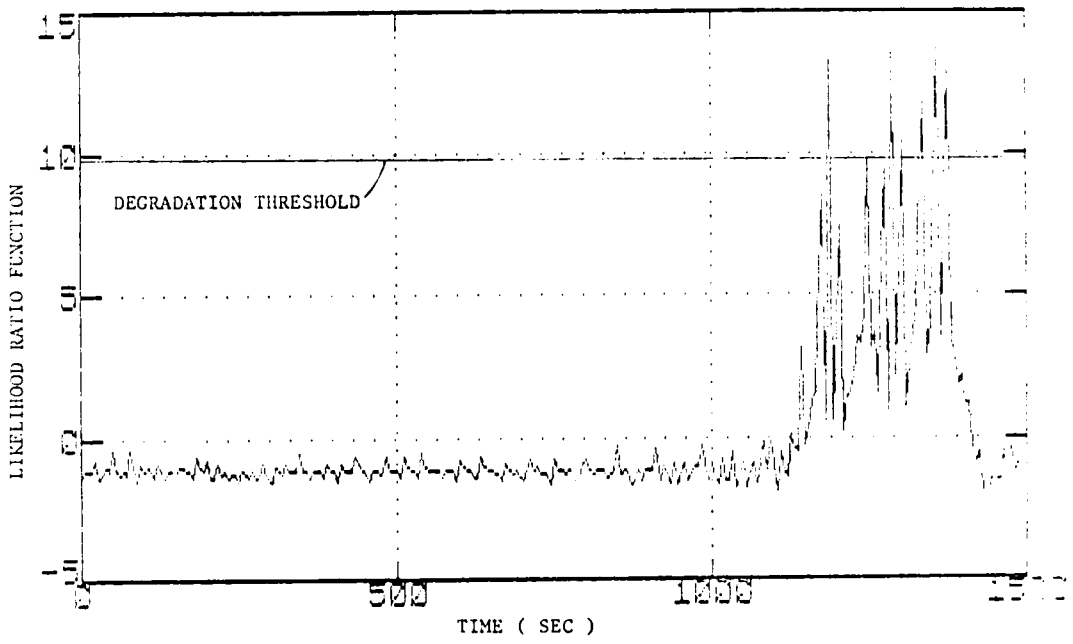



Figure 4.4.11 The LLR for the Second Thermocouple ( B1=9.9 ).

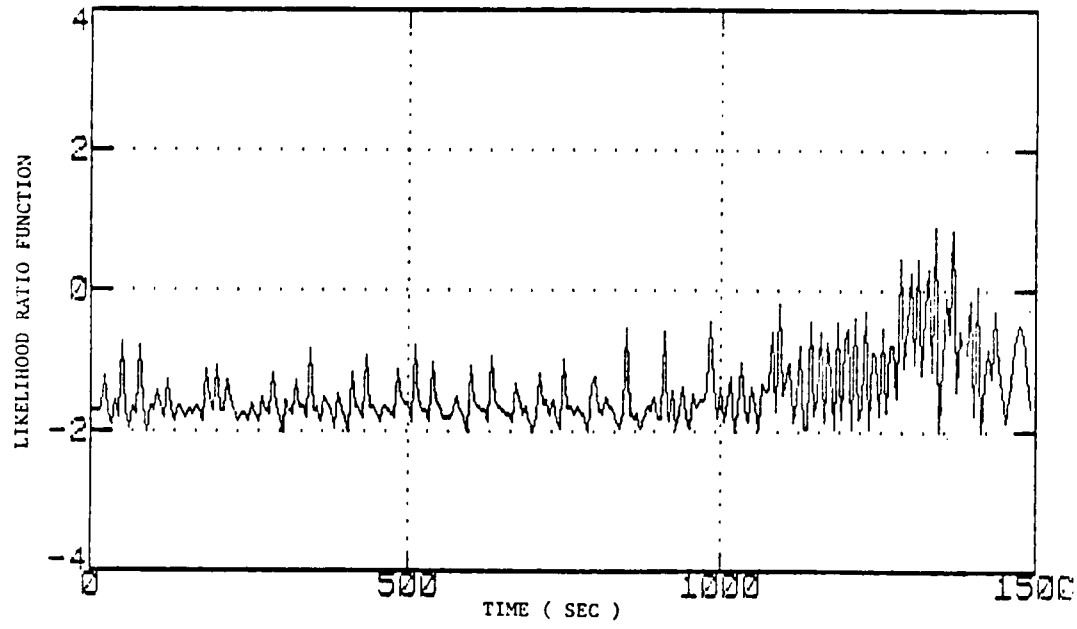



Figure 4.4.12 The LLR for the First Thermocouple ( B1=9.9 ).

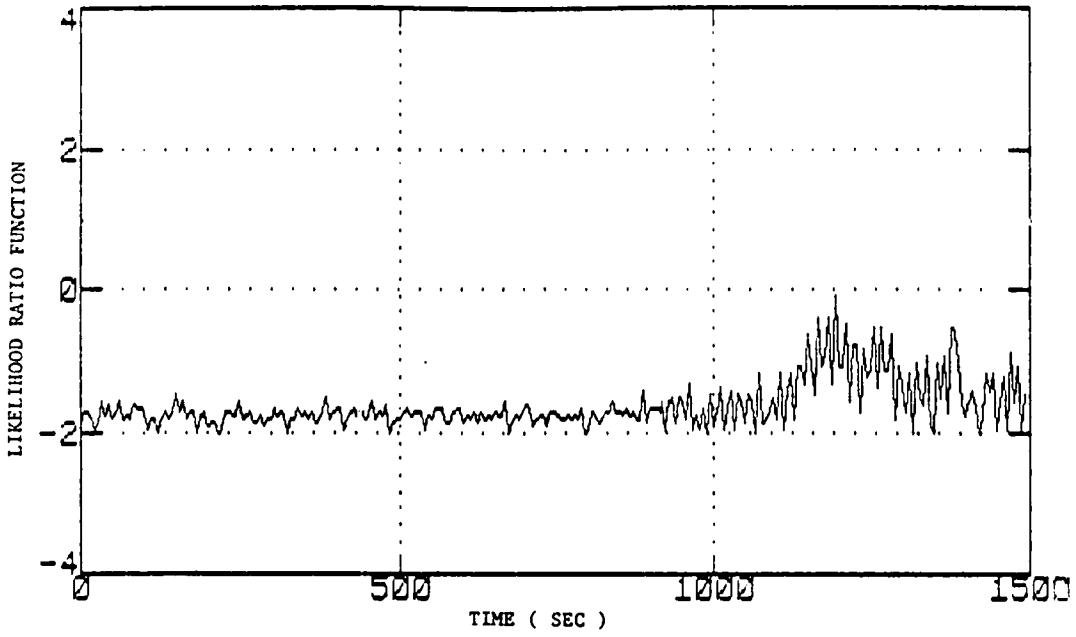



Figure 4.4.13 The LLR for the Third Thermocouple ( B1=9.9 ).

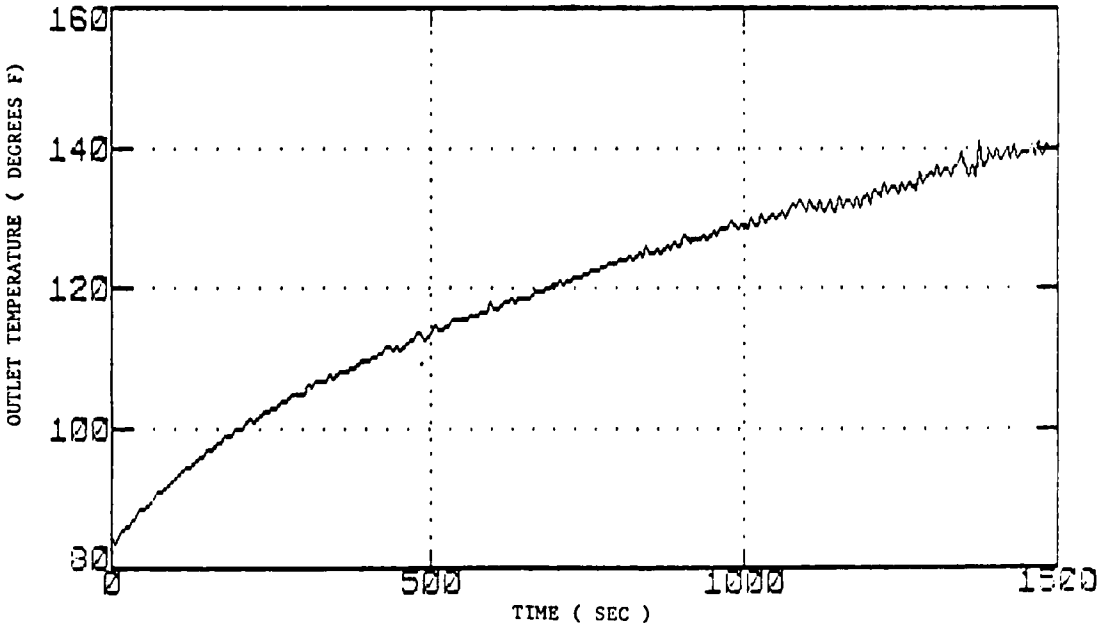



Figure 4.4.14 The Weighted Average of the Three Thermocouples.

for noise degradation the low frequency components of the signals are filtered using a high-pass filter and the SPRT is applied to the remaining fluctuating components. The excessive noise content of the second thermocouple is illustrated in Fig. 4.4.15.

In this case the maximum bias limits are specified such that no bias degradations can be detected. Excessive noise levels therefore exist, while the fluctuations do not exceed the maximum mean degradation limits. The input variables are specified as follows: for the GCC, $b = \pm 2$ °F, for the SPRT, $\Theta_1 = 4$ °F, $\sigma_0^2 = 1$ °F$^2$, and $\sigma_1^2 = 4$ °F$^2$. Since the SPRT is applied to noise signals the missed and false alarm probabilities are specified as: $\alpha = 0.001$ and $\beta = 0.001$. The corresponding mean time between false alarms of 10,000 time steps is used.

The estimate of the signal as determined by the GCC is provided in Fig. 4.4.16. In this case no bias degradations are detected, and the estimate is therefore just the average of the three signals. The LLRs as computed for bias degradation for the three signals are presented in Figs. 4.4.17, 4.4.18, and 4.4.19. At no sample do any of the three LLRs exceed the maximum value of 11.29 and no bias failures are detected. The effect of the noisy second signal is observable only to a limited extent.

The LLRs computed for noise degradation strongly respond to the increase in the variance. The LLR for the second signal as computed for the noise degradation is presented in Fig. 4.4.20. The LLR is clearly very sensitive to the added noise and increases relatively fast. The bound of 6.9 is exceeded rapidly. The periodic nature of

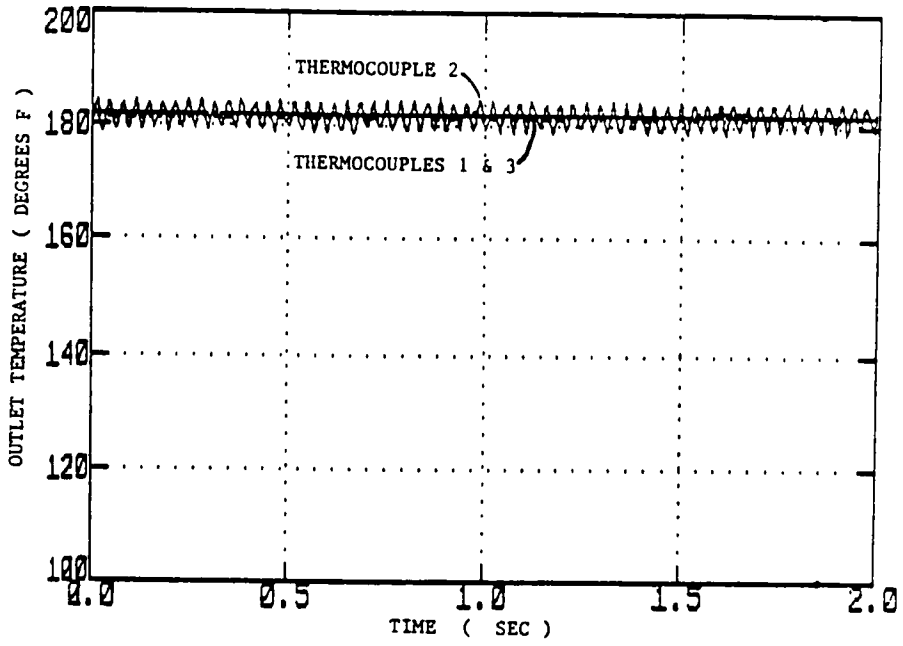



Figure 4.4.15 Periodic Noise Degradation in the Second Thermocouple.

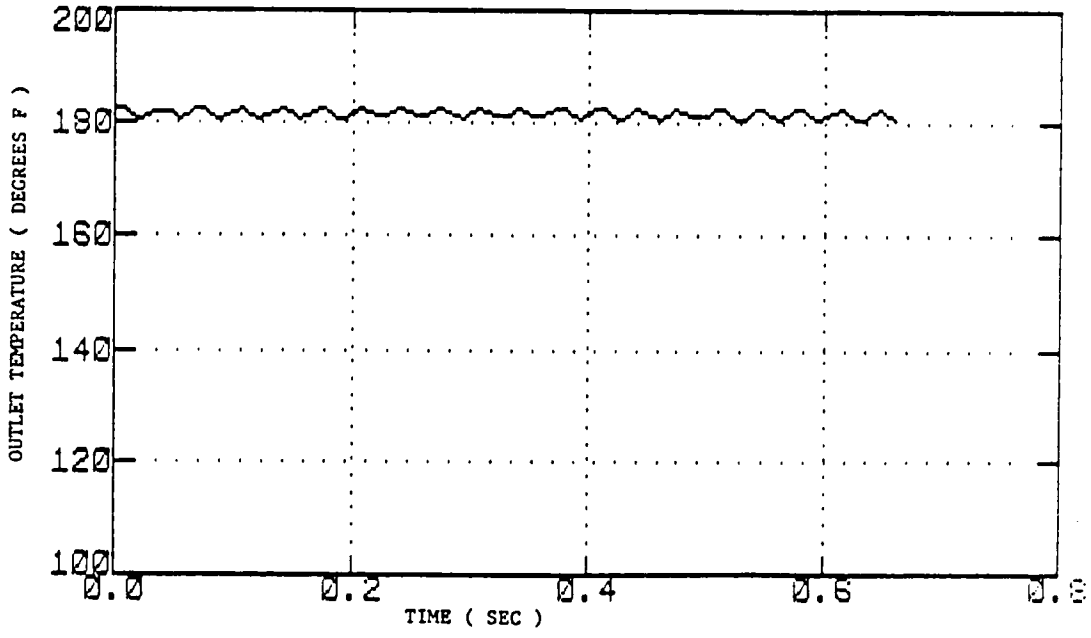



Figure 4.4.16 The Weighted Average of the Three Thermocouples.

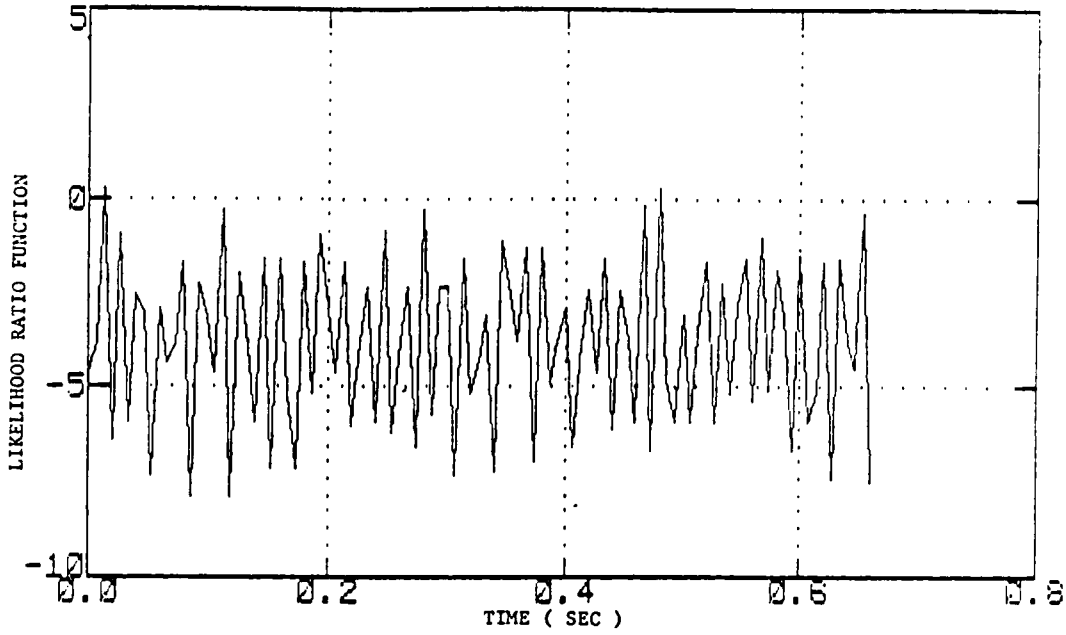



Figure 4.4.17 Bias LLR for the Second Thermocouple ( B1=11.3 ).

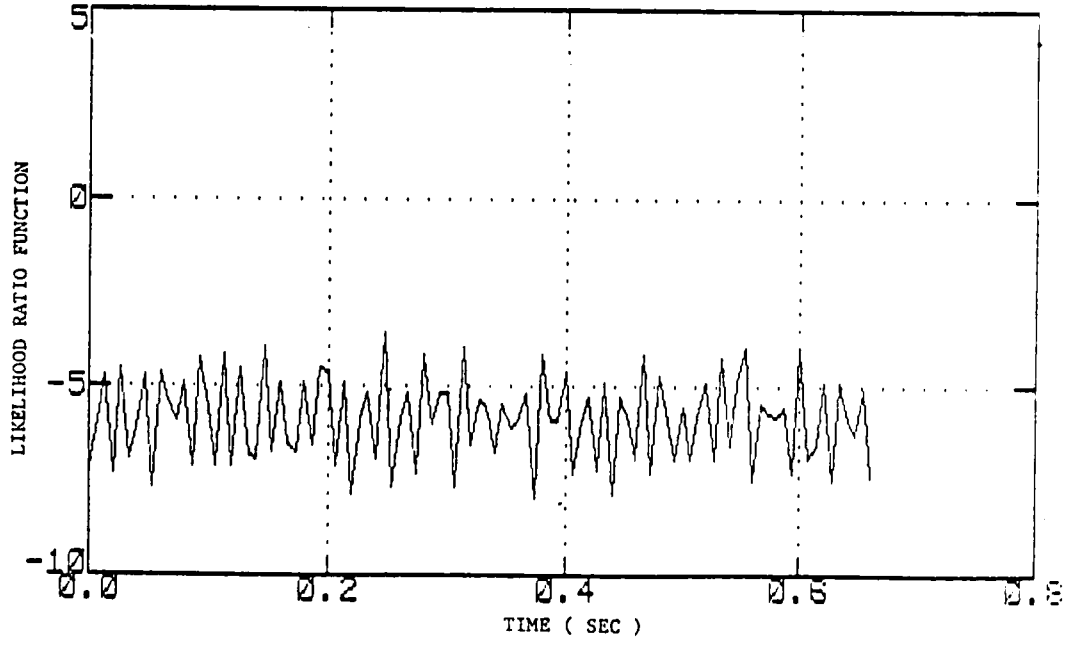



Figure 4.4.18 Bias LLR for the First Thermocouple ( B1=11.3 ).

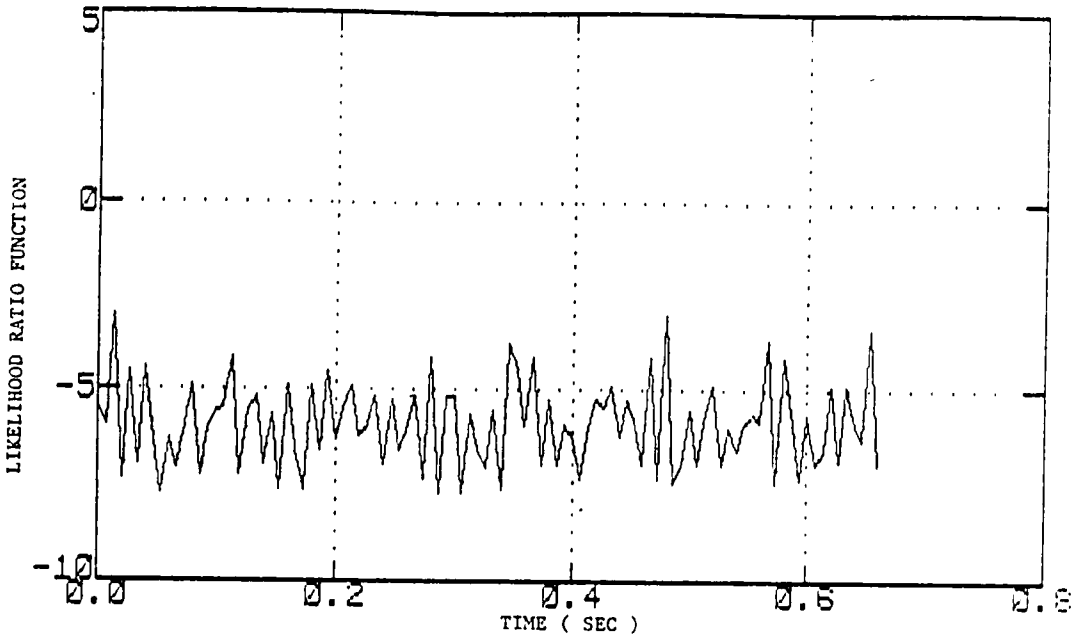



Figure 4.4.19 Bias LLR for the Third Thermocouple ( B1=11.3 ).

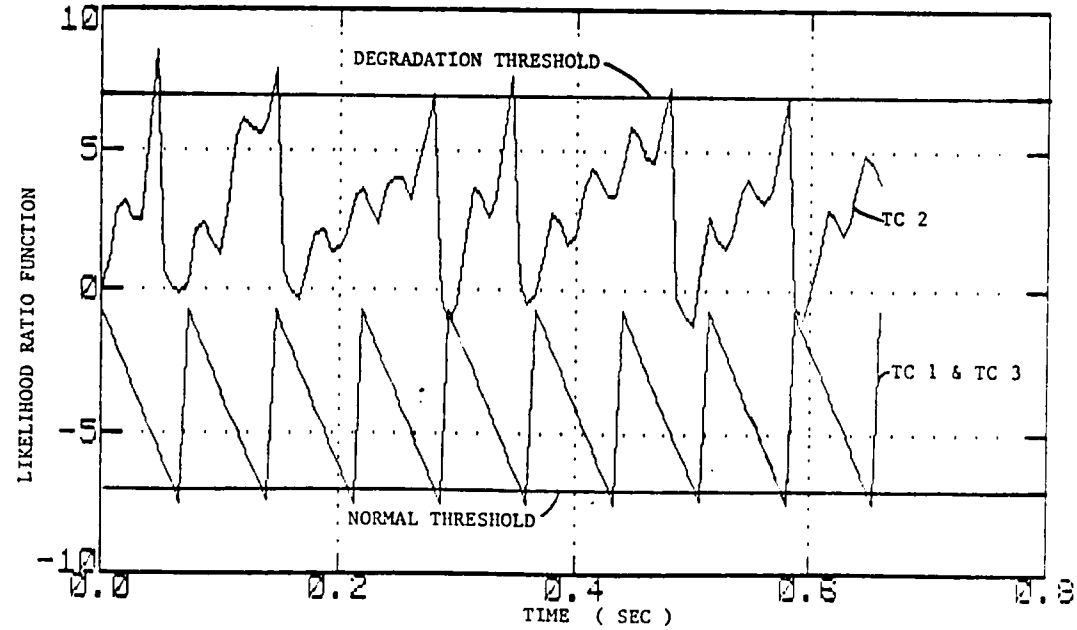



Figure 4.4.20 Variance LLR for all Three Thermocouples ( A=-6.9, B=6.9 ).

the degradation is also observable in the LLR. The maximum rate of increase in the LLR occurs at the maximum deviations in the signal from the mean. Also presented in Fig. 4.4.20 are the LLRs for the two normal signals. These LLRs are negative and exceed the bound -6.9, indicating normal noise levels for thermocouples one and three. Note that unlike for the bias degradation case, the LLRs for the noise case decrease until the bound A is exceeded. This is because the unmodified SPRT is used and no control action is used in the SPRT as applied to noise degradations.

This test illustrates that the signal validation system is capable of detecting noise degradations when no bias degradations exist. The correct sensor is isolated and no false alarms are given.

So far the signal validation system has been applied to a mean degradation with no variance failures for both a steady state and a transient case, and to a steady state case containing a variance degradation only. The next step is to check the applicability of the signal validation system to a signal containing both a bias and a noise degradation.

### 4.4.4 Simultaneous Bias and Noise Degradation

To determine the ability of the signal validation system to detect and distinguish between bias and noise degradations occurring simultaneously, a computer simulation is used to generate three measurements. A random number generator is used to introduce excessive variance levels in the second signal after 100 seconds. At the same time a drift in the mean of the second signal is introduced. The rate at

which the degradation occurs is 0.1 °F/second. The standard deviation of the normal signals is 0.1 °F, while the standard deviation of the degraded signal is 2.0 °F. The three signals are depicted in Fig. 4.4.21. The fluctuating component of a normal signal is presented in Fig. 4.4.22, and the fluctuating component of the signal containing the noise degradation is illustrated in Fig. 4.4.23. Figures 4.4.21 and 4.4.23 clearly illustrate the drift in the mean value and the increase in the noise variance.

The input variables to the algorithm are specified as follows:

- time step = 1.0 seconds,
- $\alpha$ = 0.001,
- $\beta$ = 0.001,
- MTIME = 10,000,
- b = 2 °F,
- $\Theta_1$ = 4 °F,
- $\sigma_0^2$ = 1 °F$^2$ ,
- $\sigma_1^2$ = 4 °F$^2$ .

The maximum allowable value for the bias degradation LLR is 11.29, while the bounds for the noise degradation check are ± 6.91.

The bias degradation is detected and isolated 13 seconds after the mean drift is initiated as indicated by the LLR for mean degradation in Fig. 4.4.24. At this point the deviation due to the bias drift has not yet reached the 4 °F limit as specified by $\Theta_1$, but the signal does contain a superimposed variance of 4 °F$^2$ and this causes the deviation to exceed the limit. The LLR remains at -2 until 100 seconds when the

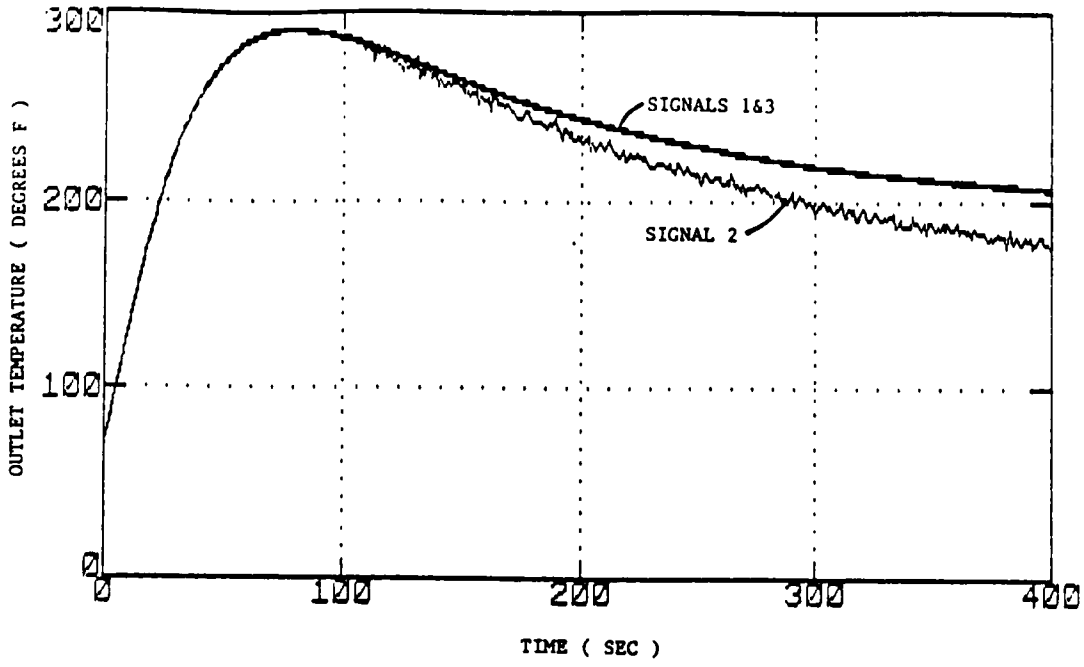



Figure 4.4.21 Bias and Noise Degradations in Second Signal.

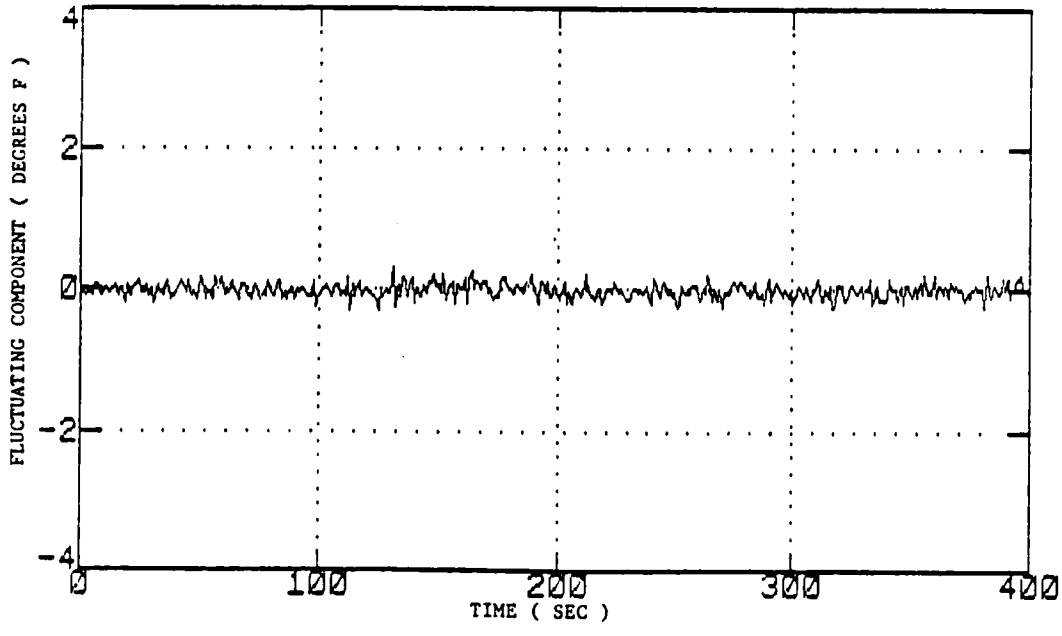



Figure 4.4.22 Normal Noise Components of Signals.

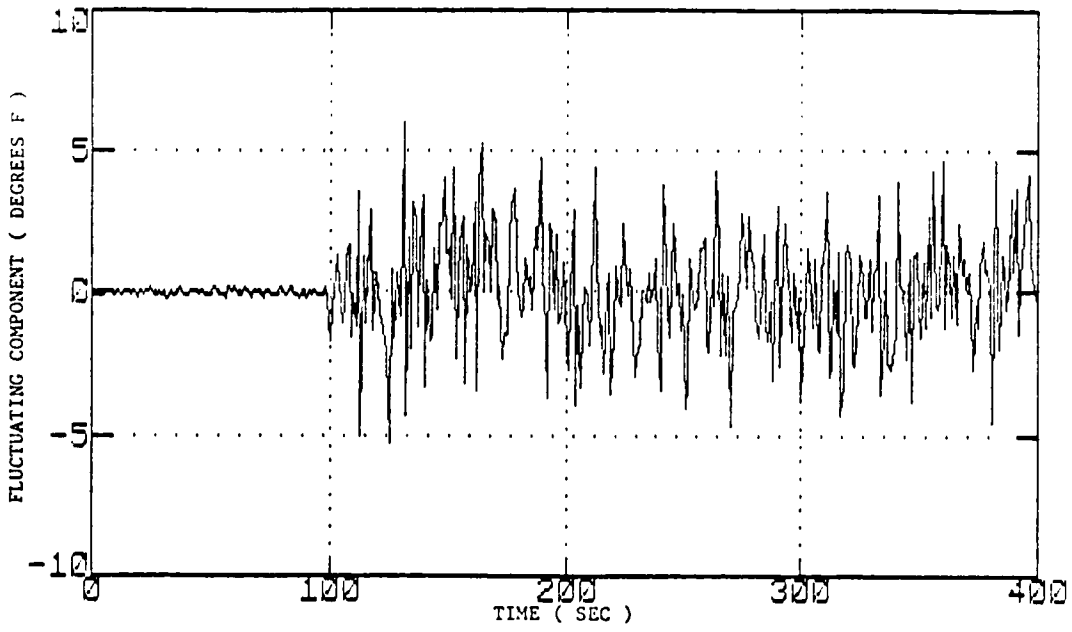



Figure 4.4.23 Degraded Noise Component in signal 2.

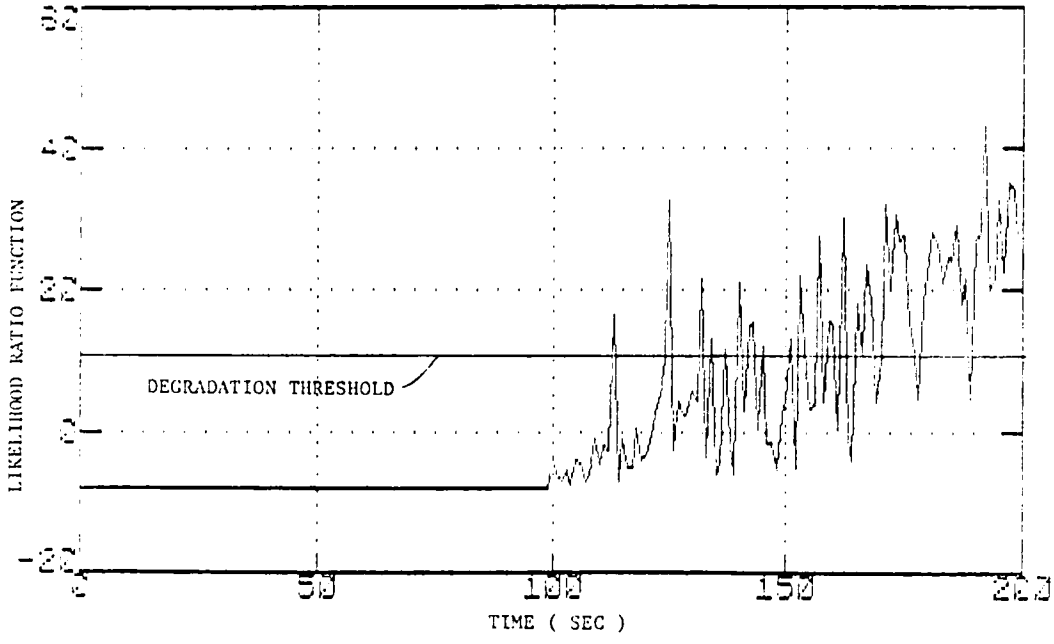



Figure 4.4.24 Bias LLR for the Second Signal ( Bl=11.3 ).

sensor starts to degrade. After this point the LLR increases rapidly. The increased noise level in the second signal is also clearly observable in the LLR as compared to the case depicted in Fig. 4.4.2 (p. 60) where the degradation is of the bias type only.

The LLRs as computed for the bias degradation checking of the first and third signals are depicted in Figs. 4.4.25 and 4.4.26. These LLRs show the effect of the second signal on the estimate as it starts to degrade. The LLRs remain negative and no false alarms are given. After 60 seconds the second signal is inconsistent with the first and the third, except for a few samples due to the noisy nature of the second signal. After about 80 seconds the degradation in the second signal is large enough so that it is continuously weighted with zero in the estimate of the signal, and the LLRs for the first and third signals return to normal.

The variance degradation is detected 13 seconds after the degradation occurs. At this point the LLR for the degraded signal is reinitialized. After about another 13 seconds the failure is again detected. The LLRs as computed for the noise degradation detection are depicted in Fig. 4.4.27. The repeated detection of the excessive noise levels of the second signal is clearly observable. Also notice that the LLRs for the two normal signals repeatedly exceed the bound -6.91 indicating the normal mode for both sensors.

Figure 4.4.28 represents the weighted average of the three signals as computed by the GCC. Also plotted in Fig. 4.4.28 are signals one and two. It is clear that the average agrees closely with the normal first signal. This example illustrates the ability of the signal

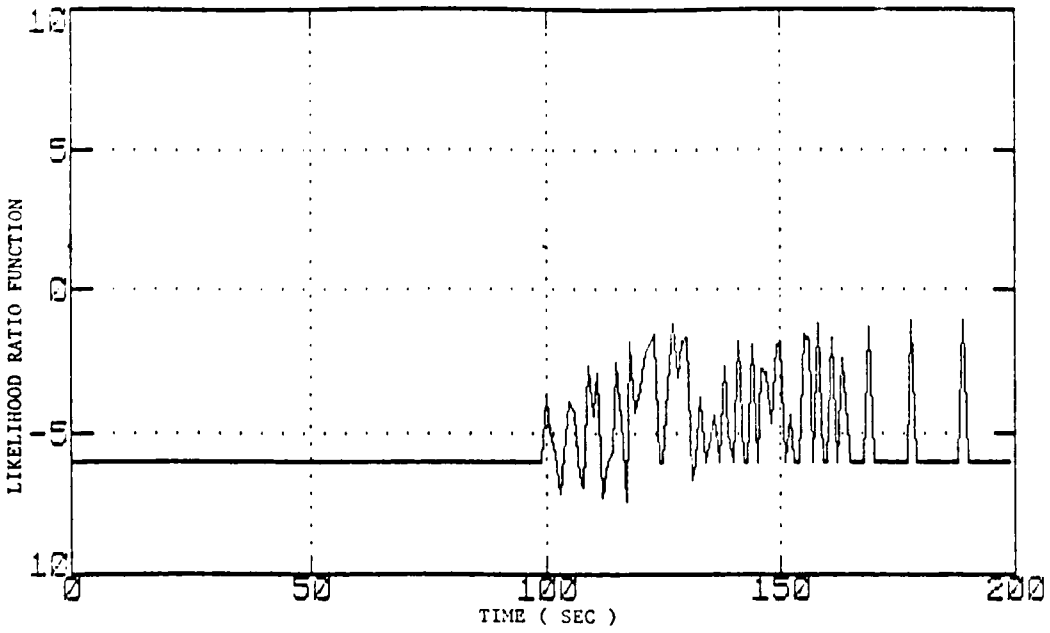



Figure 4.4.25 Bias LLR for the First Signal ( B1=11.3 ).

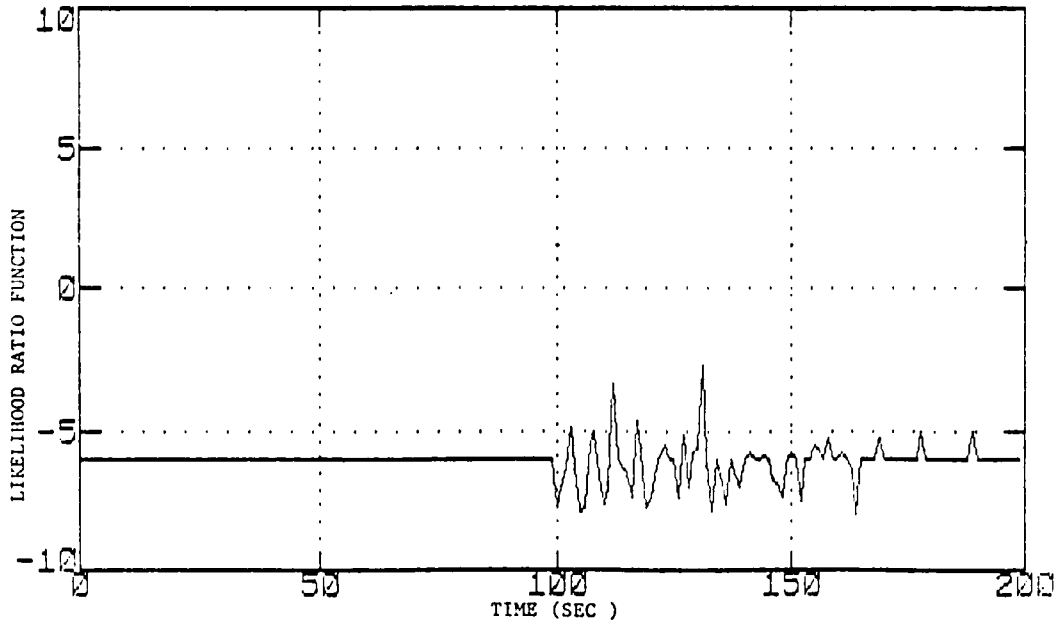



Figure 4.4.26 Bias LLR for the Third Signal ( B1=11.3 ).

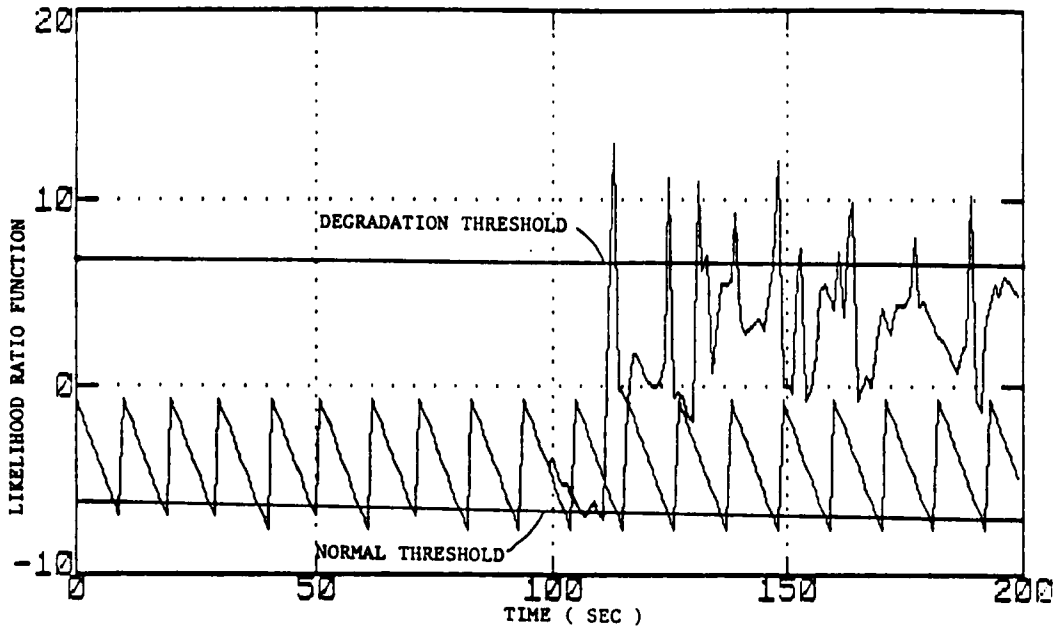



Figure 4.4.27 Noise LLR s for all Three Signals ( A=-6.91, B=6.91).

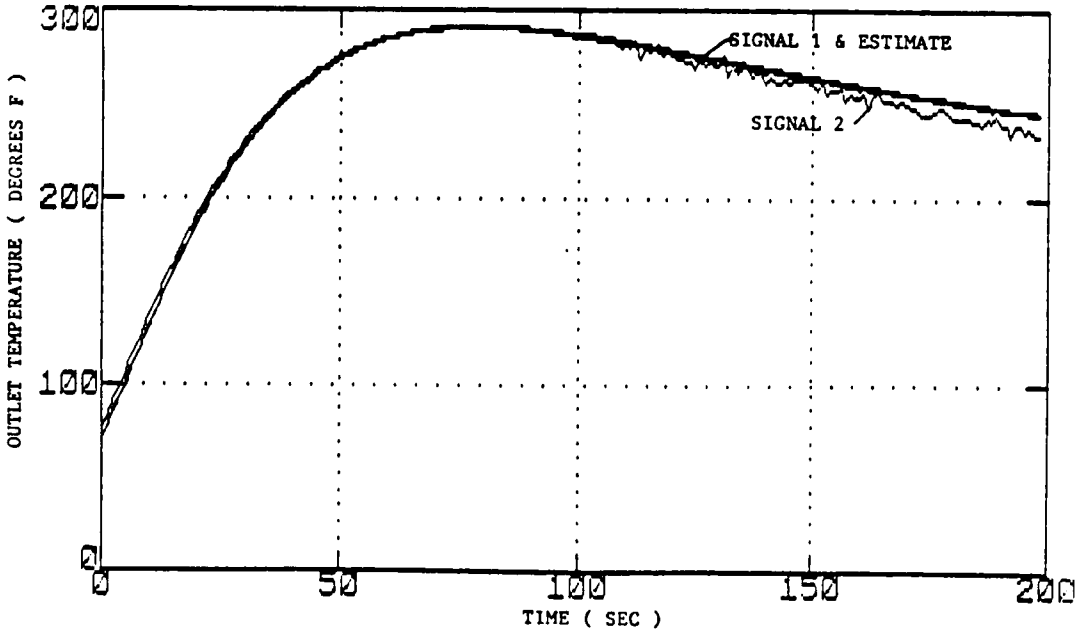



Figure 4.4.28 Comparison of Normal Signal 1, Degraded Signal 2, and Estimate.

validation system to detect and isolate noise and and bias failures. The next type of failure investigated consists of repeated bias spikes introduced into one of the signals.

### 4.4.5 Repeated Pulse Type Failure

It is interesting to investigate the performance of the signal validation system in the case of pulses occurring in the signal output. The pulses may be separated by regular or irregular time intervals. Unlike the previous mean or variance degradations the signal output returns to normal after each pulse. The degradation is therefore not exactly of the bias type, nor is it exactly a noise degradation. The three outlet temperature sensors are used to provide two normal signals and one signal contaminated with recurring pulses. The contamination is introduced by grounding and ungrounding one of the amplifiers. The air loop is operated at steady state conditions while the signals are recorded. The pulses introduced vary from a few degrees to ± 10 °F. Since the pulses are relatively narrow, a small time step is used to make sure that they can be detected. The input variables to the algorithm are specified as follows:

- time step = 0.05 second,
- $\alpha$ = 0.001,
- $\beta$ = 0.001,
- MTIME = 10,000,
- b = 3 °F,
- $\Theta_1$ = 6 °F,
- $\sigma_o^2$ = 1 °F$^2$ , and
- $\sigma_1^2$ = 9 °F$^2$ .

The maximum value computed for the bias LLR is 12.10 and the bounds for the variance LLR are ± 6.91. The signal from the first thermocouple containing the pulses is shown in Fig. 4.4.29. Note that some of the pulses exceed the 6 °F bound while others do not.

The algorithm is applied and the pulses are detected both as bias type and noise type failures. This is true, since some of the pulses are large enough such that they represent bias failures, while the pulses are also part of the high frequency component of the signal and therefore exist in the separately recorded noise signal. The LLR as computed for the first signal is presented in Fig. 4.4.30. The LLR remains normal except at the samples containing the pulses. Notice that when the pulses exceed the bound of 6 °F a bias failure is detected. These are the samples at which the LLR exceeds the bound of 12.10. The LLRs for the two normal signals as computed for bias degradations are presented in Figs. 4.4.31 and 4.4.32. The pulses affect these values to a much reduced extent and at no sample is a false alarm made.

The effect of the pulses is almost completely eliminated by the weighting procedure. Only the smaller pulses remain. At the samples containing the large pulses the degraded signal has a weighting of zero in the computation of the estimate. The greatly reduced effect of the pulses is clearly observable in the plot of the weighted average as computed by the GCC, and is presented in Fig. 4.4.33.

As mentioned above, the pulses are also detected as noise type failures. This is confirmed by the behavior of the LLRs computed for the variance type degradation. Fig. 4.4.34 contains the plot of the LLR for signal 1.

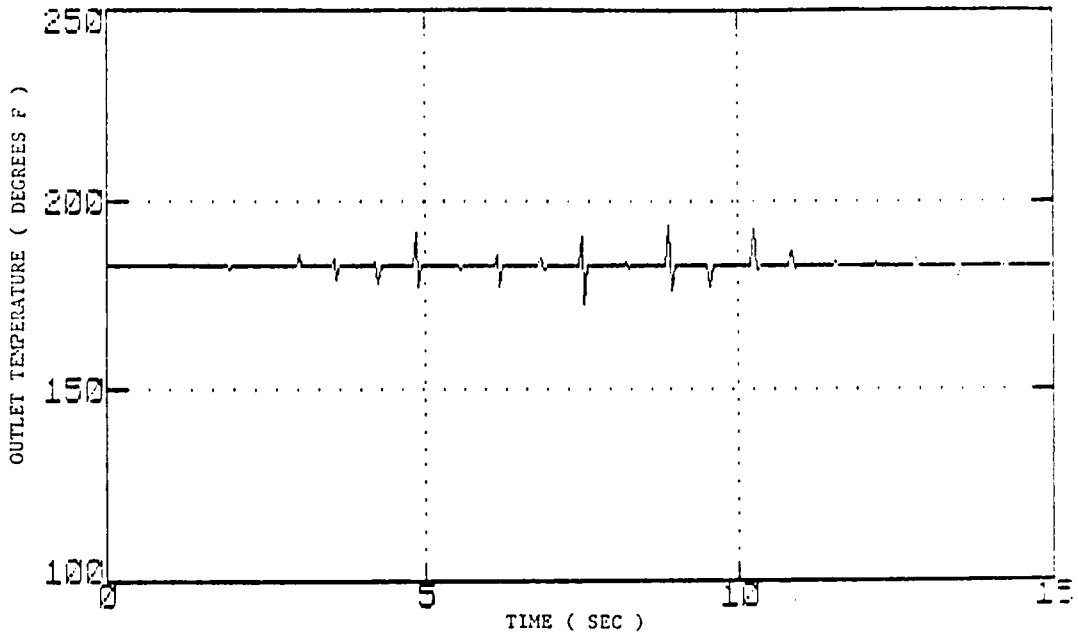



Figure 4.4.29 The First Thermocouple Signal Containing Pulses.

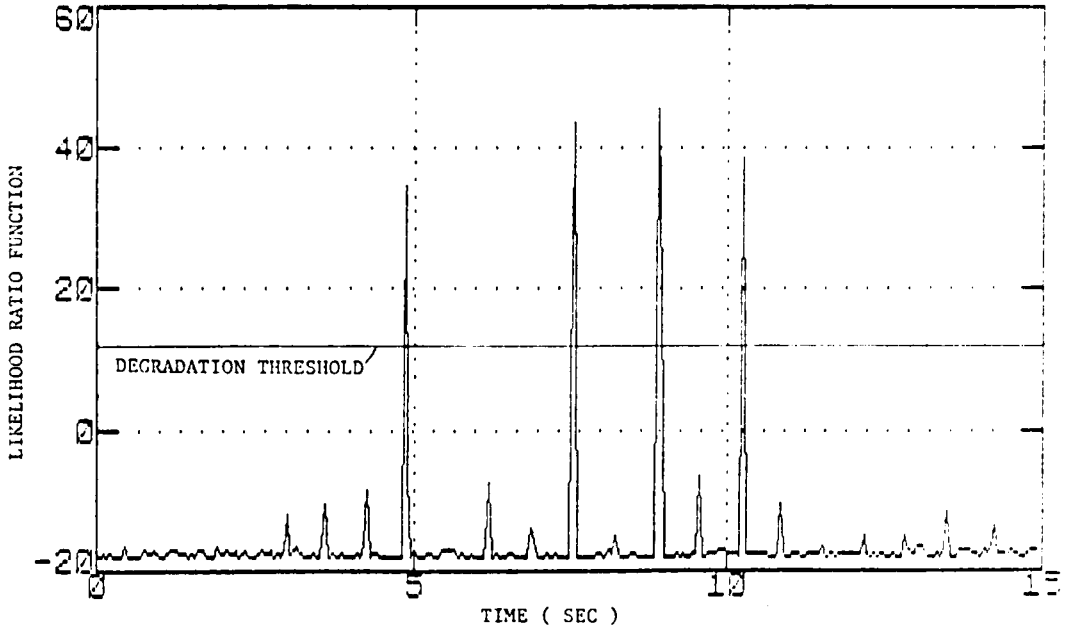



Figure 4.4.30 The Bias LLR for the First Thermocouple ( B1=12.1 ).

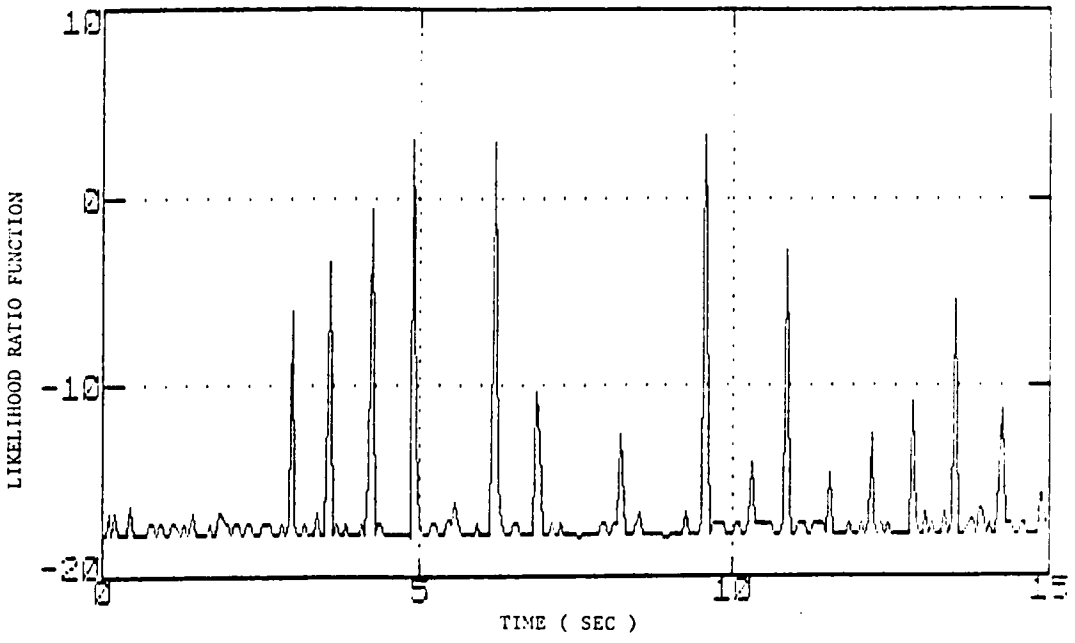



Figure 4.4.31 The Bias LLR for the Second Thermocouple ( B1= 12.1 ).

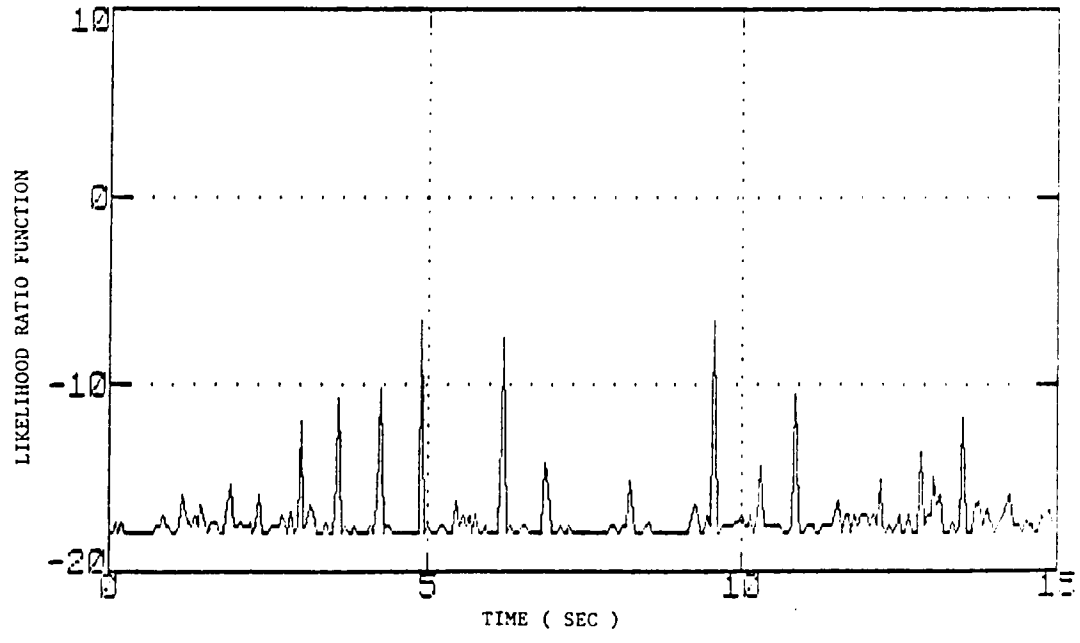



Figure 4.4.32 The Bias LLR for the Third Thermocouple ( B1= 12.1 ).

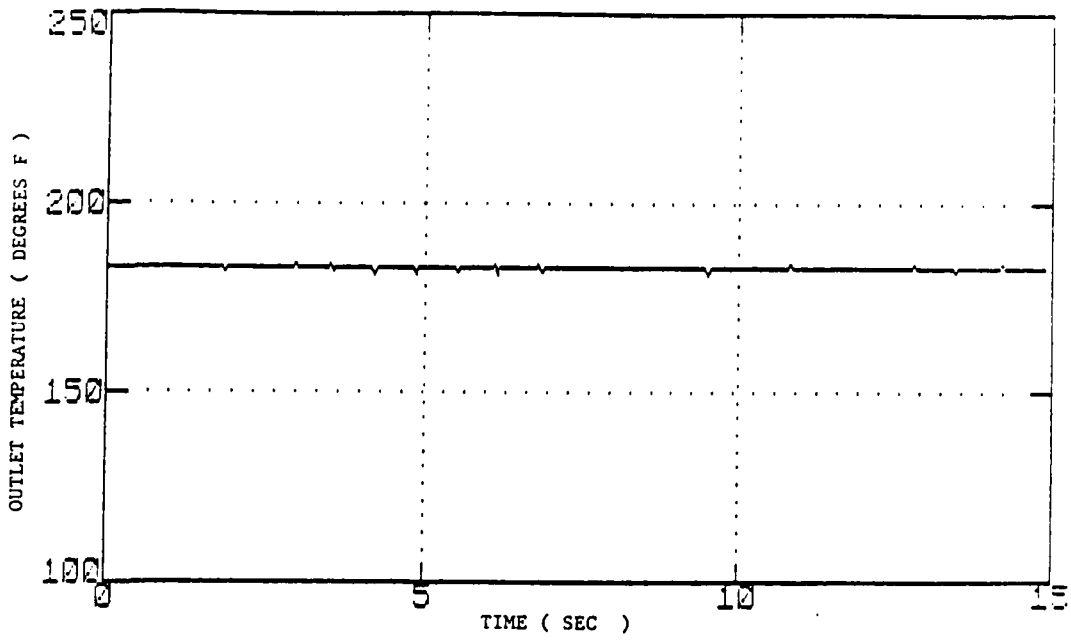



Figure 4.4.33 The Weighted Average of the Three Thermocouples.

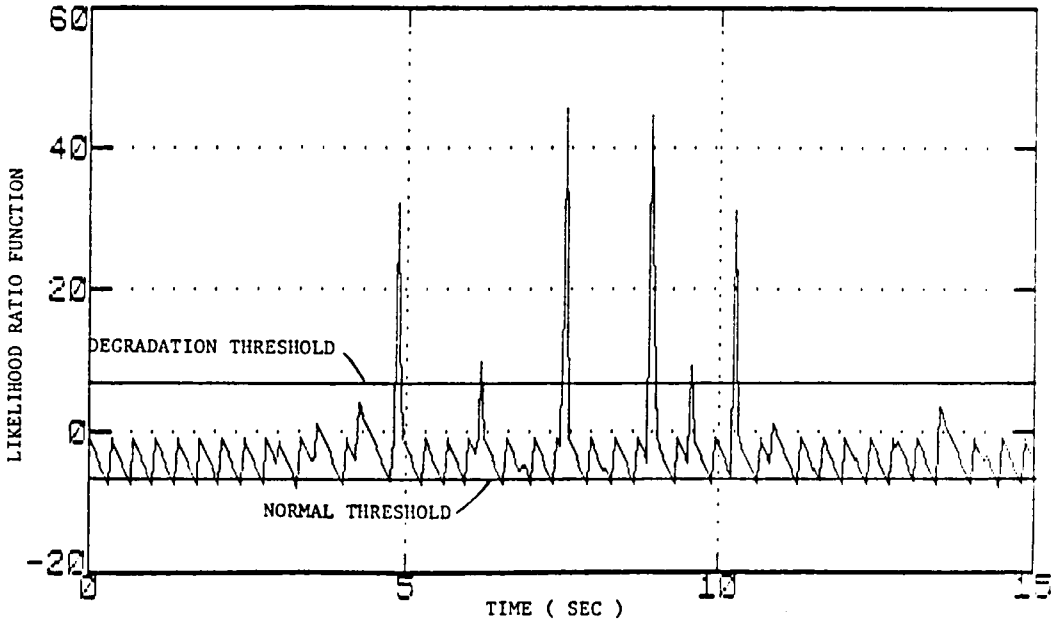



Figure 4.4.34 The Variance LLR for the First Thermocouple ( A=-6.91, B=6.91 ).

Observe the particularly interesting behavior of the LLR computed for the signal containing the pulses. At each pulse there exists a jump increase in the LLR. The larger the pulse magnitude the larger the increase in the LLR. After each jump the signal is normal again, and the LLR becomes negative until the normal mode bound of -6.91 is exceeded. The LLRs computed for the bias type degradation also return to normal levels between pulses. The LLRs computed for the two normal signals repeatedly exceed the normal mode bound of -6.91 as can be observed in Fig. 4.4.35.

This test illustrates the capability of the signal validation system to detect rapid bias type degradations with large magnitudes, as well as repeated pulse type failures as noise degradations. It also confirms that a desirable reconstructed signal can successfully be computed under these conditions.

## 4.5 Redundancy from Independent Sensors

An example of an empirical technique for obtaining an independent estimation of the change in the flow rate of air in the flow loop is described in this section. The concept consists of the determination of the transit time of air between two thermocouples using the fluctuating components of the thermocouple signals. One of the approaches for estimating the fluid transit time is to calculate the cross correlation function between the two thermocouple signals. This technique has been applied to determine the fluid flow velocities in cores of both boiling water and pressurized water reactors [26,27]. For this example the cross correlation function of the first and the seventh thermocouples (see Fig. 4.3.1, p. 56) is used. The random noise in the

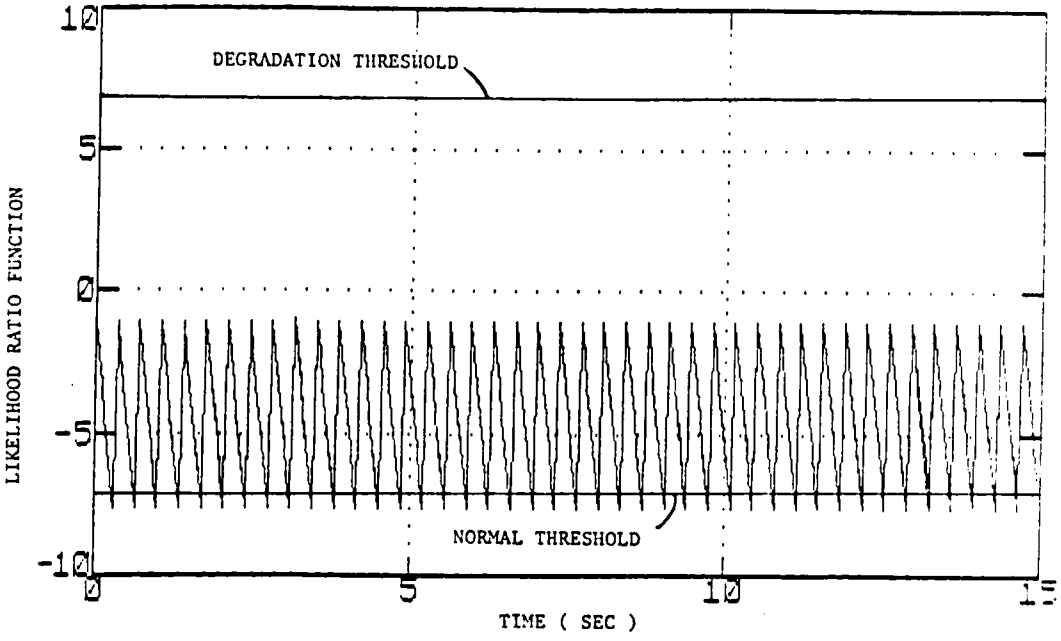



Figure 4.4.35 The Variance LLR s for the Second and Third Thermocouples ( A=-6.91,B=6.91 ).

thermocouple signals is generated mainly by fluctuations in the air flow caused by the fan.

Let $x_1(t)$ and $x_2(t)$ be the fluctuating components of the thermocouple signals $T_1(t)$ and $T_7(t)$. The cross correlation function between $x_1(t)$ and $x_2(t)$ is defined by

$$R_{12}(\tau) = E[x_1(t)\ x_2(t + \tau)] \tag{4.5.1}$$

where $\tau$ = time lag.

The assumption is now made that the signal $x_2(t)$ may be expressed as a function of $x_1(t)$ as follows

$$x_2(t) = c\ x_1(t - D) \tag{4.5.2}$$

where $c$ = constant and

$D$ = transit time of air between $x_1$ and $x_2$.

Substituting Eq. (4.5.2) into Eq. (4.5.1) the following expression results

$$\begin{aligned} R_{12}(\tau) &= E[x_1(t)\ cx_1(t - D + \tau)] \\ &= c\ E[x_1(t)\ x_1(t - D + \tau)] \\ &= c\ R_{11}\ (\tau - D) \end{aligned} \tag{4.5.3}$$

where $\tau - D$ = time lag.

It is known that the maximum value of an autocorrelation function exists at a time lag equal to zero. Equation (4.5.3) shows that the maximum value of $R_{12}(\tau)$ occurs at the maximum of $R_{11}(\tau - D)$. The

maximum value of $R_{11}(\tau - D)$ occurs at $\tau = D$ $(\tau - D = 0)$. This shows that the maximum value of the cross correlation function $R_{12}$ occurs at a time lag equal to the transit time D of air between $x_1$ and $x_2$. A change in the location of the peak of the cross correlation function should indicate a change in the transit time or flow velocity of the air. This provides, in principle, an empirical estimation of the air flow rate based on signals from two temperature sensors.

The experimental air flow loop is used to obtain the signals. Signals are recorded for three cases:

(1) full heater power and reduced flow rate,

(2) full heater power and full flow rate, and

(3) reduced heater power and full flow rate.

The amplitude probability density function as computed for one of the noise signals is shown in Fig. 4.5.1. The density function is Gaussian in nature and this confirms the assumption made in Chap. 3 concerning the Gaussian nature of the signals.

The cross correlation functions between the first and seventh thermocouples are computed for all three cases and shown in Figs. 4.5.2, 4.5.3, and 4.5.4. Due to uncertainties in measuring the exact air flow rate as determined by the pitot tube, the application discussed here yields a qualitative verification of the transit time using diverse sensors. However, the quantitative verification should be possible using this technique.

The ratio of the flow rate in case (1) to the flow rate in case (2) is 0.59. The corresponding ratio (taken in the reverse order) of the peak locations shown in Figs. 4.5.2 and 4.5.3 is 0.66. These

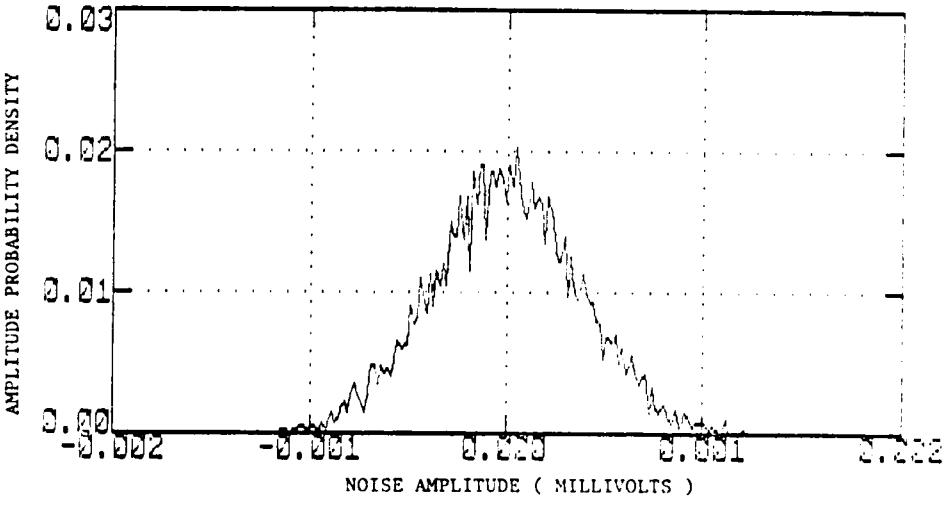



Figure 4.5.1 Amplitude Probability Density Function for Outlet Thermocouple Noise Confirming the Gaussian Nature of the Signal ( Variance= $1.38 \times 10^{-7}$ $mV^2$ ).

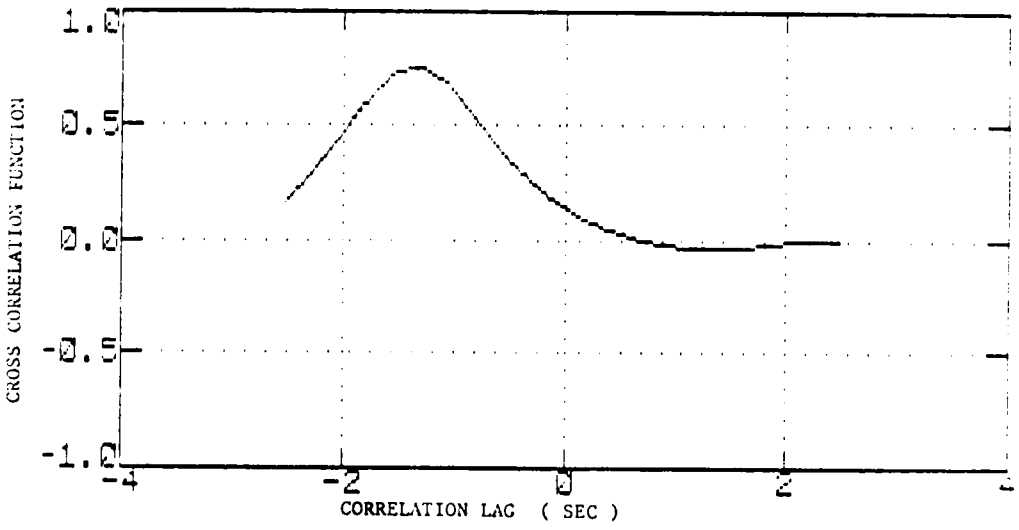



Figure 4.5.2 Cross Correlation Function of the First and Seventh Thermocouples ( Case 1 ).

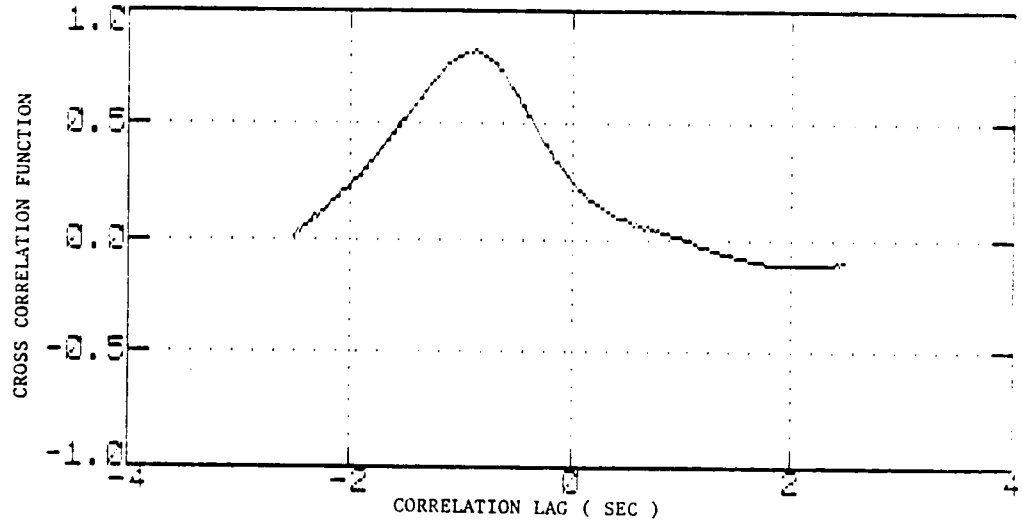



Figure 4.5.3 Cross Correlation Function for the First and Seventh Thermocouples ( Case 2).

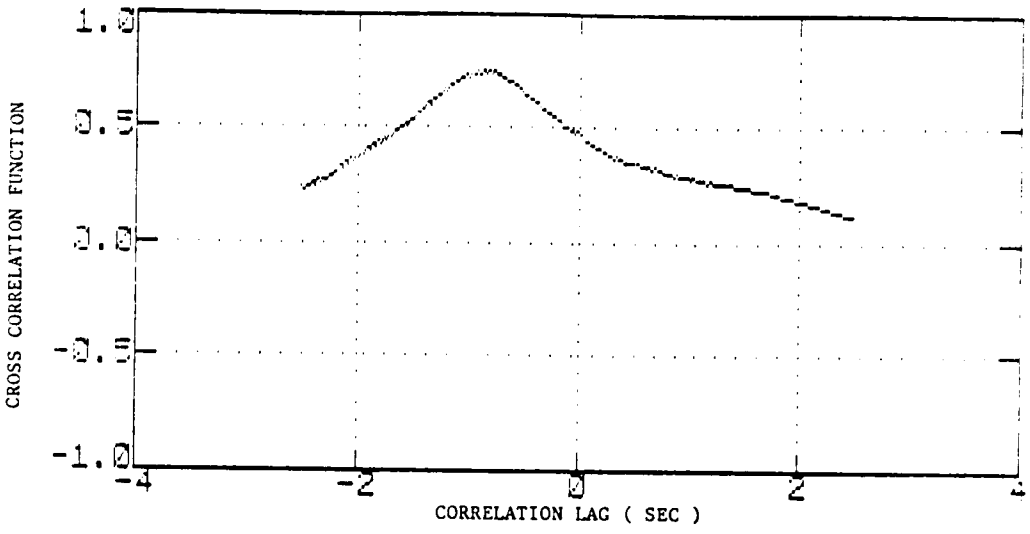



Figure 4.5.4 Cross Correlation Function of the First and Seventh Thermocouples ( Case 3 ).

ratios are within 10% of each other. This illustrates that as the air flow rate is increased there is a corresponding decrease in the transit time of air between the two thermocouples. Thus a qualitative verification of the flow sensor is obtained.

In case (3) the flow rate is the same as in case (2) while the heater power is changed. As shown in Fig. 4.5.4 the shape of the cross correlation function changes, while the location of the peak remains the same. Again this verifies the flow sensor measurement. The other implication of this observation is that the thermocouples are verified to be installed correctly. A thermocouple, removed partially from the pipe, measures the temperature in a different location and since the cross sectional flow profile is not constant across the piping, a change in the transit time should be observed. The same argument applies to the location of the pitot tube. A complete verification of the sensor anomaly requires the knowledge of actual values of the process variables. Additional verification of the transit time may be obtained by computing the cross correlation functions between other pairs of thermocouples. This is an example of the use of redundancy by diversity, in sensor failure detection.

Although a quantitative approach may be pursued, the qualitative discussion above may be very helpful in the construction of an expert system for signal validation. The transit time concept is one example of the features that may be included in the knowledge base of such an expert system.

## 4.6 Summary

A description of the application and verification of the signal validation system is presented in this chapter. In order to obtain realistic signals an experimental facility consisting of a closed air flow loop has been constructed. Thermocouples measuring the air temperature at various locations in the loop are used to provide signals contaminated with a variety of failures.

An analog to digital conversion and recording system has been developed for data acquisition. The complete software package consists of programs for data collection, display, data processing, and for signal validation.

The signal validation system proved successful in detecting a variety of failures. Bias, variance, and pulse type failures can be detected under both steady state and transient conditions. The results presented in this chapter confirm this conclusion.

As illustrated in Sect. 4.4 the trajectories of the LLRs have different characteristics when computed for different failure situations. These characteristics may contain useful information even in situations where failures may not be detected. This information would be useful if included in the data base of a future expert system for signal validation.

The use of diverse sensors to monitor a measurement is illustrated through the use of temperature fluctuations from adjacent thermocouples. The deviation of the specific parameter is also an indication of faults in the diverse sensors themselves.

# CHAPTER 5

# CONCLUDING REMARKS

## 5.1 Conclusions

This study contains the development of a signal validation system based on the use of redundant arrays of sensors measuring the same process variable. Two signal validation methodologies, the Generalized Consistency Check and the Sequential Probability Ratio Test, are combined to form a single system incorporating the advantages of both. The Generalized Consistency Check provides a computationally efficient management system for redundant arrays of signals, and provides a reconstruction of the signal under consideration. The Sequential Probability Ratio Test utilizes the measurement history of a sensor to make decisions about the degradation condition of each sensor based on specified missed and false alarm probabilities. The combined Generalized Consistency Check and Sequential Probability Ratio Test is utilized mainly to detect constant bias and bias drift type degradations. The Sequential Probability Ratio Test is also extended to the detection of increasing signal noise variance levels. This is accomplished by separating the mean value and the fluctuating component of the signals during data recording. A signal validation algorithm has been developed based on the above concept. Other algorithms used for recording, processing, and for display of signals are part of the overall software package.

An experimental air loop has been constructed to generate signals

from a real process for use in the verification of the signal validation algorithm. Thermocouples used to measure air temperatures at different locations in the loop provided the necessary signals. Various combinations of bias and noise type degradations have been introduced in the signals by removing sensors from the air flow and by changing the filtering and grounding of the signals.

The verification procedure confirmed the ability of the signal validation system to detect and isolate bias degradations, noise-type degradations, simultaneous bias and noise degradations, and repeated pulse type degradations under both steady state and transient conditions. The system is also able to provide a reconstruction of the signal during all the failure situations considered.

An example illustrating the use of the fluctuating components of the thermocouple signals to obtain an independent verification of the flow measurement is also presented. In this case a qualitative verification is obtained. However, it is expected that this technique can be improved to obtain quantitative verification.

## 5.2 Recommendations

The final step in the verification of the signal validation system is its successful application to an actual industrial process. In order to accomplish this the data collection, data processing, and the signal validation algorithms must be incorporated into a single unit. It may not be necessary to convert voltage signals to actual physical units of the process variables. This would speed up the signal validation procedure.

An improved Sequential Probability Ratio Test utilizing a statistical control law does not yet exist for the case of noise degradation. The development of such a test would improve the efficiency of the noise degradation detection.

Many interesting possibilities exist for the utilization of artificial intelligence techniques for signal validation. The parallel use of knowledge-based expert systems and the approach used in this study should yield a very efficient signal validation system.

The extension of the signal validation system to the processing of redundant signals containing common cause failures is a natural extension of the study described in this work. A good discussion of a generalized decision estimator for common cause failures is given in Ref. [25].

The methodology is also readily applicable for monitoring slow sensor degradation such as calibration changes. In this application, the data would be recorded at large sampling intervals.

# LIST OF REFERENCES

## LIST OF REFERENCES


1. Upadhyaya, B. R., "Sensor failure Detection and Estimation," Nuclear Safety, Vol. 26, No. 1, January 1985, pp. 32-43.

2. Potter, J. E. and Suman, M. C., "Thresholdless Redundancy Management with Arrays of Skewed Instruments," Integrity in Electronic Flight Control Systems, NATO AGARDOGRAPH-224, 1977, pp. 15-1 to 15-25.

3. Deutch, O. L., "Signal Validation Advanced Development for On-Line Monitoring and Control," Proceedings of the Sixth Power Plant Dynamics, Control, and Testing Symposium, Knoxville, Tennessee, April 1986, pp. 37.01 - 37.09.

4. Ray, A., Bernard, J. A., and Lanning, D. D., "On-Line Signal Validation and Feedback Control in a Nuclear Reactor," Proceedings of the Fifth Power Plant Dynamics, Control, and Testing Symposium, Knoxville, Tennessee, March 1983, pp. 38.01 - 38.20.

5. Zwingelstein, "Application of the Parity Space Technique to the Validation of the Water Level Measurement of Pressurizer for Steady State and Transients," Proceedings of the Fifth Power Plant Dynamics, Control, and Testing Symposium, Knoxville, Tennessee, March 1983, pp. 28.01 - 28.24.

6. Fisher, J. et al., "Demonstration of a BWR Suppression Pool Signal-Validation Program," EPRI NP-3641, RP2126-3, Final Report, July 1984.

7. Reed, R. P., "Validation Diagnostics for Defective Thermocouple Circuits," Temperature, Its Measurement and Control in Science and Industry, Vol. 5, American Institute of Physics, 1982, pp. 931-938.

8. Code of Federal Regulaitons, "Energy", 10 CFR 0.735-1, U. S. Government Printing Office, Jan. 1986, p. 498.

9. Wald, A., "Sequential Tests of Statistical Hypotheses," Ann. Math. Statist., Vol. 16, pp. 117-186, 1945.

10. Ball, K. E., "Mini-Perspective in Safety Controls," Intech, International Journal of Instrumentation and Control, p. 47, July 1986.

11. Ludtka, G. M. and Kollie, T. G., "Performance of Chromel Versus Alumel and Nicrosil Versus Nisil Thermocouple Assemblies in Vacuum and Argon between 1000° and 1200°C," Proceedings of the Industrial Temperature Measurement Symposium, Knoxville, Tennessee, September, 1986, pp. 5.01 - 5.21.

12. Myhre, D. C. and Stickney, T. M., "Estimating the Accuracy of an Installed Platinum Resistance Thermometer," Proceedings of the Industrial Temperature Measurement Symposium, Knoxville, Tennessee, September, 1984, pp. 12.01 - 12.18.

13. Code, B., "Long-Term Thermistor Stability at an Elevated Temperature," Proceedings of the Industrial Temperature Measurement Symposium, Knoxville, Tennessee, September, 1984, pp. 17.01 - 17.13.

14. Upadhyaya, B. R., "Survey of In-Core Instrumentation in Operating Pressurized Water Reactors," Analysis & Measurement Services, AMS-BRU 8301RO, March 1984.

15. Cikanek, H. A., "Space Shuttle Main Engine Failure Detection," IEEE Control Systems Magazine, June 1986, pp. 13-18.

16. Deckert, J. C. et al., "F-8 DFBW Sensor Failure Identification Using Analytic Redundancy," IEEE Transactions on Automatic Control, Vol. AC-22, No. 5, October 1977, pp. 795-803.

17. Desai, M. N. and Deckert, J. C., "Dual-Sensor Failure Identification Using Analytic Redundancy," J. Guidance and Control, Vol 2, No. 3, May-June 1979, pp. 213 - 220.

18. Chien, T. T. and Adams, M. B., "A Sequential Failure Detection Technique and Its Application," IEEE Transactions on Automatic Control, Vol. AC-21, No. 5, October 1976, pp. 750 - 757.

19. Deutsch, O. L. and Ornedo, R. S., "Implementation of Real-Time Signal Validation at EBR-II," ANS Transactions, Vol. 45, pp. 660 - 661 (1983).

20. Blancaflor, M. M. et al, "The Implementation of Signal Validation on the Millstone Point Unit no. 3 Safety Parameter Display System," Proc. of EPRI Seminar: Nuclear Power Plant Safety Control Technology, Palo Alto, California, Feb. 4-6, 1985.

21. Clark, R. N. and Campbell, B., "Instrument Fault Detection in a Pressurized Water Reactor Pressurizer," Nucl. Techn., 56, 23 (1982)

22. Chien, T. T., "An Adaptive Technique for a Redundant-Sensor Navigation System," Charles S. Draper Lab., Cambridge, MA, Rep. T-560, February 1972, p. 238.

23. Desai, M. N. and Ray, A., "A Fault Detection and Isolation Methology," Proc. 20th IEEE Conf. on Decision and Control, San Diego, California, December 1981, pp. 1363 - 1369.

24. Newbold, P. M. and Ho, Y. C., "Detection of Changes in the Characteristics of a Gauss-Markov Process," IEEE Trans. Aerospace and Electronic Systems, Vol. AES-4, No. 5, Sept. 1968, pp. 707 - 718.

25. Kujawski, E. et al., "Signal Validation Software Utilizing a Generalized Decision Estimator for Common Cause Failures," Trans. ANS Topical Meeting on Computer Applications for Nuclear Power Plant Operation and Control, 1985, pp. 551 - 558.

26. Kosaly, G. et al., "Investigation of the Local Component of the Neutron Noise in a BWR and Its Application to the Study of Two-Phase Flow," Prog. in Nucl. Energy, 1 (2-4): 99-117 (1977), Pergamon Press.

27. Sweeney, F. J. and Upadhyaya, B. R., "Measurement of Core Coolant Flow Velocities in PWRs Using Temperature-Neutron Noise Cross Correlations," Trans. Am. Nucl. Soc., 43: pp. 795-797 (1982).

# APPENDIXES

# APPENDIX A

# DECISION ESTIMATOR MODULE

# APPENDIX A

# DECISION ESTIMATOR MODULE

## A.1 Introduction

This Appendix is devoted to the discussion of the decision estimator module, "CONSIST.EXE". The first part of the discussion concerns the input to the algorithm. An explanation of each of the user interactive inputs is provided. Secondly, the output messages generated while the algorithm is executed are discussed. Finally, a listing of the algorithm "CONSIST.FOR" is given.

The algorithm "CONSIST.FOR" is written in Microsoft FORTRAN 77 V3.30. "CONSIST.FOR" has been linked and compiled using an IBM PC with an 8087 mathematics coprocessor. The algorithm can be executed on most IBM compatible microcomputers using the MS DOS 2.10 operating system.

## A.2 Input Information

To initiate the execution of "CONSIST.EXE" the user must respond to a number of interactive prompts from the key board. In this discussion each prompt is listed and the proper response is discussed.

"ENTER TIME STEP BETWEEN SAMPLES"

This input requires a real number that specifies the period between subsequent samples. The value is determined by the frequency at which the analog to digital conversion is performed during the recording of the data. The value is typically of the

order of 1 second in the nuclear and aerospace applications.

"ENTER ANALYTICAL FILE NAME"

This input requires the name of the data file containing the analytical signal as generated by an analytical model. The file name consists of a six character name and a three character extension. Only one signal is used from this file. If no analytical signal is available the name of the experimental file may be entered and the first experimental signal may be used in place of the analytical signal. This implies that the analytical and experimental files have the same file structure.

The analytical and experimental files consist of up to 1000 records. This limit depends on the available memory of the computer used and may be increased or decreased by changing the type declaration statement in "CONSIST.FOR". Each record contains up to 15 real numbers representing simultaneous samples of different signals at a sampling instant. Each signal is represented by a real number with record length equal to four bytes.

The algorithm is arranged such that a maximum of one signal from the analytical file, and a maximum of four signals from the experimental file may be used. If no analytical signal is available, a maximum of five experimental signals may be used.

"NUMBER OF SIGNALS IN ANALYTICAL FILE"

This is an integer input ranging between 1 and 15. The number indicates the number of signals in each record. The maximum

value of 15 depends on the memory size of the computer that is used.

"CHANNEL NO. OF DESIRED ANALYTICAL SIGNAL"

This is an integer input ranging between 1 and the number of signals in the file as specified above. The number indicates which analytical signal to use in the signal validation process.

"NUMBER OF POINTS IN FILE?"

This value is an integer and indicates the number of records in the analytical and experimental file. This number has a maxiumum of 1000 in the current version of the program, and is limited by the memory size of the computer. It can be changed by changing the type declaration statement in "CONSIST.FOR". It must be pointed out that the algorithm is capable of processing 1000 records or samples. The data file may contain more than 1000 samples. As discussed below, the option does exist to skip a variable number of records. It is therefore possible to apply the algorithm to a data file containing 2000 records by skipping every second record. The operator must keep in mind that the time step between processed records is now not the same as the original time step between subsequent records.

"EXPERIMENTAL FILE NAME?"

This entry is similar to the analytical file name and consists of a six character file name with a three character extension name. The file structure is the same as that for the analytical file, except that up to four signals from the experimental file may be used by

the algorithm for signal validation purposes.

"NUMBER OF SIGNALS IN EXP. FILE?"

Similar to the analytical file this is an integer number ranging between 1 and 15. It depends on the number of signals that are digitized. Not all of these signals may be necessary for signal validation. The number of signals and which signals to use from the experimental file are specified in the next two interactions.

"NUMBER OF SIGNALS TO USE IN EXP. FILE?"

This is an integer value ranging between one and four. The maximum is limited only by the memory size of the computer and may be changed by changing the dimension statement of the algorithm. This value specifies how many of the signals in the experimental file will be used for signal validation purposes. It also determines how many times the following prompt will appear.

"CHAN. NUM. OF SIGNAL"

This prompt appears as many times as the number of signals to use from the experimental file. It allows the operator to specify which signals to use from the experimental file. Each time the prompt appears the operator responds with an integer channel number ranging between one and the maximum number of signals in the experimental file. The responses do not have to be in increasing or decreasing numerical order.

"ENTER IFLAG = 1 TO PROCESS NOISE DATA"

To process noise signals where a noise data file exists IFLAG is

set equal to one. If IFLAG is not equal to one the algorithm will not attempt to process the noise signals and the following three prompts are skipped.

"ALPHA = ? , BETA = ?"

This prompt requires the specification of the missed alarm probability, $\alpha$, and the false alarm probability, $\beta$. These numbers are real and have typical values of 0.1%. The Sequential Probability Ratio Test, as developed by Wald[9], is directly applied to the noise signals.

"NOISE DATA FILE NAME?"

Similar to the analytical and the experimental file names, the noise data file contains a six character name and a three character extension. The structure of the noise file is also the same as that of the previous data files. The data file consists of a number of records. Each record corresponds to a different sample. Each record may contain one to fifteen noise signals. This is again limited by the computer memory size. The fluctuation or noise signals are obtained by filtering out the low frequency components of the original signals. The filtering may be performed digitally or by using conventional high pass filters.

"NO. OF SIGNALS IN NOISE FILE"

This prompt requires an integer response corresponding to the total number of signals to be validated. The value specifies the

number of signals in each record of the noise data file.

"NUMBER OF SAMPLES TO SKIP?"

This prompt allows the operator to skip samples between samples used for signal validation purposes. This may be the case if the original sampling rate is too high. The number should be an integer larger than or equal to one. Increasing the time step also helps to avoid correlation concerns that arise during the application of the Sequential Probability Ratio Test.

"ENTER THE SPECIFIED MTIME BETWEEN FALSE ALARMS"

This is where the false alarm criterion for the modified Sequential Probability Ratio Test is specified. The relationship between the missed and false alarm probabilities and the mean time between false alarms is presented in Fig. 3.4.3. Keep in mind that the missed alarm probability is zero in the modified Sequential Probability Ratio Test. To get values of the false alarm probability of the order of 0.1%, values of MTIME of the order of 10,000 time steps is recommended. This input is an integer number.

ENTER B, THETA1, VARØ, VAR1 FOR EACH SIGNAL"

This interaction specifies acceptable and unacceptable signal error magnitudes and variances. The variables are defined as follows:

B = the error bound for the generalized consistency check,

THETA1 = the maximum acceptable degradation magnitude as implemented by the Sequential Probability Ratio Test. (It is normally taken to be twice the value of B, but may be specified independently.),

VARØ = the normal variance of the signal under analysis,

VAR1 = the maximum acceptable variance of the signal under analysis.

All the above values may be entered under free format. The values are entered as follows

B, $\Theta_1$, $\sigma_o^2$, $\sigma_1^2$ (RETURN), for each signal.

"ENTER ESTIMATE, SPRM, SPRV FNAMES"

This is the point where the output file names are specified. These files are created during the execution of "CONSIST.EXE." Each file name contains a six character file name and a three character extension. The files are defined as:

ESTIMATE = the file containing the single estimate of the signal at each sample,

SPRM = the file containing the value of the sequential ratio for bias degradation at each sample and,

SPRV = the file containing the Sequential Ratio for noise degradation at each sample.

The estimate file contains one variable. Each new sample is recorded in a new record. The SPRM and SPRV files contain the

sequential ratios of each signal. The ratios for all signals at each sample are contained in a single record. The number of ratios per record are determined by the number of signals. If noise failures are not considered all the values in the SPRV file remain zero.

This completes the discussion of the required operator interaction to initialize the algorithm. During the development of the algorithm one of the objectives was to maintain a simple input structure. This is accomplished by limiting the number of inputs and avoiding the inclusion of complex theoretical aspects in the algorithm/operator interaction. The display generated during the program execution is discussed in the next section.

## A.3 Display Information

The display generated during the execution of the algorithm "CONSIST" is in the form of a table containing several variables for each signal. A sample output table is presented in Fig. A.3.1. The table is currently updated at every sample. In future versions an option will be added such that the table will not be updated at every sample. Depending on the computer used and the sampling rate it may be hard to read the table at every sample. The information appearing on the screen is now discussed in the order of appearance.

"TABLE UPDATED AT ALL SAMPLES & FAILURES
FAILURES INDICATED BY FLASHING COLUMN HEAD"

This statement indicates that every sample is represented on the

TABLE UPDATED AT ALL SAMPLES & FAILURES
FAILURES INDICATED BY FLASHING COLUMN HEAD

FAILURE INFORMATION:
ALL SENSORS ARE USED IN ESTIMATION
SENSOR #: , TYPE:

ESTIMATE (DEG. F) = 181.17 , SAMPLE # = 1

| | SIGNAL#1 | SIGNAL#2 | SIGNAL#3 |
|---|---|---|---|
| VALUE | 180.33 | 181.38 | 181.80 |
| SPRM | -4.64 | -7.16 | -5.47 |
| SPRV | -.66 | .07 | -.66 |
| BOUND | 4.38 | 4.38 | 4.38 |
| BOUNDA | -6.91 | -6.91 | -6.91 |
| BOUNDB | 6.91 | 6.91 | 6.91 |
| INDEX | .00 | .00 | .00 |

Figure A.3.1 Example of Information Displayed to the Operator.

screen. No sample is skipped. It also indicates that in addition to a failure message, the column head of the failed sensor will flash.

"FAILURE INFORMATION"

When a failure or inconsistency is detected it will be indicated under this heading. The first line represents "consistency" messages generated by the Generalized Consistency Check. The second line, "SENSOR#: , TYPE: ", indicates which sensor has degraded and what kind of degradation is detected by the sequential testing portion of the algorithm. (A detailed description of the Sequential Probability Ratio Test and the Generalized Consistency Check is presented in Chap. 3.)

"ESTIMATE (DEG.F) = , SAMPLE # = "

This line indicates the estimate of the temperature, as obtained by the generalized consistency check, as a function of sample number or time. This is one of the more important displays of the algorithm.

"SIGNAL #1 SIGNAL #2 SIGNAL #3"

The rest of the pertinent information that is used and generated during the algorithm execution is generated under this heading. The algorithm may be used to validate more than three signals even though only up to three signals will be displayed. To increase the number of signals displayed require only minor changes. Each of the values appearing in the table is discussed below.

"VALUE"

To the right of this statement the value of each sensor output is displayed and updated at every sample. This allows easy comparison of a particular sensor value to the estimate of the signal and the other sensors.

"SPRM"

To the right of this statement the sequential ratio for each sensor as calculated by the Sequential Probability Ratio Test is displayed. These values represent the result of the Sequential Test as applied to bias type degradations.

"SPRV"

The values of the sequential ratios as calculated for noise type degradation are displayed next to this statement.

"BOUND"

This display contains the bound against which the values "SPRM" are compared to check for bias degradations. This bound is calculated using the mean time between false alarms, degraded mean $\Theta_1$, and the degraded variance $\sigma_1^2$.

"BOUNDA"

These values represent the bounds against which the values "SPRV" are checked to determine whether noise degradations do exist.

"BOUNDB"

These values represent the bounds against which the values "SPRV" are checked to determine whether no noise degradations exist.

"INDEX"

The values are generated by the generalized consistency check and indicates which sensors are inconsistent with respect to others. The consistency messages displayed at the top of the table correspond to these index values. (Details about the index can be found in Chap. 3.)

This completes the discussion of the display. The objective of the display is to provide the operator with a simplified representation of the pertinent information. However, enough information is presented such that an operator familiar with the signal validation is able to make judgements about the performance of the failure detection system. A detailed knowledge of the signal validation system is not required for the use of the algorithm.

## A.4 Algorithm Listing

A listing of the program CONSIST.FOR is given in this section. Effort has gone into providing adequate documentation and information statements throughout the algorithm. The documentation statements conform to statements appearing in Fig. A.4.1, the flow chart upon which the consistency checking component of the algorithm is based.

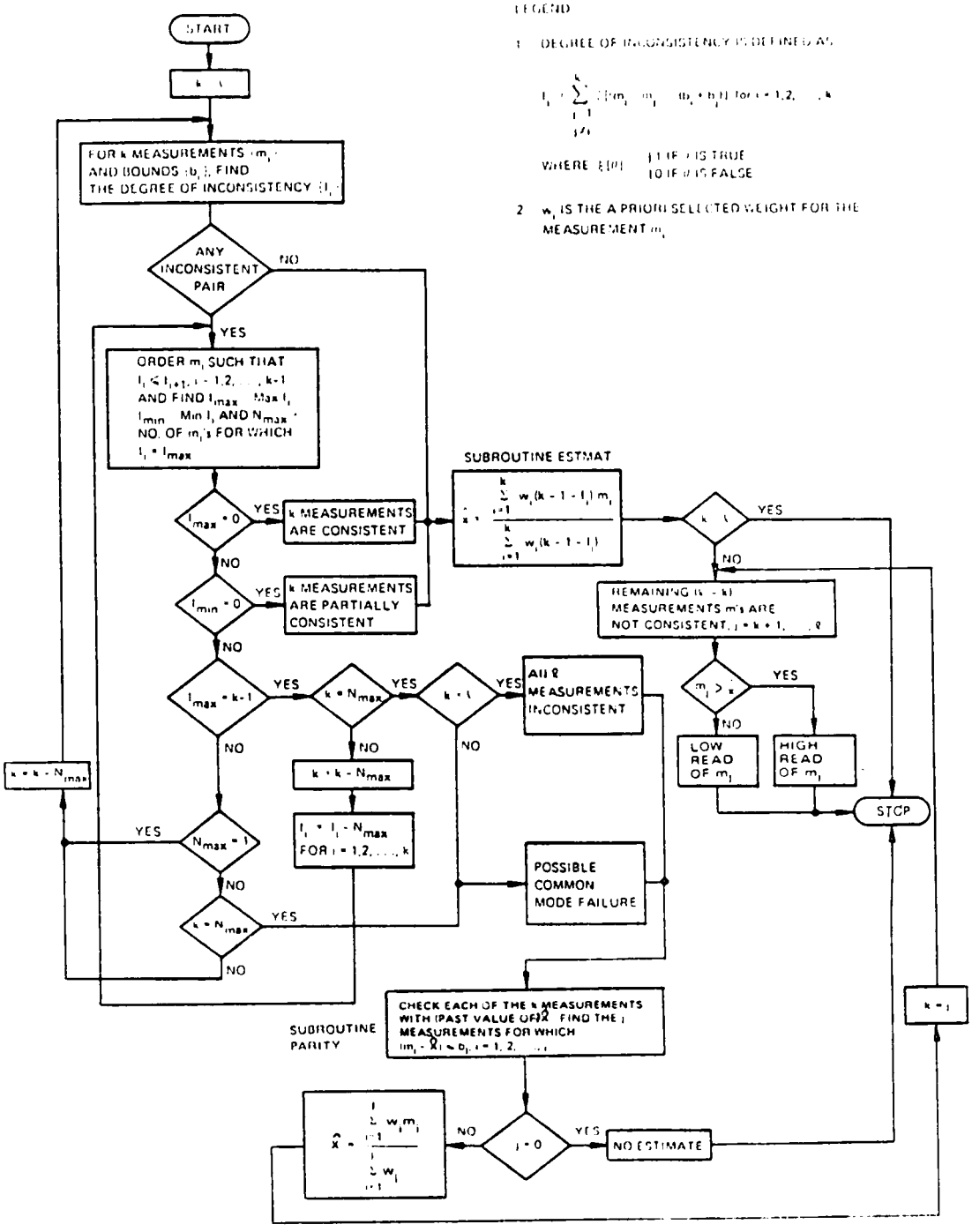



Figure A.4.1 Flow Chart Illustrating the Generalized Consistency Checking Procedure ( From Ref. 6, p. A-6 ).

```
D Line# 1     7                          Microsoft FORTRAN77 V3.30 March 1985
      1 $STORAGE:2
      2 $NOFLOATCALLS
      3 C***      PROGRAM GENERALIZED CONSISTENCY CHECK
      4 C
      5           REAL*4 MI(5),X(5),Y(5),B(5),E,B1,M(5,4),MP(5),FALSE,MISS
      6           REAL*4 XESTMT,P(5,4),SUM1,SUM2,X1(1000),Y2(1000,5),TEST
      7           REAL*4 SPRM(5),SPRV1(5),DELTAT,TIME,MTIME,BOUND(5),TEMP1
      8           REAL*4 TEMP3,TEMP4,SPRV2(5),VAR0(5),VAR1(5),THETA1(5)
      9           REAL*4 NOISE(1000,5),ALPHA,BETA,BOUNDA,BOUNDB
     10           INTEGER*2 N1,IRECL1,CHAN1,NPOINT1,N2,IEXP,NSAMPL,NSKIP,ITOTAL
     11           INTEGER*2 IK,INDEX(5),CHAN2(5),NINCNST,INUMSIG,N(5),J,L,N3
     12           CHARACTER*40 FNAME1,FNAME2,FNAME3,FNAME4,FNAME5,FNAME6
     13 C
     14 C***      SET UP AN ARRAY OF PAST MEASUREMENTS (ASSUME ALL=0 FOR NOW)
     15 C
     16           DO 5,I=1,15
1    17           P(I,1)=I
1    18           P(I,2)=I
1    19           P(1,4)=0
1    20 5         CONTINUE
     21           P(1,3)=0
     22           P(2,3)=0
     23           P(3,3)=0
     24           P(4,3)=0
     25 C
     26 C***      ENTER FILE INFORMATION
     27 C***      NAME OF ANALYTICAL FILE,EXPERIMENTAL FILE,NOISE FILE
     28 C
     29           WRITE(*,*) 'ENTER TIME STEP BETWEEN SAMPLES'
     30           READ(*,15) DELTAT
     31 10        FORMAT(6I6)
     32 15        FORMAT(F7.3)
     33           WRITE(*,'(A/)') ' ENTER ANALYTICAL FILE NAME'
     34           READ(*,'(A)') FNAME1
     35           WRITE(*,'(A/)') ' NUMBER OF SIGNALS IN ANALYTICAL FILE'
     36           READ(*,10) N1
     37           IRECL1=4*N1
     38           WRITE(*,'(A/)') ' CHANNEL NO. OF DESIRED ANALYTICAL SIGNAL'
     39           READ(*,10) CHAN1
     40           WRITE(*,'(A/)') ' NUMBER OF POINTS IN FILE?'
     41           READ(*,10) NPOINT1
     42 C***      EXPERIMENTAL FILE INFORMATION
     43           WRITE(*,'(A/)') ' EXPERIMENTAL FILE NAME ?'
     44           READ(*,'(A)') FNAME2
     45           WRITE(*,'(A/)') ' NUMBER OF SIGNALS IN EXP. FILE ?'
     46           READ(*,10) N2
     47           IRECL2=4*N2
     48           WRITE(*,'(A/)') ' NUMBER OF SIGNALS TO USE IN EXP. FILE ?'
     49           READ(*,10) IEXP
     50           DO 100,J=1,IEXP
1    51           WRITE(*,'(A/)') ' CHAN NUM. OF SIGNAL'
1    52 100       READ(*,10) CHAN2(J)
     53 C
     54 C***      ENTER NOISE DATA INFORMATION
     55 C
     56           WRITE(*,*) ' ENTER FLAG=1 TO PROCESS NOISE DATA'
```

```
D Line# 1      7                          Microsoft FORTRAN77 V3.30 March 1985
      57            READ(*,10) INOISE
      58            IF (INOISE.EQ.1) THEN
      59            WRITE(*,*) 'ALPHA=? , BETA=?'
      60            READ(*,15) ALPHA,BETA
      61            BOUNDB=LOG((1-BETA)/ALPHA)
      62            BOUNDA=LOG(BETA/(1-ALPHA))
      63            WRITE(*,*) ' NOISE DATA FILE NAME ?'
      64            READ(*,'(A)') FNAME6
      65            WRITE(*,*) ' NO. OF SIGNALS IN NOISE FILE'
      66            READ(*,10) N3
      67            IRECL7=4*N3
      68            ENDIF
      69            WRITE(*,'(A/)') ' NUMBER OF SAMPLES TO SKIP ?'
      70            READ(*,10) NSAMPL
      71            NSKIP=NSAMPL+1
      72            ITOTAL=NPOINT1/NSKIP
      73 C
      74 C***       ENTER THE SPECIFIED MEAN TIME BETWEEN FALSE ALARMS
      75 C
      76            WRITE(*,*) ' ENTER SPECIFIED MTIME BETWEEN FALSE ALARMS'
      77            READ(*,*) MTIME
      78            DO 315,K=1,INUMSIG
1     79            SPRM(K)=0
1     80            SPRV1(K)=0
1     81            BOUND(K)=0
1     82 315        SPRV2(K)=0
      83 C
      84 C***       READ IN THE ERROR BOUNDS FOR EACH SIGNAL
      85 C
      86            WRITE(*,'(A)') ' ENTER THE B,THETA1,VAR0,VAR1 FOR EACH SIGNAL'
      87            INUMSIG=1+IEXP
      88            DO 300,L=1,INUMSIG
1     89            READ(*,*) B(L),THETA1(L),VAR0(L),VAR1(L)
1     90            BOUND(L)=ALOG(MTIME*THETA1(L)**2/(2*VAR0(L)))
1     91 300        CONTINUE
      92            WRITE(*,*) 'ENTER ESTIMATE ,SPRM,SPRV FNAMES'
      93            READ(*,'(A)') FNAME3
      94            READ(*,'(A)') FNAME4
      95            READ(*,'(A)') FNAME5
      96 C          READ(*,'(A)') FNAME6
      97            IRECL5=4
      98            OPEN(22,FILE=FNAME3,STATUS='NEW',ACCESS='DIRECT',RECL=IRECL5)
      99            IRECL6=4*INUMSIG
     100            IRECL8=4*N3
     101            OPEN(35,FILE=FNAME4,STATUS='NEW',ACCESS='DIRECT',RECL=IRECL6)
     102            OPEN(40,FILE=FNAME5,STATUS='NEW',ACCESS='DIRECT',RECL=IRECL8)
     103 C          OPEN(50,FILE=FNAME6,STATUS='NEW',ACCESS='DIRECT',RECL=IRECL6)
     104            CALL CLS
     105            CALL PUTXY(IX,IY,IPAGE)
     106            IX=12
     107            IY=20
     108            CALL GOTOXY(IX,IY,IPAGE)
     109            WRITE(*,302)BOUNDA
     110 302        FORMAT(F7.2,\)
     111            IX=28
     112            IY=20
```

```
D Line# 1     7                          Microsoft FORTRAN77 V3.30 March 1985
     113          CALL GOTOXY(IX,IY,IPAGE)
     114          WRITE(*,304)BOUNDA
     115 304      FORMAT(F7.2,\)
     116          IX=44
     117          IY=20
     118          CALL GOTOXY(IX,IY,IPAGE)
     119          WRITE(*,306)BOUNDA
     120 306      FORMAT(F7.2,\)
     121          IX=12
     122          IY=22
     123          CALL GOTOXY(IX,IY,IPAGE)
     124          WRITE(*,308)BOUNDB
     125 308      FORMAT(F7.2,\)
     126          IX=28
     127          IY=22
     128          CALL GOTOXY(IX,IY,IPAGE)
     129          WRITE(*,310)BOUNDB
     130 310      FORMAT(F7.2,\)
     131          IX=44
     132          IY=22
     133          CALL GOTOXY(IX,IY,IPAGE)
     134          WRITE(*,312)BOUNDB
     135 312      FORMAT(F7.2,\)
     136 C
     137 C***     READ IN EACH SAMPLE AND VALIDATE SIGNAL AT EACH SAMPLE
     138 C
     139          xestmt=0
     140          DO 2000,I=1,ITOTAL
1    141 C        PAUSE
1    142          IK=NSKIP*(I-1)+1
1    143 C        WRITE(*,*) ' BOUND=',BOUND(1),BOUND(2),BOUND(3)
1    144 C        WRITE(50,REC=I) (BOUND(K),K=1,INUMSIG)
1    145          TIME=IK*DELTAT
1    146 C        WRITE(*,*) ' SAMPLE NUMBER=',IK
1    147          OPEN(10,FILE=FNAME1,STATUS='OLD',ACCESS='DIRECT',RECL=IRECL1)
1    148          READ(10,REC=IK) (X(J),J=1,N1)
1    149          CLOSE(10)
1    150          X1(IK)=X(CHAN1)
1    151 C        WRITE(*,*) 'ANALYTICAL SIGNAL=',X1(IK)
1    152          OPEN(20,FILE=FNAME2,STATUS='OLD',ACCESS='DIRECT',RECL=IRECL2)
1    153          READ(20,REC=IK) (Y(IL),IL=1,N2)
1    154          CLOSE(20)
1    155          DO 201,JK=1,IEXP
2    156          Y2(IK,JK)=Y(CHAN2(JK))
2    157 C        WRITE(*,*) 'EXP.SIGNAL NO.',JK,'=',Y2(IK,JK)
2    158 201      CONTINUE
1    159 C
1    160 C***     READ IN NOISE SIGNALS
1    161 C
1    162          IF (INOISE.EQ.1) THEN
1    163          OPEN(15,FILE=FNAME6,STATUS='OLD',ACCESS='DIRECT',RECL=IRECL7)
1    164          READ(15,REC=IK) (NOISE(IK,IT),IT=1,N3)
1    165          CLOSE(15)
1    166 C
1    167 C***     CHECK FOR NOISE DEGRADATIONS
1    168 C
```

```
D Line# 1     7                          Microsoft FORTRAN77 V3.30 March 1985
1   169           CALL SEQNOISE(NOISE,N3,BOUNDA,BOUNDB,VAR0,VAR1,IK,MTIME,SPRV1,I
1   170         * )
1   171 C         WRITE(*,*) 'SPRV=',SPRV1(1),SPRV1(2),SPRV1(3)
1   172           ENDIF
1   173 C
1   174 C***      SET UP A SINGLE ARRAY CONTAINING ALL SENSORS(EXCLUDING NOISE)
1   175 C
1   176           MI(1)=X1(IK)
1   177           DO 30,J=1,IEXP
2   178           MI(1+J)=Y2(IK,J)
2   179 30        CONTINUE
1   180 C
1   181 C***      ASSUME ALL WEIGHTS =1 FOR NOW
1   182 C***      DECISION ESTIMATION FOLLOWS
1   183 C
1   184           K=INUMSIG
1   185 400       CONTINUE
1   186 C
1   187 C***      FIND DEGREE OF INCONSISTENCY FOR K MEASUREMENTS
1   188 C
1   189           DO 450,J=1,K
2   190 450       INDEX(J)=0
1   191           DO 500,J=1,K
2   192           DO 550,L=1,K
3   193           E=ABS(MI(J)-MI(L))
3   194           B1=B(J)+B(L)
3   195           IF(E.GT.B1) THEN
3   196           INDEX(J)=INDEX(J)+1
3   197           ENDIF
3   198 550       CONTINUE
2   199 500       CONTINUE
1   200 C
1   201 C***      SET UP ARRAY CONTAINING SENSOR ORDER,NUMBER,ITS VALUE,AND INDEX
1   202 C
1   203           DO 750,J=1,K
2   204           M(J,1)=J
2   205           M(J,2)=J
2   206           M(J,3)=MI(J)
2   207           M(J,4)=INDEX(J)
2   208 750       CONTINUE
1   209 C
1   210 C***      TEST FOR ANY INCONSISTENT PAIRS
1   211 C
1   212           TEST=0
1   213           DO 600,J=1,K
2   214 600       TEST=TEST+INDEX(J)
1   215           IF(TEST.EQ.0) THEN
1   216           XESTMT=0
1   217           CALL ESTMAT(M,K,XESTMT,I)
1   218           IF(K.EQ.INUMSIG) THEN
1   219           IX=6
1   220           IY=4
1   221           CALL GOTOXY(IX,IY,IPAGE)
1   222           WRITE(*,605)
1   223 605       FORMAT( ' ALL SENSORS ARE USED IN ESTIMATION              ',\)
1   224           GOTO 200
```

```
D Line# 1      7                          Microsoft FORTRAN77 V3.30 March 1985
1   225            ELSE
1   226            GOTO 1600
1   227            ENDIF
1   228            ENDIF
1   229 700        CONTINUE
1   230 C
1   231 C***       ORGANIZE M(I,J) IN INCREASING ORDER OF MAGNITUDE FOR M(I,4)
1   232 C
1   233            JMIN=0
1   234            DO 800,L=1,K-1
2   235            JMIN=L
2   236            DO 900,J=L+1,K
3   237            IF(M(J,4).LT.M(JMIN,4)) JMIN=J
3   238 900        CONTINUE
2   239            TEMP1=M(L,4)
2   240            TEMP3=M(L,2)
2   241            TEMP4=M(L,3)
2   242            M(L,4)=M(JMIN,4)
2   243            M(L,2)=M(JMIN,2)
2   244            M(L,3)=M(JMIN,3)
2   245            M(JMIN,4)=TEMP1
2   246            M(JMIN,2)=TEMP3
2   247            M(JMIN,3)=TEMP4
2   248 800        CONTINUE
1   249 C          DO 850,J=1,K
1   250 C850        WRITE(*,*) M(J,1),M(J,2),M(J,3),M(J,4)
1   251 C
1   252 C***       FIND WITH MAX. AND MIN. INDEX
1   253 C***       FIND NUMBER OF MAX. INDICES
1   254 C
1   255            IMAX=M(K,4)
1   256            IMIN=M(1,4)
1   257            ICOUNT=0
1   258            DO 1000,L=K,1,-1
2   259            IF(M(L,4).EQ.IMAX) THEN
2   260            ICOUNT=ICOUNT+1
2   261            ELSE
2   262            GOTO 1100
2   263            ENDIF
2   264 1000       CONTINUE
1   265 1100       IF(IMAX.EQ.0) THEN
1   266            IX=6
1   267            IY=4
1   268            CALL GOTOXY(IX,IY,IPAGE)
1   269            WRITE(*,1105)K
1   270 1105       FORMAT( ' NUMBER   OF   CONSISTENT    MEASUREMENTS  = ',I4,\)
1   271            IX=1
1   272            IY=0
1   273            CALL GOTOXY(IX,IY,IPAGE)
1   274            PAUSE
1   275            IX=1
1   276            IY=0
1   277            CALL GOTOXY(IX,IY,IPAGE)
1   278            WRITE(*,1106)
1   279 1106       FORMAT( ' TABLE UPDATED AT ALL SAMPLES & FAILURES',\)
1   280            IX=0
```

```
D Line# 1     7                          Microsoft FORTRAN77 V3.30 March 1985
1   281           IY=1
1   282           CALL GOTOXY(IX,IY,IPAGE)
1   283           WRITE(*,1107)
1   284 1107      FORMAT( ' FAILURES INDICATED BY FLASHING COLUMN HEAD',\)
1   285           CALL ESTMAT(M,K,XESTMT,I)
1   286           IF(K.EQ.INUMSIG) THEN
1   287           IX=6
1   288           IY=4
1   289           CALL GOTOXY(IX,IY,IPAGE)
1   290           WRITE(*,1110)
1   291 1110      FORMAT( ' ALL SENSORS ARE USED IN ESTIMATION               ',\)
1   292           ELSE
1   293           GOTO 1600
1   294           ENDIF
1   295           ELSEIF(IMIN.EQ.0) THEN
1   296           IX=6
1   297           IY=4
1   298           CALL GOTOXY(IX,IY,IPAGE)
1   299           WRITE(*,1115)K
1   300 1115      FORMAT( ' NUMBER OF PARTIALLY CONSISTENT MEASUREMENTS=',I4,\)
1   301           IX=1
1   302           IY=0
1   303           CALL GOTOXY(IX,IY,IPAGE)
1   304           PAUSE
1   305           IX=1
1   306           IY=0
1   307           CALL GOTOXY(IX,IY,IPAGE)
1   308           WRITE(*,1116)
1   309 1116      FORMAT( ' TABLE UPDATED AT ALL SAMPLES & FAILURES',\)
1   310           IX=0
1   311           IY=1
1   312           CALL GOTOXY(IX,IY,IPAGE)
1   313           WRITE(*,1117)
1   314 1117      FORMAT( ' FAILURES INDICATED BY FLASHING COLUMN HEAD',\)
1   315           CALL ESTMAT(M,K,XESTMT,I)
1   316           IF(K.EQ.INUMSIG) THEN
1   317           IX=6
1   318           IY=4
1   319           CALL GOTOXY(IX,IY,IPAGE)
1   320           WRITE(*,1120)
1   321 1120      FORMAT( ' ALL SENSORS ARE USED IN ESTIMATION               ',\)
1   322           ELSE
1   323           GOTO 1600
1   324           ENDIF
1   325           ELSEIF(IMAX.EQ.K-1) THEN
1   326           IF(K.EQ.ICOUNT) THEN
1   327           IF(K.EQ.INUMSIG) THEN
1   328           IX=6
1   329           IY=4
1   330           CALL GOTOXY(IX,IY,IPAGE)
1   331           WRITE(*,1125)
1   332 1125      FORMAT( ' ALL MEASUREMENTS ARE INCONSISTENT                ',\)
1   333           IX=1
1   334           IY=0
1   335           CALL GOTOXY(IX,IY,IPAGE)
1   336           PAUSE
```

```
D Line# 1      7                          Microsoft FORTRAN77 V3.30 March 1985
1   337            IX=1
1   338            IY=0
1   339            CALL GOTOXY(IX,IY,IPAGE)
1   340            WRITE(*,1126)
1   341 1126       FORMAT( ' TABLE UPDATED AT ALL SAMPLES & FAILURES',\)
1   342            IX=0
1   343            IY=1
1   344            CALL GOTOXY(IX,IY,IPAGE)
1   345            WRITE(*,1127)
1   346 1127       FORMAT( ' FAILURES INDICATED BY FLASHING COLUMN HEAD',\)
1   347 C
1   348 C***       CHECK EACH OF K MEASUREMENTS WITH PAST VALUE P.FIND J MEASURE-
1   349 C***       MENTS FOR WHICH :M(I,3)-P:<=B(I),I=1,2,...
1   350 C
1   351            CALL PARITY(M,INUMSIG,B,K,P,N,MP,XESTMT)
1   352            GOTO 1500
1   353            ELSE
1   354            IX=6
1   355            IY=4
1   356            CALL GOTOXY(IX,IY,IPAGE)
1   357            WRITE(*,1130)
1   358 1130       FORMAT( ' POSSIBLE COMMON MODE FAILURE                    ',\)
1   359            IX=1
1   360            IY=0
1   361            CALL GOTOXY(IX,IY,IPAGE)
1   362            PAUSE
1   363            IX=1
1   364            IY=0
1   365            CALL GOTOXY(IX,IY,IPAGE)
1   366            WRITE(*,1135)
1   367 1135       FORMAT( ' TABLE UPDATED AT ALL SAMPLES & FAILURES',\)
1   368            IX=0
1   369            IY=1
1   370            CALL GOTOXY(IX,IY,IPAGE)
1   371            WRITE(*,1140)
1   372 1140       FORMAT( ' FAILURES INDICATED BY FLASHING COLUMN HEAD',\)
1   373            CALL PARITY(M,INUMSIG,B,K,P,N,MP,XESTMT)
1   374            ENDIF
1   375            ELSE
1   376            K=K-ICOUNT
1   377            DO 1200,L=1,K
2   378            M(L,4)=M(L,4)-ICOUNT
2   379 1200       CONTINUE
1   380            GOTO 700
1   381            ENDIF
1   382            ELSEIF(ICOUNT.EQ.1) THEN
1   383            K=K-ICOUNT
1   384            GOTO 400
1   385            ELSEIF(K.EQ.ICOUNT) THEN
1   386            IX=6
1   387            IY=4
1   388            CALL GOTOXY(IX,IY,IPAGE)
1   389            WRITE(*,1205)
1   390 1205       FORMAT( ' POSSIBLE COMMON MODE FAILURE                    ',\)
1   391            IX=1
1   392            IY=0
```

```
D Line# 1      7                          Microsoft FORTRAN77 V3.30 March 1985
1    393            CALL GOTOXY(IX,IY,IPAGE)
1    394            PAUSE
1    395            IX=1
1    396            IY=0
1    397            CALL GOTOXY(IX,IY,IPAGE)
1    398            WRITE(*,1210)
1    399 1210       FORMAT( ' TABLE UPDATED AT ALL SAMPLES & FAILURES',\)
1    400            IX=0
1    401            IY=1
1    402            CALL GOTOXY(IX,IY,IPAGE)
1    403            WRITE(*,1215)
1    404 1215       FORMAT( ' FAILURES INDICATED BY FLASHING COLUMN HEAD',\)
1    405            CALL PARITY(M,INUMSIG,B,K,P,N,MP,XESTMT)
1    406            GOTO 1500
1    407            ELSE
1    408            K=K-ICOUNT
1    409            GOTO 400
1    410            ENDIF
1    411            GOTO 1800
1    412 1500       CONTINUE
1    413            IF(JP.EQ.0) THEN
1    414            IX=6
1    415            IY=4
1    416            CALL GOTOXY(IX,IY,IPAGE)
1    417            WRITE(*,1505)
1    418 1505       FORMAT( ' NO ESTIMATE                                    ',\)
1    419            IX=1
1    420            IY=0
1    421            CALL GOTOXY(IX,IY,IPAGE)
1    422            PAUSE
1    423            IX=1
1    424            IY=0
1    425            CALL GOTOXY(IX,IY,IPAGE)
1    426            WRITE(*,1510)
1    427 1510       FORMAT( ' TABLE UPDATED AT ALL SAMPLES & FAILURES',\)
1    428            IX=0
1    429            IY=1
1    430            CALL GOTOXY(IX,IY,IPAGE)
1    431            WRITE(*,1515)
1    432 1515       FORMAT( ' FAILURES INDICATED BY FLASHING COLUMN HEAD',\)
1    433            GOTO 1800
1    434            ELSE
1    435 C
1    436 C***       CALCULATE AN ESTIMATE BASED ON PREVIOUS VALUES
1    437 C***       LET WEIGHT =1 FOR NOW
1    438 C
1    439            W=1
1    440            SUM1=0
1    441            SUM2=0
1    442            DO 1400,L=1,JP
2    443            SUM1=W*MP(N(L))+SUM1
2    444            SUM2=SUM2+W
2    445 1400       CONTINUE
1    446            ENDIF
1    447            XESTMT=SUM1/SUM2
1    448            IX=6
```

```
D Line# 1      7                          Microsoft FORTRAN77 V3.30 March 1985
1   449            IY=4
1   450            CALL GOTOXY(IX,IY,IPAGE)
1   451            WRITE(*,1405)XESTMT
1   452 1405       FORMAT( ' ESTIMATE  BASED  ON  PREVIOUS  VALUES  = ',F7.2,\)
1   453            IX=1
1   454            IY=0
1   455            CALL GOTOXY(IX,IY,IPAGE)
1   456            PAUSE
1   457            IX=1
1   458            IY=0
1   459            CALL GOTOXY(IX,IY,IPAGE)
1   460            WRITE(*,1410)
1   461 1410       FORMAT( ' TABLE UPDATED AT ALL SAMPLES & FAILURES',\)
1   462            IX=0
1   463            IY=1
1   464            CALL GOTOXY(IX,IY,IPAGE)
1   465            WRITE(*,1415)
1   466 1415       FORMAT( ' FAILURES INDICATED BY FLASHING COLUMN HEAD',\)
1   467            K=JP
1   468 1600       CONTINUE
1   469            NINCNST=INUMSIG-K
1   470            IX=6
1   471            IY=4
1   472            CALL GOTOXY(IX,IY,IPAGE)
1   473            WRITE(*,1605)NINCNST
1   474 1605       FORMAT( ' NUMBER OF REMAINING  INCONSISTENT  MEASURM.=',I4,\)
1   475            IX=1
1   476            IY=0
1   477            CALL GOTOXY(IX,IY,IPAGE)
1   478            PAUSE
1   479            IX=1
1   480            IY=0
1   481            CALL GOTOXY(IX,IY,IPAGE)
1   482            WRITE(*,1607)
1   483 1607       FORMAT( ' TABLE UPDATED AT ALL SAMPLES & FAILURES',\)
1   484            IX=0
1   485            IY=1
1   486            CALL GOTOXY(IX,IY,IPAGE)
1   487            WRITE(*,1609)
1   488 1609       FORMAT( ' FAILURES INDICATED BY FLASHING COLUMN HEAD',\)
1   489 C
1   490 C***       CHECK IF SENSOR VALUE IS HIGH OR LOW COMPARED TO ESTIMATE
1   491 C
1   492            DO 1700,J=K+1,INUMSIG
2   493            IF(M(J,3).GT.XESTMT) THEN
2   494            IX=6
2   495            IY=4
2   496            CALL GOTOXY(IX,IY,IPAGE)
2   497            WRITE(*,1610)M(J,2)
2   498 1610       FORMAT( ' SIGNAL NUMBER',F7.2,' READS HIGH                    ',\)
2   499            IX=1
2   500            IY=0
2   501            CALL GOTOXY(IX,IY,IPAGE)
2   502            PAUSE
2   503            IX=1
2   504            IY=0
```

```
D Line# 1     7                          Microsoft FORTRAN77 V3.30 March 1985
2   505           CALL GOTOXY(IX,IY,IPAGE)
2   506           WRITE(*,1615)
2   507 1615      FORMAT( ' TABLE UPDATED AT ALL SAMPLES & FAILURES',\)
2   508           IX=0
2   509           IY=1
2   510           CALL GOTOXY(IX,IY,IPAGE)
2   511           WRITE(*,1620)
2   512 1620      FORMAT( ' FAILURES INDICATED BY FLASHING COLUMN HEAD',\)
2   513 C         WRITE(*,*) ' ITS VALUE =',M(J,3)
2   514           ENDIF
2   515 1700      CONTINUE
1   516           DO 1750,J=K+1,INUMSIG
2   517           IF(M(J,3).LT.XESTM) THEN
2   518           IX=6
2   519           IY=4
2   520           CALL GOTOXY(IX,IY,IPAGE)
2   521           WRITE(*,1705)M(J,2)
2   522 1705      FORMAT( ' SIGNAL NUMBER',F7.2,' READS LOW                    ',\)
2   523           IX=1
2   524           IY=0
2   525           CALL GOTOXY(IX,IY,IPAGE)
2   526           PAUSE
2   527           IX=1
2   528           IY=0
2   529           CALL GOTOXY(IX,IY,IPAGE)
2   530           WRITE(*,1710)
2   531 1710      FORMAT( ' TABLE UPDATED AT ALL SAMPLES & FAILURES',\)
2   532           IX=0
2   533           IY=1
2   534           CALL GOTOXY(IX,IY,IPAGE)
2   535           WRITE(*,1715)
2   536 1715      FORMAT( ' FAILURES INDICATED BY FLASHING COLUMN HEAD',\)
2   537 C         WRITE(*,*) ' ITS VALUE =',M(J,3)
2   538           ENDIF
2   539 1750      CONTINUE
1   540 C
1   541 C***      SET UP AN ARRAY OF PAST VALUES USING CURRENT VALUES
1   542 C
1   543 1800      DO 1900,L=1,INUMSIG
2   544           N(L)=NINT(M(L,2))
2   545           P(N(L),1)=M(N(L),1)
2   546           P(N(L),2)=M(N(L),2)
2   547           P(N(L),3)=M(N(L),3)
2   548           P(N(L),4)=0
2   549 1900      CONTINUE
1   550 C
1   551 C***      CALL SUBROUTINE SEQUENTIAL
1   552 C
1   553 200       CALL SEQUENTIAL(SPRM,SPRV1,INUMSIG,M,MTIME,XESTMT,B,TIME,VAR0,
1   554          *VAR1,BOUND,THETA1,I)
1   555           IX=12
1   556           IY=16
1   557           CALL GOTOXY(IX,IY,IPAGE)
1   558           WRITE(*,215)SPRV1(1)
1   559 215       FORMAT(F7.2,\)
1   560           IX=28
```

```
D Line# 1      7
1   561            IY=16
1   562            CALL GOTOXY(IX,IY,IPAGE)
1   563            WRITE(*,217)SPRV1(2)
1   564 217        FORMAT(F7.2,\)
1   565            IX=44
1   566            IY=16
1   567            CALL GOTOXY(IX,IY,IPAGE)
1   568            WRITE(*,219)SPRV1(3)
1   569 219        FORMAT(F7.2,\)
1   570            IX=12
1   571            IY=18
1   572            CALL GOTOXY(IX,IY,IPAGE)
1   573            WRITE(*,221)BOUND(1)
1   574 221        FORMAT(F7.2,\)
1   575            IX=28
1   576            IY=18
1   577            CALL GOTOXY(IX,IY,IPAGE)
1   578            WRITE(*,223)BOUND(2)
1   579 223        FORMAT(F7.2,\)
1   580            IX=44
1   581            IY=18
1   582            CALL GOTOXY(IX,IY,IPAGE)
1   583            WRITE(*,225)BOUND(3)
1   584 225        FORMAT(F7.2,\)
1   585            IX=12
1   586            IY=24
1   587            CALL GOTOXY(IX,IY,IPAGE)
1   588            WRITE(*,227)M(1,4)
1   589 227        FORMAT(F7.2,\)
1   590            IX=28
1   591            IY=24
1   592            CALL GOTOXY(IX,IY,IPAGE)
1   593            WRITE(*,229)M(2,4)
1   594 229        FORMAT(F7.2,\)
1   595            IX=44
1   596            IY=24
1   597            CALL GOTOXY(IX,IY,IPAGE)
1   598            WRITE(*,231)M(3,4)
1   599 231        FORMAT(F7.2,\)
1   600            IX=20
1   601            IY=7
1   602            CALL GOTOXY(IX,IY,IPAGE)
1   603            WRITE(*,233)XESTMT
1   604 233        FORMAT(F7.2,\)
1   605            IX=42
1   606            IY=7
1   607            CALL GOTOXY(IX,IY,IPAGE)
1   608            WRITE(*,235)IK
1   609 235        FORMAT(2X,I4,\)
1   610 2000       CONTINUE
    611            CLOSE(22)
    612            CLOSE(35)
    613            CLOSE(40)
    614            CLOSE(50)
    615            END
```

D Line# 1 7 Microsoft FORTRAN77 V3.30 March 1985

| Name | Type | Offset | P | Class |
|---|---|---|---|---|
| ABS | | | | INTRINSIC |
| ALPHA | REAL | 44566 | | |
| B | REAL | 44306 | | |
| B1 | REAL | 44910 | | |
| BETA | REAL | 44570 | | |
| BOUND | REAL | 44416 | | |
| BOUNDA | REAL | 44578 | | |
| BOUNDB | REAL | 44574 | | |
| CHAN1 | INTEGER*2 | 44510 | | |
| CHAN2 | INTEGER*2 | 44436 | | |
| DELTAT | REAL | 44448 | | |
| E | REAL | 44906 | | |
| FALSE | REAL | ***** | | |
| FNAME1 | CHAR*40 | 44465 | | |
| FNAME2 | CHAR*40 | 44514 | | |
| FNAME3 | CHAR*40 | 44646 | | |
| FNAME4 | CHAR*40 | 44686 | | |
| FNAME5 | CHAR*40 | 44726 | | |
| FNAME6 | CHAR*40 | 44582 | | |
| I | INTEGER*2 | 44446 | | |
| ICOUNT | INTEGER*2 | 45000 | | |
| IEXP | INTEGER*2 | 44558 | | |
| IK | INTEGER*2 | 44882 | | |
| IL | INTEGER*2 | 44890 | | |
| IMAX | INTEGER*2 | 44996 | | |
| IMIN | INTEGER*2 | 44998 | | |
| INDEX | INTEGER*2 | 44406 | | |
| INOISE | INTEGER*2 | 44564 | | |
| INUMSI | INTEGER*2 | 44638 | | |
| IPAGE | INTEGER*2 | 44776 | | |
| IRECL1 | INTEGER*2 | 44508 | | |
| IRECL2 | INTEGER*2 | 44556 | | |
| IRECL5 | INTEGER*2 | 44766 | | |
| IRECL6 | INTEGER*2 | 44768 | | |
| IRECL7 | INTEGER*2 | 44624 | | |
| IRECL8 | INTEGER*2 | 44770 | | |
| IT | INTEGER*2 | 44896 | | |
| ITOTAL | INTEGER*2 | 44630 | | |
| IX | INTEGER*2 | 44772 | | |
| IY | INTEGER*2 | 44774 | | |
| J | INTEGER*2 | 44560 | | |
| JK | INTEGER*2 | 44892 | | |
| JMIN | INTEGER*2 | 44978 | | |
| JP | INTEGER*2 | 45902 | | |
| K | INTEGER*2 | 44636 | | |
| L | INTEGER*2 | 44642 | | |
| LOG | | | | INTRINSIC |
| M | REAL | 44326 | | |
| MI | REAL | 16 | | |
| MISS | REAL | ***** | | |
| MP | REAL | 24076 | | |
| MTIME | REAL | 44632 | | |
| N | INTEGER*2 | 44296 | | |

```
D Line# 1     7
N1       INTEGER*2      44506
N2       INTEGER*2      44554
N3       INTEGER*2      44622
NINCNS   INTEGER*2      46230
NINT                            INTRINSIC
NOISE    REAL           24296
NPOINT   INTEGER*2      44512
NSAMPL   INTEGER*2      44626
NSKIP    INTEGER*2      44628
P        REAL           24096
SPRM     REAL           24236
SPRV1    REAL           24256
SPRV2    REAL           24276
SUM1     REAL           46064
SUM2     REAL           46068
TEMP1    REAL           44984
TEMP3    REAL           44988
TEMP4    REAL           44992
TEST     REAL           44916
THETA1   REAL           24216
TIME     REAL           44884
VAR0     REAL           24176
VAR1     REAL           24196
W        REAL           46060
X        REAL              36
X1       REAL           20076
XESTM    REAL           46562
XESTMT   REAL           44876
Y        REAL              56
Y2       REAL              76


     616 C
     617 C***      SUBROUTINES USED IN MAIN PROGRAM
     618 C
     619           SUBROUTINE ESTMAT(M,K,XESTMT,I)
     620           REAL*4 M(5,4),XESTMT,SUM1,SUM2
     621           INTEGER*2 K
     622           SUM1=0
     623           SUM2=0
     624           DO 10,J=1,K
1    625           SUM1=SUM1+(K-1-M(J,4))*M(J,3)
1    626           SUM2=SUM2+(K-1-M(J,4))
1    627 10        CONTINUE
     628           XESTMT=SUM1/SUM2
     629 C         WRITE(*,*) 'ESTIMATE=',XESTMT
     630           WRITE(22,REC=I) XESTMT
     631           RETURN
     632           END

Name     Type         Offset P Class

I        INTEGER*2        12 *
J        INTEGER*2     46830
K        INTEGER*2         4 *
M        REAL              0 *
```

```
D Line# 1     7                          Microsoft FORTRAN77 V3.30 March 1985
SUM1   REAL           46822
SUM2   REAL           46826
XESTMT REAL               8 *


    633 C
    634 C***      SUBROUTINE PARITY
    635 C
    636           SUBROUTINE PARITY(M,INUMSIG,B,K,P,N,MP,XESTMT)
    637           REAL*4 M(5,4),B(5),P(5,4),MP(5),XESTMT
    638           REAL*4 X(5),L,MTEMP(5,4)
    639           INTEGER*2 K,I,JP,T,INUMSIG,N(5)
    640           DO 10,I=1,K
1   641           N(I)=NINT(M(I,2))
1   642 10        CONTINUE
    643           JP=0
    644           DO 100,I=1,K
1   645           X(N(I))=ABS(M(N(I),3)-XESTMT)
1   646           IF(X(N(I)).LE.B(N(I))) THEN
1   647           MP(N(I))=M(N(I),3)
1   648           JP=JP+1
1   649           ELSE
1   650           IX=6
1   651           IY=4
1   652           CALL GOTOXY(IX,IY,IPAGE)
1   653           WRITE(*,12)N(I)
1   654 12        FORMAT( ' SIGNAL NO.',I4,' IS INCONSIST. WITH PAST VALUE
1   655       *    ',\)
1   656           IX=1
1   657           IY=0
1   658           CALL GOTOXY(IX,IY,IPAGE)
1   659           PAUSE
1   660           IX=1
1   661           IY=0
1   662           CALL GOTOXY(IX,IY,IPAGE)
1   663           WRITE(*,14)
1   664 14        FORMAT( ' TABLE UPDATED AT ALL SAMPLES & FAILURES',\)
1   665           IX=0
1   666           IY=1
1   667           CALL GOTOXY(IX,IY,IPAGE)
1   668           WRITE(*,13)
1   669 13        FORMAT( ' FAILURES INDICATED BY FLASHING COLUMN HEAD',\)
1   670 20        FORMAT(6I6)
1   671 C         WRITE(*,*) ' ITS VALUE =',M(N(I),3)
1   672 15        FORMAT(F10.5)
1   673           ENDIF
1   674 100       CONTINUE
    675           RETURN
    676           END

Name    Type         Offset P Class

ABS                          INTRINSIC
B       REAL              8 *
I       INTEGER*2     46934
INUMSI  INTEGER*2         4 *
```

```
D Line# 1     7                          Microsoft FORTRAN77 V3.30 March 1985
IPAGE   INTEGER*2     46946
IX      INTEGER*2     46942
IY      INTEGER*2     46944
JP      INTEGER*2     46938
K       INTEGER*2        12 *
L       REAL          *****
M       REAL             0 *
MP      REAL            24 *
MTEMP   REAL          46854
N       INTEGER*2        20 *
NINT                          INTRINSIC
P       REAL             16 *
T       INTEGER*2     *****
X       REAL          46834
XESTMT  REAL             28 *


    677 C
    678 C***      SUBROUTINE TO PERFORM SEQUENTIAL TESTING( BIAS DEGRADATIONS)
    679 C
    680           SUBROUTINE SEQUENTIAL(SPRM,SPRV1,INUMSIG,M,MTIME,XESTMT,B,TIME,
    681          *VAR0,VAR1,BOUND,THETA1,I)
    682           REAL*4 M(5,4),SPRM(5),THETA1(5),MTIME,TIME,B(5)
    683           REAL*4 XUSED1(5),VAR0(5),SPRMNEW(5),BOUND(5)
    684           REAL*4 VAR1(5)
    685           INTEGER*2 MIN(5)
    686 C
    687 C***      SET UP ARRAY CONTAINING M(I,3)-XESTMT AND M(I,2)
    688 C***      CONVERT M(I,2) TO INTEGER VALUES
    689           DO 10,J=1,INUMSIG
1   690           MIN(J)=NINT(M(J,2))
1   691           XUSED1(MIN(J))=ABS(M(MIN(J),3)-XESTMT)
1   692 10        CONTINUE
    693 C
    694 C***      CALCULATE SPRM
    695 C
    696           DO 30,K=1,INUMSIG
1   697           SPRMNEW(MIN(K))=SPRM(MIN(K))+THETA1(MIN(K))*(XUSED1(MIN(K))-THE
1   698          *   TA1(MIN(K))/2.)/VAR0(MIN(K))
1   699           SPRM(MIN(K))=SPRMNEW(MIN(K))
1   700 30        CONTINUE
    701           WRITE(35,REC=I) (SPRM(MIN(L)),L=1,INUMSIG)
    702           IX=12
    703           IY=12
    704           CALL GOTOXY(IX,IY,IPAGE)
    705           WRITE(*,203)M(MIN(1),3)
    706 203       FORMAT(F7.2,\)
    707           IX=28
    708           IY=12
    709           CALL GOTOXY(IX,IY,IPAGE)
    710           WRITE(*,205)M(MIN(2),3)
    711 205       FORMAT(F7.2,\)
    712           IX=44
    713           IY=12
    714           CALL GOTOXY(IX,IY,IPAGE)
    715           WRITE(*,207)M(MIN(3),3)
```

```
D Line# 1      7                          Microsoft FORTRAN77 V3.30 March 1985
     716 207       FORMAT(F7.2,\)
     717           IX=12
     718           IY=14
     719           CALL GOTOXY(IX,IY,IPAGE)
     720           WRITE(*,209)SPRM(MIN(1))
     721 209       FORMAT(F7.2,\)
     722           IX=28
     723           IY=14
     724           CALL GOTOXY(IX,IY,IPAGE)
     725           WRITE(*,211)SPRM(MIN(2))
     726 211       FORMAT(F7.2,\)
     727           IX=44
     728           IY=14
     729           CALL GOTOXY(IX,IY,IPAGE)
     730           WRITE(*,213)SPRM(MIN(3))
     731 213       FORMAT(F7.2,\)
     732 C
     733 C***       TEST IF DECITION BOUND REACHED BY SPRM
     734 C
     735           DO 40,K=1,INUMSIG
1    736           IF(SPRM(MIN(K)).LE.0) THEN
1    737           SPRM(MIN(K))=0
1    738           ELSEIF(SPRM(MIN(K)).GE.BOUND(MIN(K))) THEN
1    739           IX=16
1    740           IY=5
1    741           CALL GOTOXY(IX,IY,IPAGE)
1    742           WRITE(*,14)M(MIN(K),2)
1    743 14        FORMAT(F7.2,\)
1    744           IX=33
1    745           IY=5
1    746           CALL GOTOXY(IX,IY,IPAGE)
1    747           WRITE(*,16)
1    748 16        FORMAT( ' EXPERIENCED A BIAS FAILURE ',\)
1    749           IX=1
1    750           IY=0
1    751           CALL GOTOXY(IX,IY,IPAGE)
1    752           PAUSE
1    753           IX=1
1    754           IY=0
1    755           CALL GOTOXY(IX,IY,IPAGE)
1    756           WRITE(*,17)
1    757 17        FORMAT( ' TABLE UPDATED AT ALL SAMPLES & FAILURES',\)
1    758           IX=0
1    759           IY=1
1    760           CALL GOTOXY(IX,IY,IPAGE)
1    761           WRITE(*,19)
1    762 19        FORMAT( ' FAILURES INDICATED BY FLASHING COLUMN HEAD',\)
1    763 C         PAUSE
1    764           SPRM(MIN(K))=0
1    765           ENDIF
1    766           IF(TIME.GT.MTIME) THEN
1    767           SPRM(MIN(K))=0
1    768           ENDIF
1    769 40        CONTINUE
     770           RETURN
     771           END
```

D Line# 1 7 Microsoft FORTRAN77 V3.30 March 1985

| Name | Type | Offset | P | Class |
|---|---|---|---|---|
| ABS | | | | INTRINSIC |
| B | REAL | 24 | * | |
| BOUND | REAL | 40 | * | |
| I | INTEGER*2 | 48 | * | |
| INUMSI | INTEGER*2 | 8 | * | |
| IPAGE | INTEGER*2 | 47246 | | |
| IX | INTEGER*2 | 47242 | | |
| IY | INTEGER*2 | 47244 | | |
| J | INTEGER*2 | 47230 | | |
| K | INTEGER*2 | 47234 | | |
| L | INTEGER*2 | 47238 | | |
| M | REAL | 12 | * | |
| MIN | INTEGER*2 | 47220 | | |
| MTIME | REAL | 16 | * | |
| NINT | | | | INTRINSIC |
| SPRM | REAL | 0 | * | |
| SPRMNE | REAL | 47200 | | |
| SPRV1 | REAL | 4 | * | |
| THETA1 | REAL | 44 | * | |
| TIME | REAL | 28 | * | |
| VAR0 | REAL | 32 | * | |
| VAR1 | REAL | 36 | * | |
| XESTMT | REAL | 20 | * | |
| XUSED1 | REAL | 47180 | | |

```
      772 C
      773 C***      SUBROUTINE TO CHECK FOR NOISE DEGRADATIONS
      774 C
      775           SUBROUTINE SEQNOISE(NOISE,N3,BOUNDA,BOUNDB,VAR0,VAR1,IK,MTIME,
      776         *  SPRV1,I)
      777           REAL*4 NOISE(1000,5),SPRV1(5),SPRVN(5),VAR0(5),VAR1(5),BOUNDA
      778           REAL*4 MTIME,BOUNDB
      779 C
      780 C***      CALCULATE SPRV
      781 C
      782           DO 10,L=1,N3
 1    783           SPRVN(L)=SPRV1(L)+.5*(1/VAR0(L)-1/VAR1(L))*NOISE(IK,L)**2+.5*
 1    784         *  LOG(VAR0(L)/VAR1(L))
 1    785           SPRV1(L)=SPRVN(L)
 1    786 10        CONTINUE
      787           WRITE(40,REC=I) (SPRV1(J),J=1,N3)
      788 C
      789 C***      CHECK IF BOUND REACHED BY EACH SIGNAL
      790 C
      791           DO 20,L=1,N3
 1    792           IF(SPRV1(L).LE.BOUNDA) THEN
 1    793           SPRV1(L)=0.
 1    794           ELSEIF(SPRV1(L).GE.BOUNDB) THEN
 1    795           IX=16
 1    796           IY=5
 1    797           CALL GOTOXY(IX,IY,IPAGE)
 1    798           WRITE(*,28)L
```

```
D Line# 1     7                          Microsoft FORTRAN77 V3.30 March 1985
1   799 28        FORMAT(I4,\)
1   800           IX=33
1   801           IY=5
1   802           CALL GOTOXY(IX,IY,IPAGE)
1   803           WRITE(*,26)
1   804 26        FORMAT( ' EXPERIENCED A NOISE FAILURE',\)
1   805           IX=1
1   806           IY=0
1   807           CALL GOTOXY(IX,IY,IPAGE)
1   808           PAUSE
1   809           IX=1
1   810           IY=0
1   811           CALL GOTOXY(IX,IY,IPAGE)
1   812           WRITE(*,24)
1   813 24        FORMAT( ' TABLE UPDATED AT ALL SAMPLES & FAILURES',\)
1   814           IX=0
1   815           IY=1
1   816           CALL GOTOXY(IX,IY,IPAGE)
1   817           WRITE(*,22)
1   818 22        FORMAT( ' FAILURES INDICATED BY FLASHING COLUMN HEAD',\)
1   819 C         PAUSE
1   820           DO 30,J=1,N3
2   821 30        SPRV1(J)=0
1   822           ENDIF
1   823 20        CONTINUE
    824           RETURN
    825           END

Name     Type         Offset P Class

BOUNDA REAL              8 *
BOUNDB REAL             12 *
I      INTEGER*2        36 *
IK     INTEGER*2        24 *
IPAGE  INTEGER*2     47532
IX     INTEGER*2     47528
IY     INTEGER*2     47530
J      INTEGER*2     47522
L      INTEGER*2     47518
LOG                        INTRINSIC
MTIME  REAL             28 *
N3     INTEGER*2         4 *
NOISE  REAL              0 *
SPRV1  REAL             32 *
SPRVN  REAL          47498
VAR0   REAL             16 *
VAR1   REAL             20 *


    826 C
    827 C***      SUBROUTINE PUTXY
    828 C
    829       SUBROUTINE PUTXY(IX,IY,IPAGE)
    830       INTEGER*2 IX,IY,IPAGE
    831 C     REAL Z
    832       IPAGE=0
```

```
D Line# 1      7                          Microsoft FORTRAN77 V3.30 March 1985
    833 C      CALL CLS
    834        IX=1
    835        IY=0
    836        CALL GOTOXY(IX,IY,IPAGE)
    837        WRITE(*,20)
    838 20     FORMAT( ' TABLE UPDATED AT ALL  SAMPLES & FAILURES',\)
    839        IX=0
    840        IY=1
    841        CALL GOTOXY(IX,IY,IPAGE)
    842        WRITE(*,25)
    843 25     FORMAT( ' FAILURES INDICATED BY FLASHING COLUMN HEAD',\)
    844        IX=0
    845        IY=3
    846        CALL GOTOXY(IX,IY,IPAGE)
    847        WRITE(*,30)
    848 30     FORMAT( ' FAILURE INFORMATION:',\)
    849 C      IX=6
    850 C      IY=4
    851 C      CALL GOTOXY(IX,IY,IPAGE)
    852 C      WRITE(*,35)
    853 C35    FORMAT( ' "CONSISTENCY MESSAGE"',\)
    854        IX=7
    855        IY=5
    856        CALL GOTOXY(IX,IY,IPAGE)
    857        WRITE(*,40)
    858 40     FORMAT( ' SENSOR #:         ,  TYPE:',\)
    859        IX=0
    860        IY=7
    861        CALL GOTOXY(IX,IY,IPAGE)
    862        WRITE(*,45)
    863 45     FORMAT( ' ESTIMATE (DEG. F) =         ,   SAMPLE # =         ',\)
    864        IX=12
    865        IY=10
    866        CALL GOTOXY(IX,IY,IPAGE)
    867        WRITE(*,50)
    868 50     FORMAT( ' SIGNAL#1        SIGNAL#2        SIGNAL#3',\)
    869        IX=0
    870        IY=12
    871        CALL GOTOXY(IX,IY,IPAGE)
    872        WRITE(*,60)
    873 60     FORMAT( ' VALUE ',\)
    874        IX=0
    875        IY=14
    876        CALL GOTOXY(IX,IY,IPAGE)
    877        WRITE(*,65)
    878 65     FORMAT( ' SPRM  ',\)
    879        IX=0
    880        IY=16
    881        CALL GOTOXY(IX,IY,IPAGE)
    882        WRITE(*,70)
    883 70     FORMAT( ' SPRV  ',\)
    884        IX=0
    885        IY=18
    886        CALL GOTOXY(IX,IY,IPAGE)
    887        WRITE(*,75)
    888 75     FORMAT( ' BOUND ',\)
```

```
D Line# 1      7
    889        IX=0
    890        IY=20
    891        CALL GOTOXY(IX,IY,IPAGE)
    892        WRITE(*,80)
    893 80     FORMAT( ' BOUNDA ',\)
    894        IX=0
    895        IY=22
    896        CALL GOTOXY(IX,IY,IPAGE)
    897        WRITE(*,82)
    898 82     FORMAT( ' BOUNDB ',\)
    899        IX=0
    900        IY=24
    901        CALL GOTOXY(IX,IY,IPAGE)
    902        WRITE(*,84)
    903 84     FORMAT( ' INDEX ',\)
    904        IF(IY.LT.0)STOP
    905        WRITE(*,1010)
    906 1010   FORMAT(\)
    907 C      CALL GOTOXY(IX,IY,IPAGE)
    908 C      WRITE(*,1020)Z
    909 C1020  FORMAT(F7.2,\)
    910 C      IF(IY.EQ.0)THEN
    911 C        IY=23
    912 C      ELSE
    913 C        IY=9
    914 C      ENDIF
    915 C      IX=0
    916 C      WRITE(*,1010)
    917 C      CALL GOTOXY(IX,IY,IPAGE)
    918 C      GO TO 100
    919        RETURN
    920        END

Name     Type         Offset P Class

IPAGE    INTEGER*2         8 *
IX       INTEGER*2         0 *
IY       INTEGER*2         4 *


Name     Type           Size   Class

CLS                            SUBROUTINE
ESTMAT                         SUBROUTINE
GOTOXY                         SUBROUTINE
MAIN                           PROGRAM
PARITY                         SUBROUTINE
PUTXY                          SUBROUTINE
SEQNOI                         SUBROUTINE
SEQUEN                         SUBROUTINE


 Pass One     No Errors Detected
             920 Source Lines
```

# APPENDIX B

## LISTING FOR "DATAGET.FOR"

```
D Line# 1      7                          Microsoft FORTRAN77 V3.30 March 1985
      1 C***      THIS PROGRAM DATAGET
      2 C***      THIS PROGRAM USES FORTRAN COMPATIBLE PC LAB
      3 C***      FOR DATA A/D CONVERTION AND WRITES DATA TO A DISK
      4 $STORAGE:2
      5           CHARACTER*10 FNAME
      6           INTEGER ARRAY(25000),VALUE,STRTCHN,ENDCHN,GAIN,TIMSOR
      7           INTEGER SCNLNGTH,NUMSCN,FREQ,ERROR
      8           INTEGER*4 CLCKDIV
      9           VALUE=25001
     10 10        FORMAT(6I6)
     11 C***      CALL SETUP.ADC
     12 C***      ENTER TIMING.SOURSE,START.CHAN,END.CHAN,GAIN (INTEGERS)
     13 C
     14           TIMSOR=0
     15           WRITE(*,'(A/)') ' ENTER START CHAN,END CHAN,GAIN(INTEGERS)'
     16           WRITE(*,'(A/)') ' LEGAL GAINS 1,2,4,8'
     17           READ(*,10) STRTCHN,ENDCHN,GAIN
     18           CALL XSA(TIMSOR,STRTCHN,ENDCHN,GAIN)
     19           WRITE(*,'(A/)') ' ERROR='
     20           WRITE(*,*) ERROR
     21           SCNLNGTH=(ENDCHN+1)-STRTCHN
     22           NUMSCN=(VALUE/SCNLNGTH)-1
     23           WRITE(*,'(A/)') ' NUMBER OF SCANS='
     24           WRITE(*,10) NUMSCN
     25 C
     26 C***      SET INTERNAL CLOCK
     27 C
     28           WRITE(*,'(A/)') ' ENTER CLOCK FREQENCY (13-12000)HZ (INTEGER)'
     29           READ(*,10) FREQ
     30           CLCKDIV=(800000/FREQ)
     31           CALL XSCD(CLCKDIV)
     32 C
     33 C***      TURN OFF PC TIME OF DAY CLOCK
     34 C***      DISABLE.SYSTEM.CLOCK
     35 C
     36           CALL XDSC
     37 C
     38 C***      ACTUAL DATA COLLECTION (ADC.SERIES)
     39 C
     40           WRITE(*,'(A/)') ' STARTING DATA COLLECTION'
     41           CALL XAS(VALUE,ARRAY(0))
     42           WRITE(*,'(A/)') ' FINISHED'
     43 C***      SET TIME OF DAY CLOCK
     44           CALL XESC
     45 C
     46 C***      WRITE DATA TO DISK
     47 C
     48           WRITE(*,'(A/)') ' ENTER DATA FILE NAME'
     49           READ(*,'(A)') FNAME
     50           IRECL=2
     51           OPEN(10,FILE=FNAME,STATUS='NEW',ACCESS='DIRECT',RECL=IRECL)
     52           DO 30 I=1,VALUE+1
1    53            WRITE(10,REC=I) ARRAY(I-1)
1    54 30        CONTINUE
     55           CLOSE(10)
     56           STOP
```

```
D Line# 1      7                          Microsoft FORTRAN77 V3.30 March 1985
     57            END

Name     Type          Offset P Class

ARRAY   INTEGER*2         16
CLCKDI  INTEGER*4      50092
ENDCHN  INTEGER*2      50080
ERROR   INTEGER*2      50084
FNAME   CHAR*10        50096
FREQ    INTEGER*2      50090
GAIN    INTEGER*2      50082
I       INTEGER*2      50108
IRECL   INTEGER*2      50106
NUMSCN  INTEGER*2      50088
SCNLNG  INTEGER*2      50086
STRTCH  INTEGER*2      50078
TIMSOR  INTEGER*2      50076
VALUE   INTEGER*2      50016


      58
      59

Name     Type            Size    Class

MAIN                             PROGRAM
XAS                              SUBROUTINE
XDSC                             SUBROUTINE
XESC                             SUBROUTINE
XSA                              SUBROUTINE
XSCD                             SUBROUTINE


  Pass One     No Errors Detected
               59 Source Lines
```

# APPENDIX C

## LISTING FOR "GDATRAN.FOR"

```
D Line# 1     7                          Microsoft FORTRAN77 V3.30 March 1985
      1 C*************************************************************************C
      2 C****              PROGRAM GDATRAN                                      ****C
      3 C****    THIS PROGRAM TRANSFORMS THE DATA ACQUIRED FROM THE EXPT.       ****C
      4 C****    AIR-LOOP (INTEGER) INTO PHYSICAL VARIABLES WITH ACTUAL         ****C
      5 C****                 UNITS (F)                                         ****C
      6 C*************************************************************************C
      7 C
      8           INTEGER*2 ISTO(10),ICOUNT,ISIGF(10)
      9           DIMENSION GAIN(10),X1(10),XMV(3500),XTEMP(3500)
     10           CHARACTER*20  FILIN,FILOUT,FNAME
     11           WRITE(*,'(A/)') ' NO.OF SIGNALS IN A/D FILE?'
     12           READ(*,10) NSIGNL
     13     10    FORMAT(6I6)
     14           IRECL1=2*NSIGNL
     15           IRECL2=4*NSIGNL
     16           WRITE(*,'(A/)') ' NO.OF RECORDS TO TRANSFER?'
     17           READ(*,10) NREC
     18           WRITE(*,'(A/)') ' CHAN. NO. OF REF. JUNCTION SIGNAL'
     19           READ(*,10) NJTN
     20           WRITE(*,'(A/)') ' A/D DATA FILENAME(OLD)?'
     21           READ(*,'(A)') FILIN
     22           OPEN(11,FILE=FILIN,STATUS='OLD',ACCESS='DIRECT',RECL=IRECL1)
     23 C****     OPEN FILE FOR TC-TABLE.
     24           WRITE(*,'(A/)') ' TC-TABLE FILENAME?'
     25           READ(*,'(A)') FNAME
     26           OPEN(10,FILE=FNAME,STATUS='OLD',ACCESS='DIRECT',RECL=8)
     27 C****     OPEN FILE FOR TRANSFORMED DATA.
     28           WRITE(*,'(A/)') ' A/D DATA FILENAME(NEW)?'
     29           READ(*,'(A)') FILOUT
     30           OPEN(12,FILE=FILOUT,STATUS='NEW',ACCESS='DIRECT',RECL=IRECL2)
     31 C****     READ MILLIVOLT CONVERSION FACTORS FOR ALL A/D FILE SIGNALS(OLD)
     32           DO 20 I=1,NSIGNL
  1  33           WRITE(*,11) I
  1  34     11    FORMAT(' ENTER A/D SIGNAL M-V GAIN(',I2,')=')
  1  35           READ(*,12) GAIN(I)
  1  36     20    CONTINUE
     37     12    FORMAT(3F18.2)
     38 C****     READ TC-TABLE.
     39           WRITE(*,'(A/)') ' ENTER NO.OF POINTS TO READ FROM TC-TABLE'
     40           READ(*,10) NTC
     41           DO 21 I=1,NTC
  1  42           READ(10,REC=I) XMV(I),XTEMP(I)
  1  43     21    CONTINUE
     44 C****     IDENTIFY THE TC-OUTPUTS REQUIRING CONVERSION FROM
     45 C****     MILLIVOLT TO DEG F, USING CONVERSION TABLE.
     46           DO 22 I=1,NSIGNL
  1  47           WRITE(*,15) I
  1  48     15    FORMAT(' ENTER 1 IF SIGNAL ',I2,' NEEDS CONV TO DEG F=')
  1  49           READ(*,10) IFLAG
  1  50           IF(IFLAG.EQ.1) GO TO 23
  1  51           ISIGF(I)=0
  1  52           GO TO 22
  1  53     23    ISIGF(I)=I
  1  54     22    CONTINUE
     55 C****     CONVERSION TO MILLIVOLTS.
     56           DO 100 K=1,NREC
```

```
D Line# 1      7                          Microsoft FORTRAN77 V3.30 March 1985
1    57            READ(11,REC=K) (ISTO(I),I=1,NSIGNL)
1    58            DO 30 I=1,NSIGNL
2    59     30     X1(I)=FLOAT(ISTO(I)-2048)/GAIN(I)
1    60 C****      CONVERT REF JUNCTION TEMP INTO TC-MILLIVOLTS.
1    61            F=(X1(NJTN)-273.15)*1.8+32.0
1    62 C
1    63            FMIN=XTEMP(1)
1    64            FMAX=XTEMP(2)
1    65            ICOUNT=2
1    66   101      CONTINUE
1    67            IF(F.GT.FMAX) GO TO 31
1    68            IF(F.GE.FMIN.AND.F.LE.FMAX) GO TO 32
1    69     31     FMIN=FMAX
1    70            FMAX=XTEMP(ICOUNT+1)
1    71            ICOUNT=ICOUNT+1
1    72            GO TO 101
1    73     32     VOLT=XMV(ICOUNT-1)+(XMV(ICOUNT)-XMV(ICOUNT-1))/(XTEMP(ICOUNT)
1    74        *   -XTEMP(ICOUNT-1))*(F-XTEMP(ICOUNT-1))
1    75 C****      CORRECT MILLIVOLT READINGS OF TC OUTPUTS.
1    76 C****      SOME OF THE READINGS ARE DIRECTLY CONVERTED TO DEG F.
1    77            DO 33 I=1,NSIGNL
2    78            IF(I.NE.ISIGF(I).OR.I.EQ.NJTN) GO TO 33
2    79            X1(I)=X1(I)+VOLT
2    80            FMIN=XMV(1)
2    81            FMAX=XMV(2)
2    82            ICOUNT=2
2    83   102      CONTINUE
2    84            IF(X1(I).GT.FMAX) GO TO 35
2    85            IF(X1(I).GE.FMIN.AND.X1(I).LE.FMAX) GO TO 36
2    86     35     FMIN=FMAX
2    87            FMAX=XMV(ICOUNT+1)
2    88            ICOUNT=ICOUNT+1
2    89            GO TO 102
2    90     36     X1(I)=XTEMP(ICOUNT-1)+(XTEMP(ICOUNT)-XTEMP(ICOUNT-1))/
2    91        *   (XMV(ICOUNT)-XMV(ICOUNT-1))*(X1(I)-XMV(ICOUNT-1))
2    92     33     CONTINUE
1    93            WRITE(12,REC=K) (X1(I),I=1,NSIGNL)
1    94   100      CONTINUE
     95            CLOSE(10)
     96            CLOSE(11)
     97            CLOSE(12)
     98 C****
     99            STOP
    100            END

Name     Type         Offset P Class

F        REAL          28412
FILIN    CHAR*20       28162
FILOUT   CHAR*20       28202
FLOAT                        INTRINSIC
FMAX     REAL          28420
FMIN     REAL          28416
FNAME    CHAR*20       28182
GAIN     REAL          28096
I        INTEGER*4     28222
```

Microsoft FORTRAN77 V3.30 March 1985

D Line# 1 7

| | | |
|---|---|---|
| ICOUNT | INTEGER*2 | 28424 |
| IFLAG | INTEGER*4 | 28392 |
| IRECL1 | INTEGER*4 | 28146 |
| IRECL2 | INTEGER*4 | 28150 |
| ISIGF | INTEGER*2 | 28076 |
| ISTO | INTEGER*2 | 16 |
| K | INTEGER*4 | 28396 |
| NJTN | INTEGER*4 | 28158 |
| NREC | INTEGER*4 | 28154 |
| NSIGNL | INTEGER*4 | 28136 |
| NTC | INTEGER*4 | 28328 |
| VOLT | REAL | 28426 |
| X1 | REAL | 36 |
| XMV | REAL | 76 |
| XTEMP | REAL | 14076 |

101

| Name | Type | Size | Class |
|---|---|---|---|
| MAIN | | | PROGRAM |

Pass One No Errors Detected
101 Source Lines

# APPENDIX D

## LISTING FOR "PLOT1.FOR"

```
D Line#  1      7                          Microsoft FORTRAN77 V3.30 March 1985
       1 C          PROGRAM PLOT1
       2 C          THIS PROGRAM PLOTS TWO GRAPHS ON THE SAME
       3 C          AXIS AND IN DIFFERENT COLORS ON THE MONITOR
       4 C
       5            INTEGER*2 MHSBUF(512)
       6            COMMON/MHSPLT/MHSBUF
       7            REAL*4 X1(2000),TIME(2000),X2(2000),X(2000),X3(2000)
       8            CHARACTER*20 FNAME
       9            CHARACTER*60 HEAD ,YFNAME
      10 C
      11            WRITE(*,'(A/)') ' ENTER DELTAT (SEC)= '
      12            READ(*,11) DELTAT
      13            WRITE(*,'(A)') ' NO. OF POINTS TO PLOT, N= '
      14            READ(*,10) N
      15            WRITE(*,'(A)') ' ENTER IFLAG=1 IF Y-SCALE MIN,MAX REQD= '
      16            READ(*,10) IFLAG
      17            IF(IFLAG.NE.1) GO TO 50
      18            WRITE(*,'(A/)') ' ENTER MIN,MAX FOR Y-AXIS'
      19            READ(*,11) XMIN,XMAX
      20            DO 25 I=1,N
 1    21            X1(I)=XMIN
 1    22    25      X2(I)=XMIN
      23            X1(N)=XMAX
      24            X2(N)=XMAX
      25            CALL SCALAR(1,2,N,X1,IER)
      26            CALL SCALAR(4,2,N,X2,IER)
      27            DO 26 I=1,N
 1    28            X1(I)=0.0
 1    29    26      X2(I)=0.0
      30    10      FORMAT(6I6)
      31 11         FORMAT(3F12.0)
      32 C          DEFINE AND READ FILES FOR X1-ARRAY,X2-ARRAY
      33    50      CONTINUE
      34            WRITE(*,'(A/)') ' ENTER DEVICE NO. TO PLOT (0,1,2)= '
      35            READ(*,10) IDEV
      36            TIME(1)=0.0
      37            DO 20 I=2,N
 1    38 20         TIME(I)=TIME(I-1)+DELTAT
      39 C          X1-ARRAY FILE (EXPERIMENTAL)
      40            WRITE(*,'(A/)') ' X1-ARRAY FILENAME ?'
      41            READ(*,'(A)') FNAME
      42            WRITE(*,'(A/)') ' ENTER 60 CHAR TITLE'
      43            READ(*,'(A)') HEAD
      44            WRITE(*,*) 'ENTER 60 CHARACTER Y-AXIS TITLE'
      45            READ(*,'(A)') YFNAME
      46            WRITE(*,'(A/)') ' ENTER IPRINT=1 IF PLOT PRINT REQD='
      47            READ(*,10) IPRINT
      48            WRITE(*,'(A/)') ' ENTER NO.OF RECORDS TO SKIP AT FILE START'
      49            READ(*,10) NSKIP
      50            WRITE(*,'(A/)') ' NO. OF SIGNALS IN EXPT. FILE (X1-ARRAY)?'
      51            READ(*,10) ISIGNL
      52            IRECL=4*ISIGNL
      53            OPEN(10,FILE=FNAME,STATUS='OLD',ACCESS='DIRECT',RECL=IRECL)
      54            WRITE(*,'(A/)') ' CHAN. NO. OF DESIRED EXPT. SIGNAL?'
      55            READ(*,10) ICHAN
      56            NSTART=NSKIP+1
```

```
D Line# 1      7                          Microsoft FORTRAN77 V3.30 March 1985
      57            NEND=NSKIP+N
      58            DO 21 K=NSTART,NEND
1     59             READ(10,REC=K) (X(I),I=1,ISIGNL)
1     60 21         X1(K-NSKIP)=X(ICHAN)
      61            CLOSE(10)
      62 C          X2-ARRAY FILE (ANALYTICAL)
      63            WRITE(*,'(A/)') ' X2-ARRAY FILE NAME?'
      64            READ(*,'(A)') FNAME
      65            WRITE(*,'(A/)') ' ENTER NO.OF RECORDS TO SKIP AT FILE START'
      66            READ(*,10) NSKIP
      67            WRITE(*,'(A/)') ' NO. OF SIGNALS IN ANALYTICAL FILE (X2)?'
      68            READ(*,10) ISIGNL
      69            IRECL=4*ISIGNL
      70            OPEN(10,FILE=FNAME,STATUS='OLD',ACCESS='DIRECT',RECL=IRECL)
      71            WRITE(*,'(A/)') ' CHAN. NUMBERS OF 2 DESIRED SIGNALS ?'
      72            READ(*,10) ICHAN1,ICHAN2
      73            NSTART=NSKIP+1
      74            NEND=NSKIP+N
      75            DO 22 K=NSTART,NEND
1     76             READ(10,REC=K) (X(I),I=1,ISIGNL)
1     77            X2(K-NSKIP)=X(ICHAN1)
1     78 22         X3(K-NSKIP)=X(ICHAN2)
      79            CLOSE(10)
      80            CALL SELECT(IDEV,0,IER)
      81 C          SCALE TIME AXIS
      82            CALL SCALAR(2,1,N,TIME,IER)
      83            IF(IFLAG.EQ.1) GO TO 51
      84 C          SCALE Y-AXIS (X1-ARRAY)
      85            CALL SCALAR(1,2,N,X1,IER)
      86 C          SCALE Y-AXIS (X2-ARRAY)
      87            CALL SCALAR(4,2,N,X2,IER)
      88            CALL SCALAR(4,2,N,X3,IER)
      89
      90    51      CALL SETSCL
      91            CALL ERASE(IER)
      92 C
      93            CALL SETGRD(1,1,0,0)
      94            CALL DOALL(N,TIME,X1,HEAD,999,'$TIME (SEC)$',
      95           *  999,YFNAME,999,0)
      96            CALL SETPEN(1,IER)
      97            CALL CONECT(N,TIME,X2)
      98            CALL CONECT(N,TIME,X3)
      99            IF(IPRINT.NE.1) GO TO 100
     100            CALL PRNTGR(IER)
     101   100      PAUSE
     102            CALL DESLCT(IDEV,IER)
     103            STOP
     104            END

Name     Type         Offset P Class

DELTAT REAL           40016
FNAME  CHAR*20        40124
HEAD   CHAR*60        40144
I      INTEGER*4      40036
ICHAN  INTEGER*4      40280
```

D Line# 1 7 Microsoft FORTRAN77 V3.30 March 1985

| | | | |
|---|---|---|---|
| ICHAN1 | INTEGER*4 | 40304 | |
| ICHAN2 | INTEGER*4 | 40308 | |
| IDEV | INTEGER*4 | 40116 | |
| IER | INTEGER*4 | 40044 | |
| IFLAG | INTEGER*4 | 40024 | |
| IPRINT | INTEGER*4 | 40264 | |
| IRECL | INTEGER*4 | 40276 | |
| ISIGNL | INTEGER*4 | 40272 | |
| K | INTEGER*4 | 40292 | |
| MHSBUF | INTEGER*2 | 0 | /MHSPLT/ |
| N | INTEGER*4 | 40020 | |
| NEND | INTEGER*4 | 40288 | |
| NSKIP | INTEGER*4 | 40268 | |
| NSTART | INTEGER*4 | 40284 | |
| TIME | REAL | 24016 | |
| X | REAL | 32016 | |
| X1 | REAL | 16 | |
| X2 | REAL | 8016 | |
| X3 | REAL | 16016 | |
| XMAX | REAL | 40032 | |
| XMIN | REAL | 40028 | |
| YFNAME | CHAR*60 | 40204 | |

105
106
107
108

| Name | Type | Size | Class |
|---|---|---|---|
| CONECT | | | SUBROUTINE |
| DESLCT | | | SUBROUTINE |
| DOALL | | | SUBROUTINE |
| ERASE | | | SUBROUTINE |
| MAIN | | | PROGRAM |
| MHSPLT | | 1024 | COMMON |
| PRNTGR | | | SUBROUTINE |
| SCALAR | | | SUBROUTINE |
| SELECT | | | SUBROUTINE |
| SETGRD | | | SUBROUTINE |
| SETPEN | | | SUBROUTINE |
| SETSCL | | | SUBROUTINE |

Pass One No Errors Detected
108 Source Lines

# VITA

Frederik Petrus Wolvaardt was born in Aliwal North, South Africa on December 22, 1962. He attended a primary school in that city and was graduated from Aliwal North High School in November 1980. The following February he entered the University of the Orange Free State in Bloemfontein, South Africa. In September 1981 he received a scholarship from the Electricity Supply Commission of South Africa and enrolled in the Nuclear Engineering program at the University of Tennessee, Knoxville. He received the Bachelor of Science Degree in Nuclear Engineering in June 1985.

In the summer of 1985 he accepted a research assistantship at the University of Tennessee, Knoxville and began study toward a Master's degree. This degree was awarded in December 1986.

The author is a member of Tau Beta Pi, the American Nuclear Society, and the Instrument Society of America. After graduation, Mr. Wolvaardt will be employed by the Electricity Supply Commission of South Africa.