NASA/TM-20205000446

Analytical Redundancy Using Kalman Filters for Rocket Engine Sensor Validation

Nicklaus O. Richardson
University of Illinois, Urbana-Champaign, Illinois

Edmond Wong and Kevin Melcher
Glenn Research Center, Cleveland, Ohio

October 2020

NASA STI Program . . . in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA Scientific and Technical Information (STI) Program plays a key part in helping NASA maintain this important role.

The NASA STI Program operates under the auspices of the Agency Chief Information Officer. It collects, organizes, provides for archiving, and disseminates NASA's STI. The NASA STI Program provides access to the NASA Technical Report Server—Registered (NTRS Reg) and NASA Technical Report Server—Public (NTRS) thus providing one of the largest collections of aeronautical and space science STI in the world. Results are published in both non-NASA channels and by NASA in the NASA STI Report Series, which includes the following report types:

- TECHNICAL PUBLICATION. Reports of completed research or a major significant phase of research that present the results of NASA programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA counter-part of peer-reviewed formal professional papers, but has less stringent limitations on manuscript length and extent of graphic presentations.

- TECHNICAL MEMORANDUM. Scientific and technical findings that are preliminary or of specialized interest, e.g., "quick-release" reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.

- CONTRACTOR REPORT. Scientific and technical findings by NASA-sponsored contractors and grantees.

- CONFERENCE PUBLICATION. Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or co-sponsored by NASA.

- SPECIAL PUBLICATION. Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.

- TECHNICAL TRANSLATION. English-language translations of foreign scientific and technical material pertinent to NASA's mission.

For more information about the NASA STI program, see the following:

- Access the NASA STI program home page at http://www.sti.nasa.gov

- E-mail your question to help@sti.nasa.gov

- Fax your question to the NASA STI Information Desk at 757-864-6500

- Telephone the NASA STI Information Desk at 757-864-9658

- Write to:
  NASA STI Program
  Mail Stop 148
  NASA Langley Research Center
  Hampton, VA 23681-2199

NASA/TM-20205000446

Analytical Redundancy Using Kalman Filters for Rocket Engine Sensor Validation

Nicklaus O. Richardson
University of Illinois, Urbana-Champaign, Illinois

Edmond Wong and Kevin Melcher
Glenn Research Center, Cleveland, Ohio

National Aeronautics and
Space Administration

Glenn Research Center
Cleveland, Ohio 44135

October 2020

Level of Review: This material has been technically reviewed by technical management.

Available from

NASA STI Program
Mail Stop 148
NASA Langley Research Center
Hampton, VA 23681-2199

National Technical Information Service
5285 Port Royal Road
Springfield, VA 22161
703-605-6000

This report is available in electronic form at http://www.sti.nasa.gov/ and http://ntrs.nasa.gov/

Analytical Redundancy Using Kalman Filters for Rocket Engine Sensor Validation

Nicklaus O. Richardson*
University of Illinois
Urbana-Champaign, Illinois 61801

Edmond Wong and Kevin Melcher
National Aeronautics and Space Administration
Glenn Research Center
Cleveland, Ohio 44135

Abstract

The use of sensor redundancy is crucial in aerospace systems to maintain safe, reliable operation. While hardware redundancy is more common in application, analytical redundancy can provide a viable alternative in systems where the installation of multiple redundant sensors is not viable. To this end, the use of Kalman filters to analytically validate sensor measurements within rocket engines was explored. First, a dynamic model of the RS-25 engine, a derivative of the Space Shuttle Main Engine (SSME), was reduced to a subset of relations, focused around the main combustion chamber pressure. These relations were used within the Kalman filter algorithm to generate an estimate of sensor measurements to be compared with true measurements for data validation purposes. By using a bank of Kalman filters, the residuals between the estimated and true measurements were used to detect and isolate sensor faults. Through fault simulations, the sensor validation performance of this Kalman filter bank design was compared to a hardware redundancy check. Sensor bias and drift faults of various magnitudes were injected into nominal RS-25 engine test data. Results for both approaches show comparable fault detection with most bias faults found nearly instantaneously by both algorithms. Drift fault detection results show certain cases where one algorithm is faster than the other. The key advantage of the Kalman filter algorithm is shown in fault isolation performance where it can isolate faults between two redundant sensors while the hardware redundancy comparisons cannot.

Nomenclature

CCV	Chamber Coolant Valve
FP	Fuel Preburner
HGM	Hot Gas Manifold
HPFTP	High Pressure Fuel Turbopump
HPOTP	High Pressure Oxidizer Turbopump
HRF	Hardware Redundancy Failure
LPFTP	Low Pressure Fuel Turbopump
LPOTP	Low Pressure Oxidizer Turbopump
MCC	Main Combustion Chamber
MFV	Main Fuel Valve
MOV	Main Oxidizer Valve

*Summer Intern in Lewis' Educational and Research Collaborative Internship Project (LeRCIP).

OP Oxidizer Preburner
SLS Space Launch System
WSSR Weighted Sum of Squared Residuals

1.0 Introduction

In order to ensure safe operation, most aerospace systems have some type of redundancy in flight hardware to insure fault tolerance. If a hardware fault occurs, there must be some way to first detect the fault and then to identify or isolate the faulty component. Launch vehicles in particular have a strong need to do this for sensors. In the past, sensor failures dominated engine anomaly reports for the Space Shuttle-(Ref. 1). This led to numerous aborts and investigations. Given the high costs associated with launch vehicles, technologies to help avoid unnecessary delays will be critical to overall program development. More recently, with the current push for humans to explore deep space, the need for increased reliability of launch vehicles is growing. The investigation of more reliable sensor validation techniques therefore becomes an important area of research.

Traditional sensor validation algorithms for launch vehicle systems rely on limit checks and hardware redundancy checks (Refs. 2 and 3). Limit checks ensure that individual sensor readings are within an expected nominal operating range. Hardware redundancy checks insure that a set of similar sensors, placed together at the same location, all read similar values. However, hardware redundancy checks have several limitations. First, these checks rely on having multiple sensors at the same location. This may be particularly difficult in aerospace systems where weight reduction is a fundamental design driver. Additionally, placement of redundant sensors (known as a sensor "redundancy group") at one location may not be ideal or even possible given geometric space restrictions or extreme environmental operating conditions. For example, the Main Combustion Chamber (MCC) pressure of a rocket engine is an important measurement for successful vehicle operation, but it presents practical challenges for sensor placement. Another consideration is that ideally there should be at least three sensors for sensor fault detection and isolation. Having three sensors allows for a faulty sensor to be easily discriminated from the other two. When there are only two sensors within the redundancy group, it is possible to detect a sensor fault when the sensor measurements do not agree, but it will not be possible to isolate which sensor is faulty. This situation is known as the two sensor problem. A previous study explored solutions to this problem using statistical methods (Ref. 4).

Due to the limitations of hardware redundancy, a better solution to sensor validation may be analytical redundancy. Analytical redundancy leverages known mathematical relationships between system parameters to produce estimates of sensor measurements. With this approach, it is possible to base redundancy on a set of disparately located, nonhomogeneous sensors, which measure different types of parameters, as long as the number of sensors is sufficient to provide observability into the system and the mathematical relationships between these sensors can adequately model the dynamic states of the system. For the two sensor problem, this approach is able to determine not only which sensor is faulty, but also if both sensors are faulty. Algorithms based on analytical redundancy have been shown to be effective for sensor validation on launch system power distribution systems (Ref. 3) as well as for the Space Shuttle Main Engine (SSME) performance monitoring (Ref. 1). Of particular interest is the SSME performance monitoring. Here an algorithm based on Bayesian decision theory was able to reliably detect sensor faults using relations between groups of two sensors for 22 sensors within the engine.

This paper focuses on applying analytical redundancy based on the Kalman filter for sensor validation in the RS-25 rocket engine. The Kalman filter was chosen as a reliable and proven filtering technique. It has been used extensively since its conception in 1960 (Ref. 5). In fact, it saw its first application within

the space industry during its use in Apollo navigation computers (Ref. 6). More recently, it has seen extensive investigation for use in aircraft gas turbine engine health management systems (Refs. 7 to 13). Several Kalman filter-based diagnostic techniques have been developed and applied to the SSME (Refs. 14 and 15). Given their widespread use within the aerospace industry, analytical redundancy algorithms based on Kalman filter techniques are good candidates for application in the health management system of the Space Launch System (SLS).

This paper is organized as follows. Section 2.0 provides an overview of two methodologies explored in this paper. These methods include: 1) an approach using a bank of Kalman filters, and 2) a hardware redundancy algorithm. Next, Section 3.0 presents background information on the RS-25 engine, which functions as the host system of this study. A simple RS-25 mathematical model is presented along with discussion on how the Kalman filter is applied to the RS-25. The employment of fault simulation in this work is also explained. In Section 4.0, results are shown and discussed. Finally, a conclusion is provided in Section 5.0.

2.0 Methodology

The study documented in this paper explores the application of a Kalman filter-based analytical redundancy method for validation of RS-25 sensor data. For the purposes of assessing the performance of this approach, the results are compared against those provided by another sensor validation method based on hardware redundancy. The following subsection will provide a general overview of the Kalman filter, followed by a more specific explanation of the sensor fault detection and isolation methodology based on the application of a bank of Kalman filters. Finally, a subsection is included that provides a brief explanation of the hardware redundancy algorithm.

2.1 Kalman Filtering

2.1.1 Kalman Filter Selection

The specific Kalman filter chosen is a steady-state continuous time Kalman filter. A good description of many Kalman filter types and methods is given by Simon (Ref. 16) and Zarchan (Ref. 5). In a conventional Kalman filter implementation the Kalman gain matrix and the state estimation error covariance matrix are continuously updated. However, in a steady-state Kalman filter implementation the Kalman gain matrix and state estimation error covariance matrix are pre-calculated off-line and held constant (Ref. 13). This means that repeated on-line solving of the differential Riccati equation is no longer required, reducing both system complexity and computational power required (Ref. 16).

2.1.2 Kalman Filter Algorithm

The continuous time Kalman filter uses a continuous time linear state space model given by the following set of equations

$$\begin{aligned} \dot{\boldsymbol{x}} &= A(\boldsymbol{x}) + B(\boldsymbol{u}) + \boldsymbol{w} \\ \boldsymbol{y} &= C(\boldsymbol{x}) + \boldsymbol{v} \\ \boldsymbol{w} &\sim (0, Q_c) \\ \boldsymbol{v} &\sim (0, R_c) \end{aligned} \tag{1}$$

where $\boldsymbol{x}$ is the change in the state vector, $\boldsymbol{u}$ is the change in the control input vector, $\boldsymbol{y}$ is the change in the measurement vector, $\boldsymbol{w}$ is the process noise, $\boldsymbol{v}$ is the measurement noise, and matrices A, B, and C describe the system dynamics. The process noise and measurement noise are assumed to have a zero-mean normal distribution with covariance Q_c and R_c, respectively. The changes in the state, control, and measurement vectors are with respect to nominal trim values. In a linearized system, such as that considered in this study, these trim values can be selected as the point around which the system is linearized (Ref. 10). This state space model is used to construct the Kalman filter. The state estimation error covariance matrix, P, is determined by solving the differential Riccati equation as given by

$$\dot{P} = -PC^T R_c^{-1} CP + AP + PA + Q_c \tag{2}$$

where the initial conditions are

$$P(0) = E[\big(\boldsymbol{x}(0) - \hat{\boldsymbol{x}}(0)\big)\big(x(0) - \hat{\boldsymbol{x}}(0)\big)^T] \tag{3}$$

where $\hat{\boldsymbol{x}}$ is the state estimate, x is the true state, and E is the expected value function. Assuming steady-state, as done in this study, the derivative in Equation (2) becomes zero and an algebraic equation is left to solve for P.

After the calculation of P, the next step in the algorithm is to update the state estimate based on the observations. One must first calculate the Kalman gain from the following

$$K = PC^T R_c^{-1} \tag{4}$$

where K is the Kalman gain. Once the Kalman gain is calculated, the state can be updated with the following

$$\begin{gathered} \dot{\hat{\boldsymbol{x}}} = A\hat{\boldsymbol{x}} + B\boldsymbol{u} + K(\boldsymbol{y} - C\hat{\boldsymbol{x}}) \\ \hat{y} = C\hat{\boldsymbol{x}} \end{gathered} \tag{5}$$

where the first two terms in the first equation come from the state space model in Equation (1) and the last two terms are updates from the measurement error. The Kalman gain can be thought of as the amount of confidence one has in the measurements versus the dynamic model. From Equation (5), one can see that a higher gain will lead to a higher weight on the measurement portion of the state update.

2.1.3 Kalman Filter-Based Sensor Validation

Kalman filter-based sensor fault detection and isolation can be performed by applying a bank of Kalman filters (Refs. 7 to 11). Here, multiple Kalman filters are constructed, each designed to detect a specific sensor fault. In the event that a fault does occur, all filters except the one using the correct hypothesis will produce large estimation errors, thereby allowing isolation of a specific sensor fault.

Each Kalman filter is designed with the hypothesis that one sensor is faulty and, therefore, discards that measurement along with its associated row in the C matrix and the measurement vector $\boldsymbol{y}$. A requirement for this approach is that each resulting matrix pair (A, C) must be observable in order for the corresponding Kalman filter to converge. The bank of Kalman filter design, is shown in Figure 1, and includes m Kalman filters—one for each potential faulty sensor measurement. The process involves three stages: sensor sorting, Kalman filtering, and error computation. The incoming measurement vector, composed of all system measurements under consideration, is sorted into m different vectors, each of length m-1 and excluding a different hypothesized faulty measurement. The shortened measurement vectors are then passed as inputs to the corresponding Kalman filter in the second stage of the process.

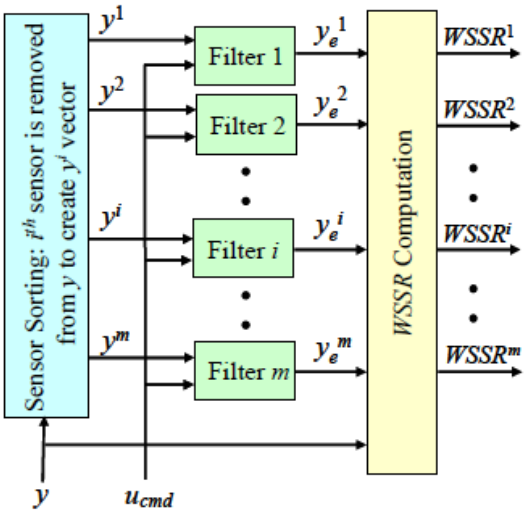

Figure 1.—Kalman filter bank design. Incoming measurement signals are sorted and passed to each Kalman filter, excluding the hypothesized faulty measurement. A measurement estimate is generated and compared to the incoming measurement signals (Ref. 9).

Each Kalman filter generates an estimate of the $m\times1$ measurement vector as determined by the second equation in Equation (5) and passes this estimate to the error computation stage. In the error computation stage, the measurement estimates are compared to the true sensor measurements. The residuals are first calculated as given by

$$e^i = \hat{y}^i - y^i \tag{6}$$

where e^i is the error vector of the i^{th} Kalman filter estimate, $\hat{y}^i$ is the measurement vector estimate produced by the i^{th} Kalman filter, and y^i is the sensor measurement vector input into the i^{th} Kalman filter. This error vector is then used to calculate a weighted sum of squared residuals (WSSR) signal using the following equation presented by Kobayashi (Ref. 8)

$$\begin{aligned} WSSR^i &= W_r^i \left(e^i\right)^T \left(\Sigma^i\right)^{-1} e^i \\ \Sigma^i &= diag\left[\sigma^i\right]^2 \end{aligned} \tag{7}$$

where σ^i is a vector of standard deviations of the i^{th} sensor subset and W_r^i is a scalar weight chosen so that the WSSR is less than a given threshold. Tuning of this parameter is a process that involves a tradeoff between false alarms and missed detections. While not explored in this study, it should be noted that analytical approaches for selecting these weights have been established in other works. Since the weighting used to determine the weighted sum of squared residuals is based on the standard deviation of the sensor measurements, which is assumed to be zero-mean normally distributed noise, the resulting WSSR signal will be a chi square distribution with k degrees of freedom (where k = number of sensors used in calculating the WSSR). Therefore, it is possible to calculate the threshold value needed to provide a desired false positive rate.

In order to detect when a fault has taken place, the WSSR of each Kalman filter in the entire bank of filters is examined. In the event of a sensor fault, any Kalman filter that uses the faulty sensor measurement will produce inaccurate sensor measurements, which will result in increased WSSR. If one or more WSSR signals exceeds the chosen threshold, a fault is detected. Isolating the faulty sensor is more difficult. The WSSR signal with the correct hypothesis should not change due to a fault as it does not include the faulty sensor. By looking at which estimate does not change, it is therefore possible to isolate which sensor is faulty. However, in the presence of sensor noise, this can be more difficult as some signals may be more prone to instantaneous changes than others.

2.2 Hardware Redundancy Check

The Kalman filter algorithm was compared to the hardware redundancy check sensor validation approach as described by Wong (Ref. 2). The hardware redundancy check algorithm compares the sensor data values from a group of hardware redundant sensors, which are concurrently sampled. The sensors comprising such a redundancy group are of a homogeneous sensor type and are generally co-located at a point within the engine to enable them to measure the same physical property at the same sensing location. As a result, if a sensor provides measurements that are unreasonably dissimilar from the measurements provided by the other sensors within the redundancy group, the algorithm is able to detect and identify the outlier.

During each computation cycle, the algorithm calculates the absolute value of the differences (i.e., the delta) between all unique pairs of sensors within the redundancy group and compares these values to a predefined threshold limit value. If the threshold limit is exceeded by the delta for any pair combination, an individual “delta limit violation” counter is incremented for each sensor involved in calculating that particular delta value. The threshold limits used by this algorithm are pre-defined for an application system and are generally based on the error tolerance specifications for the specific sensor devices used for acquiring measurements on that system.

Once the deltas for all unique pairs have been assessed for the current computation cycle, the delta limit violation counters are tallied for all of the sensors. If only one sensor has the maximum possible amount of delta limit violations possible (i.e., accrues a delta limit violation for every pair combination with which it is involved), it is considered an outlier and a Hardware Redundancy Failure (HRF) strike counter is incremented for this sensor for the current computation cycle. All other sensors’ HRF counters are reset to zero for the cycle, as well. If a sensor continues to demonstrate the outlying value persistently for three consecutive computation cycles, which corresponds with the HRF counter associated with the sensor reaching a persistence limit value (defined as three), the sensor is permanently disqualified from use. If the sensor value does not demonstrate a violation in a cycle following a cycle during which it does, the associated HRF counter is reset to zero. This persistence criteria requiring a violation for three consecutive cycles helps guard against the false positive disqualification of sensor data based on spurious violations.

If there are three or more sensors in the redundancy group and two or more sensors have the maximum possible delta limit violations, the faulty sensor cannot be determined by this algorithm alone. Therefore, the algorithm can only detect single occurrence failures. In practice, the Hardware Redundancy check is implemented in conjunction with other types of sensor data qualification algorithms, which will be relied upon in this scenario.

In the case where there are only two valid sensors in the redundancy group and the absolute delta between them violates the threshold limit, the algorithm can detect that there is a fault but cannot isolate which sensor is faulty. As a result, the HRF counter is incremented for both sensors.

3.0 Application

3.1 RS-25 Rocket Engine

3.1.1 Engine Description

This study investigates the use of Kalman filter-based sensor data validation for rocket engines systems. Specifically, the RS-25 engine was selected for this purpose as it is currently being utilized as the booster engine for the SLS. A diagram of the engine is shown in Figure 2. A summary of the engine operation is given here, but a more complete description of is provided by Tanguchi (Ref. 15). The RS-25 engine is a bipropellant, staged combustion cycle engine. In the staged combustion cycle process, an initial combustion of propellant takes place in fuel and oxidizer preburners, creating fuel-rich gas to power high and low pressure turbopumps before traveling to the Main Combustion Chamber (MCC). This process leads to an efficient, but complex system.

The RS-25 uses liquid hydrogen and liquid oxygen as fuel and oxidizer, respectively. Initially, the fuel and oxidizer are pumped into the engine by the Low Pressure Fuel Turbopump (LPFTP) and Low Pressure Oxidizer Turbopump (LPOTP), respectively. These pumps provide a net positive pressure to the two high pressure pumps. The discharges of the low pressure pumps are passed into the High Pressure Fuel Turbopump (HPFTP) and High Pressure Oxidizer Turbopump (HPOTP) where the pressures of each liquid are further increased. Here similarities between the fuel side and oxidizer side of the engine end.

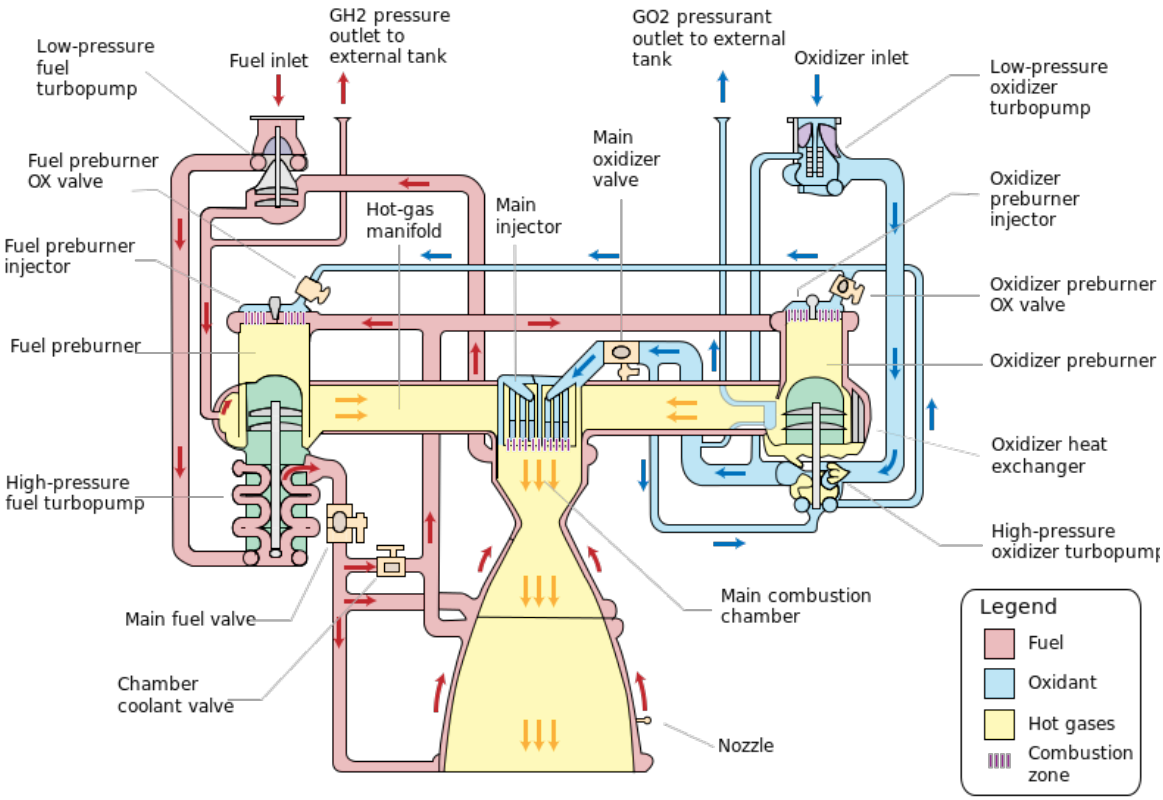

Figure 2.—RS-25 engine diagram. Shows the main architecture of the RS-25 rocket engine used in this study (Graphic by NASA).

The fuel is discharged from the HPFTP through the Main Fuel Valve (MFV) where it becomes gaseous. Part of the flow is used as coolant and splits, passing along the walls of the nozzle and MCC. The other portion of the flow passes through the Chamber Coolant Valve (CCV), combines with the coolant flow from the nozzle walls and enters the combustors of the two preburners, namely, the Fuel Preburner (FP) and the Oxidizer Preburner (OP). The coolant flow around the MCC is first diverted into the LPFTP before entering the two preburners. This flow drives the LPFTP. Once in the preburners, combustion takes place and the hot-gases are routed through the high-pressure pumps and Hot Gas Manifold (HGM) into the MCC. After combustion within the MCC, gases are accelerated through the nozzle to generate thrust.

The oxidizer side of the engine is simpler as no oxygen is used as coolant. There are two oxidizer lines from the HPOTP. The first drives the LPOTP and the second splits into two lines to the Main Oxidizer Valve (MOV) and into the boost pump. The boost pump is mounted on the end of the HPOTP and increases pressure to that necessary for the fuel preburners' fuel injectors. The flow through the MOV leads directly into the MCC fuel injectors for combustion.

3.1.2 Engine Dynamic Model

In his thesis, Lozano-Tovar (Ref. 18) derives a dynamic model of the entire RS-25 engine and lists engine constants where necessary. While this model is based on an earlier iteration of the engine, it was assumed to be applicable to the current design because the main architecture of the RS-25 engine has remained essentially the same. Although it is expected that some constants may differ in the current engine design, this model is sufficiently representative for the purposes of this study. Also, it is important to note that some measurements assumed by Lozano-Tovar were removed from the current RS-25 sensor suite. For example, there is no longer a sensor to measure the HPOTP turbine speed. Despite this, a subset of sensors and dynamic equations that could be used was identified and selected.

The dynamic model by Lozano-Tovar consists of 38 coupled differential equations. These equations are derived from the principles of rotational and fluid dynamics, energy balance, heat exchange, and time delay. Using the governing differential equations at different engine locations yields the 38 equations given by Lozano-Tovar.

3.1.3 Reduced Model

Due to the complexity of the equations, a subset of sensors and the dynamic model were selected as the basis for a proof of concept demonstration. A reduced engine model was formed using a subset of four sensors and three differential equations. The MCC pressure was the focus when forming this model since MCC pressure is an essential parameter in engine operation. Due to its significance, the MCC pressure is measured using four redundant sensors installed on the RS-25. Therefore, the performance of the Kalman filter-based analytical redundancy algorithm could be compared with the hardware redundancy check.

With MCC pressure as the starting point, its dynamic equation was examined to find which parameters would be needed to propagate the engine state forward in time for the Kalman filter time update. The differential equation of the MCC pressure is given by

$$\left(\frac{V}{JR_gT_c}\right)\frac{d}{dt}P_c = \dot{m}_{FI} + \dot{m}_{mov} - \dot{m}_{cn} \tag{8}$$

where $\left(\frac{V}{JR_gT_c}\right)$ is reduced to an engine constant given as 0.00025, P_c is the MCC pressure, $\dot{m}_{FI}$ is the mass flow rate through the main fuel injector, $\dot{m}_{mov}$ is the mass flow rate through the MOV, and $\dot{m}_{cn}$ is the mass flow rate through the nozzle. The mass flow rate through the nozzle is given by the following set of equations

$$\dot{m}_{cn} = C^* P_c \tag{9}$$

$$C^* = \frac{29122}{35000 + 210500r^* - 173500(r^*)^2} \tag{10}$$

$$r^* = \frac{\dot{m}_{mov} + \dot{m}_{op3}}{\dot{m}_{mov} + \dot{m}_{FI}} \tag{11}$$

where C^* is defined as the equilibrium flow rate factor, r^* is the mass flow rate ratio (oxidizer to fuel), and $\dot{m}_{op3}$ is the flow rate through the oxidizer boost pump. The simplifying assumption of a constant mass flow rate ratio was made to reduce nonlinearity and simplify the required state variables to exclude $\dot{m}_{op3}$. This mass flow rate is given by a separate differential equation. It is calculated by several other state parameters and would expand the problem greatly. For simplicity, r^* was assumed to be constant. Overall, the engine is assumed to be in quasi-steady state, so a constant mass flow rate can be expected for most of the engine as valves should not change position significantly during this time. The MOV for example remains fully open for most of engine operation.

By making this assumption, C^* becomes a constant and Equation (8) becomes a function of $\dot{m}_{FI}$, $\dot{m}_{mov}$, and MCC pressure only. Therefore, expressions for $\dot{m}_{FI}$ and $\dot{m}_{mov}$ are still needed. The derivative of the mass flow rate through the MOV is given by

$$\left(\frac{L}{gA}\right)\frac{d}{dt}\dot{m}_{mov} = P_{od2} - P_c - f(A)_{mov}\dot{m}_{mov}^2 - \left[\frac{\lambda}{2g\rho_{ox}A^2}\right]\dot{m}_{mov}^2 \tag{12}$$

where $\left(\frac{L}{gA}\right)$ is a constant given as 0.04, P_{od2} is the HPOTP discharge pressure, $f(A)_{mov}$ is the pressure drop due to the MOV position, and $\left[\frac{\lambda}{2g\rho_{ox}A^2}\right]$ is the flow/friction loss of the supply pipe and main injector given as a constant 0.0003573. P_{od2} is a parameter, taken from sensor data. This pressure was calculated by Lozano-Tovar (Ref. 18) with an expression involving the HPOTP turbine speed rather than the actual sensor measurement. The reason for this is unclear when the data are available through sensor data. It is thought that this may be due to the use of an older engine design. The P_{od2} sensor used may not have been available to Lozano-Tovar. The expression for $f(A)_{mov}$ is given by

$$f(A)_{mov} = 0.001358\left(\frac{A_t}{A}\right)^2 \tag{13}$$

where A_t is the nominal flow area and A is the effective valve area. The ratio of these two are given by the expressions

$$\frac{A_t}{A} = \begin{cases} \left(1 - \frac{\theta}{\theta_c}\right) - \frac{1}{\pi}\sin\left[\pi\left(1 - \frac{\theta}{\theta_c}\right)\right], & for\ \theta < \theta_c \\ 0, & for\ \theta \geq \theta_c \end{cases} \tag{14}$$

$$\theta = \frac{\pi}{2}(1 - \mathrm{X}_{mov}) \text{ and } \theta_c = 2\mathrm{asin}\left(\frac{r}{R}\right) \tag{15}$$

where X_{mov} is the valve position fraction of the MOV, r is the radius of the pipe, and R is the radius of the valve ball mechanism. The ratio of $\frac{r}{R}$ is given as 0.6. For nominal operation in steady state, the MOV is fully open and X_{mov} is therefore taken as equal to one, making Equation (12) a constant, 0.001358.

The mass flow rate of the main fuel injector, $\dot{m}_{FI}$, is the remaining parameter that requires an expression. This mass flow rate is a result of a time lag in the fuel injector mass flow rate due to a small channel radius-to-length ratio in the fuel injector. The mass flow rate at the fuel injector can be calculated given surrounding conditions, but shifts due to the conditions within the injector (Ref. 18). The expression for this is given by

$$\frac{d}{dt}\,\dot{m}_{FI} = \frac{1}{\varepsilon}(\dot{m}_{fi} - \dot{m}_{FI}) \tag{16}$$

where $\dot{m}_{FI}$ is the fuel injector mass flow rate after considering the time lag, $\dot{m}_{fi}$ is the fuel injector mass flow rate before considering the time lag, and ε is the time lag in the fuel injector taken as 0.02. The mass flow rate before considering the time delay is given by

$$\dot{m}_{fi} = C_{fi}\frac{P_{FI}}{\sqrt{T_{FI}}}\sqrt{\left(\frac{P_c}{P_{FI}}\right)^{2/\gamma} - \left(\frac{P_c}{P_{FI}}\right)^{\frac{\gamma+1}{\gamma}}} \tag{17}$$

where C_{fi} is a constant taken as 19.387, P_{FI} is the fuel injector pressure, T_{FI} is the fuel injector temperature, and γ is the heat capacity ratio. This equation assumes that the fuel injector flow is not choked. If the fuel injector flow is choked, the fuel injector mass flow rate would be given by

$$\dot{m}_{fi} = C_{fi}\frac{P_{FI}}{\sqrt{T_{FI}}}\sqrt{\left(\frac{2}{\gamma+1}\right)^{\frac{2}{\gamma-1}} - \left(\frac{2}{\gamma+1}\right)^{\frac{\gamma+1}{\gamma-1}}} \tag{18}$$

with the choked condition of

$$\frac{P_c}{P_{FI}} > \left(\frac{2}{\gamma+1}\right)^{\frac{\gamma}{\gamma+1}} \tag{19}$$

If Equation (19) no longer holds, Equation (18) must be used for the fuel injector mass flow rate in place of Equation (17).

The fuel injector pressure is a measured parameter with a redundancy of two. The fuel injector temperature is not measured or given in an expression by Lozano-Tovar. However, temperatures at the fuel and oxidizer high pressure turbopumps are measured with a redundancy of four. These can be used in a mass weighted averaging to approximate the temperature at the fuel injector. The expression for this is given by

$$T_{FI} \approx \frac{r^* T_{PO} + T_{PF}}{1 + r^*} \tag{20}$$

where T_{PO} is the oxidizer high pressure turbopump discharge temperature and T_{PF} is the fuel high pressure turbopump discharge temperature. The results of this approximation show a 2 to 4 percent error compared to tabulated results given by Lozano-Tovar (Ref. 18) for all steady state operating modes.

The heat capacity ratio is given by the expression

$$\gamma = \left[1 + \frac{R(15r^* - 16)}{r^* C_{pox} + 16(1 - r^*)C_{pf}}\right]^{-1} \tag{21}$$

where R is the gas constant given as 4.37 Btu*Kmol*R^{-1}, C_{pox} is the specific heat of oxygen given as 18.26 Btu/Kmol/R, and C_{pf} is the specific heat of hydrogen given as 16 Btu/Kmol/R.

To summarize, three main dynamic equations, Equations (8), (12), and (16), represent the reduced model to be used in the Kalman filter algorithm. They describe the dynamics of the MCC pressure, flow rate through the MOV, and fuel injector flow rate and depend on measured parameters of MCC pressure, HPOTP discharge pressure, fuel injector pressure, and fuel injector temperature.

3.2 Application of Steady State Continuous Time Kalman Filter Bank to RS-25 Engine

3.2.1 Selection of State, Measurement, and Input Variables

For this study, the selected state, measurement, and input vectors are shown in Equation **(**22). A state of three parameters were chosen based on dynamic relations described in Equations (8**)**, (12), and (16). The state consisted of the MCC pressure, P_c, the mass flow rate through the MOV, $\dot{m}_{mov}$, and the fuel injector mass flow rate, $\dot{m}_{FI}$. The control inputs were seen as inputs into the system rather than control variables and were taken as the fuel injector pressure, P_{FI}, and temperature, T_{FI}. To consolidate the redundant sensor measurements, all measurements for these two control parameters were averaged by calculating their mean before use in the input vector. Finally, the measurements consisted of six sensor measurements involving two parameters, the MCC pressure and the HPOTP discharge pressure, P_{od2}. All four MCC pressure sensors and the two HPOTP discharge pressure sensors were used. It is important to note that the second P_{od2} sensor is a ground facility sensor only and is not used in-flight. This sensor is critical as the algorithm developed depends on at least two sensor measurements per measured parameter at all times. If the P_{od2} sensor was unavailable, the algorithm could not be used without some redevelopment. However, this may also suggest the inclusion of additional sensors in-flight at certain locations, such as the P_{od2}, as a possible risk-reduction strategy if necessary.

$$\boldsymbol{x} = \begin{bmatrix} P_c \\ \dot{m}_{mov} \\ \dot{m}_{FI} \end{bmatrix}, \quad \boldsymbol{y} = \begin{bmatrix} P_c \\ P_{od2} \end{bmatrix}, \quad \boldsymbol{u} = \begin{bmatrix} P_{FI} \\ T_{FI} \end{bmatrix} \tag{22}$$

3.2.2 System dynamics

The relations of such a system are given by the state-space equations

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) \tag{23}$$

$$\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{u}) \tag{24}$$

where $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})$ and $\boldsymbol{h}(\boldsymbol{x}, \boldsymbol{u})$ describe the relations of the state and input vectors to the state derivative and measurement, respectively. Here, $\boldsymbol{f}$ is given by the three differential equations in Equations (8**)**, (12), and (17), however, $\boldsymbol{h}$ is not as easily defined. One must have a relationship describing the known sensor measurements in terms of the unknown states. This is done by assuming steady state as shown in the following equation

$$\boldsymbol{h} = \begin{bmatrix} P_c \\ P_c \\ P_c \\ P_c \\ P_c + \left(f(A)_{mov} + \left[\frac{\lambda}{2g\rho_{ox}A^2}\right]\right)\dot{m}_{mov}^2 \\ P_c + \left(f(A)_{mov} + \left[\frac{\lambda}{2g\rho_{ox}A^2}\right]\right)\dot{m}_{mov}^2 \end{bmatrix} \tag{25}$$

Because the MCC pressure is both a state and measurement, there is a one-to-one relationship between these within the $\boldsymbol{h}$ function for the four sensors. The relationship of P_{od2} to state variables is solved for by assuming steady state, setting the derivative of Equation (12) equal to zero, and solving for P_{od2}. There are two of these equations within the $\boldsymbol{h}$ function as there are two sensors measuring P_{od2}.

Engine dynamics are highly nonlinear, so one must first linearize the system around a nominal value. The three matrices from the linear state space form, Equation (1), in continuous time were found by computing the Jacobian of each nonlinear function with respect to the state and input vectors as given by

$$\begin{aligned} A &= \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right)_{x_{trim}} \\ B &= \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}\right)_{u_{trim}} \\ C &= \left(\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}\right)_{x_{trim}} \end{aligned} \tag{26}$$

where each Jacobian is evaluated at the respective trim value. Trim values of the dynamic state variables, control inputs, and measurements were established based on 5 s of quasi-steady RS-25 measurement data. This is done by first taking the mean of individual sensor and control measurements over this 5 s section to determine trim values for these parameters. Next, the state variable trim vector was calculated by setting the derivative terms of Equations (8**)**, (12), and (16) to zero to establish the steady state assumption and solving the system of three equations to determine the state variable trim vector.

3.2.3 Fault Detection and Isolation

As described above, a bank of Kalman filters is applied to detect and isolate sensor faults. Each Kalman filter had an input of five sensor measurements and produced an estimate of the sensor measurements after applying the algorithm described in Section 2.1.2. These estimates were compared to the actual sensor measurements to calculate the WSSR using weights (the W_r terms in Equation (7) of 1/18 and 1/19 for MCC pressure sensors and HPOTP discharge pressure sensors, respectively). For this study, a unity detection threshold was desired (i.e., threshold value was set to 1), and the weights were manually selected accordingly. Furthermore, these weights were selected to prevent false alarms in the nominal data but remain sensitive enough to detect faults. As previously noted, it is possible to determine these weights through analytical means. Although not employed in this study, it is recommended that future work explore such methods for selecting these weights. Values for the standard deviation of each sensor were computed by calculating the standard deviation of each over a region of nominal data within the data set. A sensor fault is declared (detected) if the WSSR signal of a Kalman filter exceeds the

detection threshold for a persistence of three consecutive samples. Once a fault was detected, fault isolation was performed by examining all WSSR values and assuming the minimum signal as having the correct hypothesis. Consequently, the sensor excluded from this signal was considered faulty by the algorithm.

3.3 Fault Simulations

Actual test data from an RS-25 engine test that was nominal was used for generating simulated fault data. A data set from an RS-25 engine test operating at 109 percent of the reference MCC pressure over 80 s, sampled at 50 Hz, was selected.

Two fault types were injected into this data, bias and drift. A bias fault is defined as a signal that is a constant value above or below the correct value. For this study, a bias fault was constructed as

$$S_{faulty} = (1 + bias\%)S \tag{27}$$

where S_{faulty} is the biased sensor measurement, S is the nominal sensor measurement and $bias\%$ is the percent bias added to the nominal measurement. This value was consistently added at all time steps from the chosen fault start time onward. A notional example of a bias fault is shown in Figure 3. As shown, the faulty signal steps up to a specified percent above the nominal signal once the fault occurs. A drift fault is defined as a signal that slowly drifts from the correct value. For this study, a linear relationship was employed as defined by

$$S_{faulty} = \left(1 + \frac{t - t_o}{t_f - t_0} dr\%\right) S \tag{28}$$

where t is the current simulation time, t_o is the simulation start time of the fault, t_f is the final simulation time, and $dr\%$ is the drift percentage. This means that the sensor measurement will drift to a value away from the correct signal and will reach its maximum deviation by the end of the simulation. A notional example of a specified percent drift fault is shown by Figure 4. As shown, once the fault occurs, the faulty signal linearly increases to the specified percent above the nominal signal.

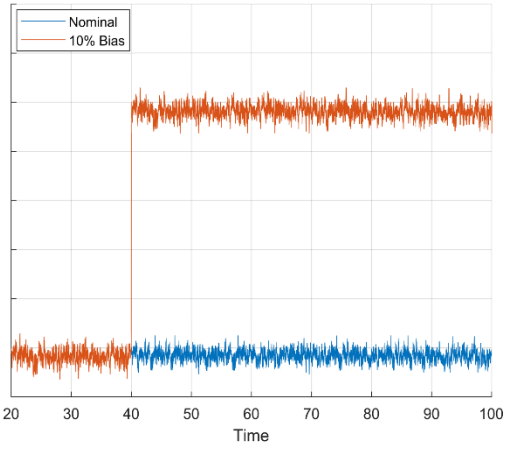

Figure 3.—Notional drift fault. Fault begins at 40s and linearly drifts to a specified percent above nominal (representative data shown).

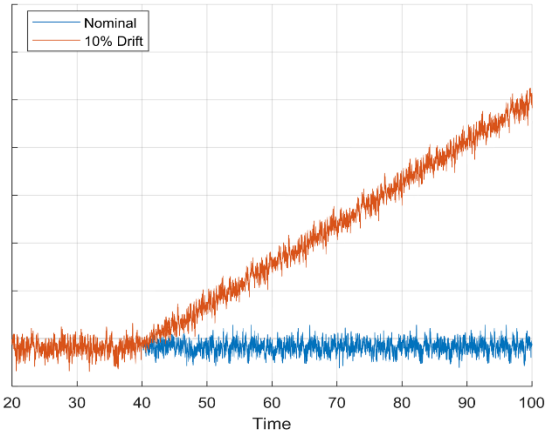

Figure 4.—Notional bias fault. Fault begins at 40s. A constant specified value is added afterwards (representative data shown).

4.0 Results and Discussion

4.1 Bias Fault Simulation Results

As stated above, simulated bias faults of various magnitudes were injected into nominal sensor data from an actual RS-25 engine test. Bias and drift percentages evaluated were 1, 5, 10, 15, 20, 25, 30, and 70 percent. These values were chosen to give a larger sample size of lower bias magnitudes. Faults were individually inserted into all six sensors. The sensors are described with their associated identification number in Table 1. The MCC pressure measurement is shown as being quad redundant, while the HPOTP discharge pressure measurement is dual redundant. These two measurements are typically used for closed-loop engine control of rocket engine propulsion systems, such as the RS-25. False readings due to sensor failures in these cases are particularly undesirable, since the engine will be controlled to them. Therefore, hardware redundant sensors are typically employed to better assure the acquisition of valid measurement data. It should be noted that the simulation approach used in this study of superimposing simulated faults onto nominal data will not be able to represent the closed-loop sensor behavior present in an actual system.

Bias faults were first injected into the nominal data in the case where all six sensors are initially free of faults and available. Sensor fault detection and isolation results for both the hardware redundancy and Kalman filter approach for this bias sensor case are shown in Table 2. As shown, both algorithms detected all faults with the exception of the hardware redundancy algorithm's inability to detect a 1 percent bias fault in sensor 5. Because sensor 6 readings were typically greater than the reading for sensor 5, adding a 1 percent bias to sensor 5 actually decreased the difference in the readings and consequently made its fault detection more difficult. Therefore, after the fault was introduced the difference between these two sensor readings remained smaller than the delta limit threshold chosen for the hardware redundancy check algorithm. The hardware redundancy check algorithm relies on this difference exceeding the delta limit threshold. As a result, this algorithm could not detect the fault. However, the Kalman filter was able to detect this fault scenario as it does not rely on this difference. In all other cases, both algorithms found the fault in 3 time steps as this was the persistence value chosen. While detection results are similar, the Kalman filter performed better because it could isolate faults in sensors 5 and 6, while hardware

TABLE 1.—SENSOR DEFINITIONS

Sensor no.	Description
1	MCC pressure 1
2	MCC pressure 2
3	MCC pressure 3
4	MCC pressure 4
5	HPOTP discharge pressure
6	HPOTP discharge pressure

TABLE 2.—BIAS RESULTS, NO DISQUALIFIED SENSORS

	Hardware redundancy		Kalman filter	
Sensor failed	% Detected	Isolated	% Detected	Isolated
1	100	Correct	100	Correct
2	100	Correct	100	Correct
3	100	Correct	100	Correct
4	100	Correct	100	Correct
5	87.5	No	100	Correct
6	100	No	100	Correct

redundancy could not. With only two sensors in this redundancy group, the Hardware Redundancy algorithm could not discriminate which sensor was faulty. This represents an example of how the Kalman filter can solve the two sensor problem.

Next, bias faults were injected into the nominal data at 40 s, assuming one of the four MCC sensors was disqualified or unavailable. Bias magnitudes in all remaining five sensors were simulated for the four scenarios of one of these MCC sensors already being disqualified. Results for this case are shown in Table 3. As shown, results are identical to the no disqualified sensor scenario. All faults were detected by both algorithms except for the 1 percent bias in sensor 5 case. Again, the hardware redundancy could not detect the fault while the Kalman filter algorithm could. Similarly, the Kalman filter correctly isolated the fault.

Finally, bias faults were injected into the nominal sensor data at 40 s, assuming two of the MCC sensors were disqualified or unavailable. In one case, sensors 1 and 2 were considered disqualified and in a second case, sensors 3 and 4 were considered disqualified. Results are shown in Table 4. Again, the hardware redundancy algorithm could not detect a 1 percent bias fault in sensor 5. The Kalman filter algorithm was able to detect this fault when sensors 3 and 4 were disqualified, but was not able to when sensors 1 and 2 were disqualified, leading to the 93.75 percent detection.

In summary, the two algorithms perform similarly when diagnosing sensor bias faults. Both algorithms can detect and isolate all faults injected into the MCC pressure data. The main difference between the two algorithms was the two HPOTP sensor scenarios. The hardware redundancy approach had a slightly lower detection rate than the Kalman filter bank, possibly due to its reliance on differences between sensor readings, and could not isolate any of the HPOTP sensor faults. The Kalman filter algorithm could detect and isolate nearly all faults in both the MCC and HPOTP sensors.

TABLE 3.—BIAS RESULTS, ONE DISQUALIFIED SENSOR

	Hardware redundancy		Kalman filter	
Sensor failed	% Detected	Isolated	% Detected	Isolated
1	100	Correct	100	Correct
2	100	Correct	100	Correct
3	100	Correct	100	Correct
4	100	Correct	100	Correct
5	87.50	No	100	Correct
6	100	No	100	Correct

TABLE 4.—BIAS RESULTS, TWO DISQUALIFIED SENSORS

	Hardware redundancy		Kalman filter	
Sensor failed	% Detected	Isolated	% Detected	Isolated
1	100	No	100	Correct
2	100	No	100	Correct
3	100	No	100	Correct
4	100	No	100	Correct
5	87.50	No	93.75	Correct
6	100	No	100	Correct

4.2 Drift Fault Simulation

Drift faults of various magnitudes were injected into the nominal sensor data at 40 s. The drift percentage values chosen were the same as the bias percentage values, but were inserted linearly over time. A similar series of tests were performed by replacing bias faults with drift faults across all scenarios. Unlike the results for the bias faults, the detection times varied significantly across the drift fault test scenarios. Therefore, graphs showing the differences in fault detection times between the two algorithms are also given.

First, drift faults of the various magnitudes were injected into the data, assuming no sensor was disqualified. The difference in fault detection times of the Kalman filter and hardware redundancy algorithms, respectively, are shown in Figure 5. As shown, the Kalman filter algorithm detected sensor faults faster than the hardware redundancy algorithm for all sensors except for sensors 2 and 6. There is also a general trend of a lower fault detection times at higher drift percentages in all cases.

Faster fault detection with increasing fault magnitude is expected as the increased fault magnitude brings the signal closer to the threshold value faster. However, the Kalman filter algorithm was not always reliable in isolating the faulty sensor as shown in Table 5, which indicates the percentage of correct isolations for each sensor fault. It did not correctly isolate the faulty sensor when sensors 1, 2, 3

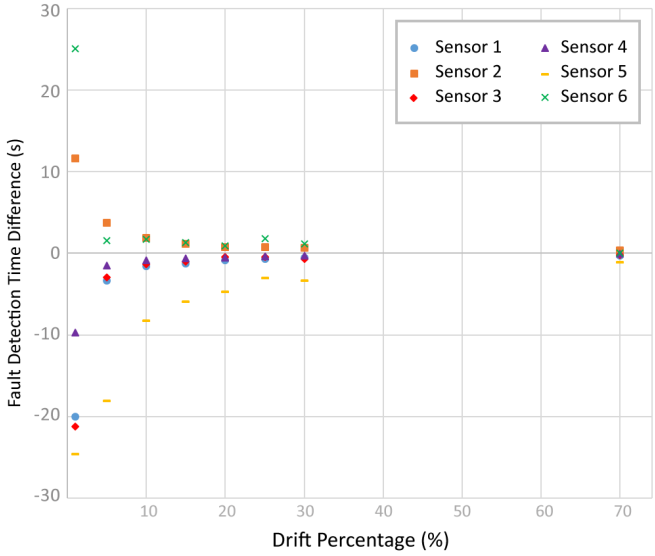

Figure 5.—Drift fault results, no disqualified sensors. Data points indicate the difference calculated for each drift percentage by subtracting the Kalman filter detection time from the hardware redundancy detection time. Negative values indicate faster detection by the Kalman filter.

TABLE 5.—DRIFT RESULTS, NO SENSOR DISQUALIFIED, CORRECT ISOLATION PERCENTAGE

Sensor failed	Hardware redundancy, percent	Kalman filter, percent
1	100	87.5
2	100	87.5
3	100	75
4	100	100
5	0	100
6	0	50

and 6 were injected with a 1 percent drift fault, when sensor 3 was injected with a 30 percent drift fault, and when sensor 6 was injected with a 5, 15, 30, and 70 percent drift fault. All other cases were correctly isolated. The missed isolation cases are possibly a result of the fact that the scalar weights used to calculate the WSSR, described in Equation (7), were not optimized, particularly with respect to the presence of specific noise in the original data for particular sensors. Extensive tuning of these weights or employing previously mentioned analytical approaches for selecting them were not investigated due to the limited scope of the current study. The hardware redundancy algorithm was able to isolate all faults correctly except in the two sensor case when either sensor 5 and 6 were given faults. Again the Kalman filter was able to isolate the fault in the two sensor case of HPOTP sensors 5 and 6 while the hardware redundancy could not.

Next, drift faults were injected into the nominal sensor data at 40 s, assuming one of the four MCC pressure sensors was disqualified. The difference in fault detection times of the Kalman filter and hardware redundancy algorithms over the various drift percentages is shown in Figure 6. The general trends are similar to the scenario where no sensors are disqualified. For this reason, the scenario where sensor 1 is disqualified is shown as an example in Figure 6, however, the other three MCC disqualification scenarios were examined as well. In all cases, the Kalman filter algorithm was quicker to detect faults in sensors 3, 4, and 5. Isolation of the faulty sensor was again not always correct. Table 6 shows the percentage of correct faulty sensor isolation taken over all four disqualified MCC sensor scenarios. As was the case with the previous set of results, the Kalman filter bank was only able to isolate a fault in sensor 6 in about half the cases. Although an improvement over the hardware redundancy algorithm, which was not able to isolate either of the faults in sensors 5 or 6, the Kalman filter bank incorrectly identified the faulty sensor as sensor five in all cases of incorrect isolation. Here again, it may be possible to improve performance with more extensive tuning or analytical selection of the scalar weights used to calculate the WSSR.

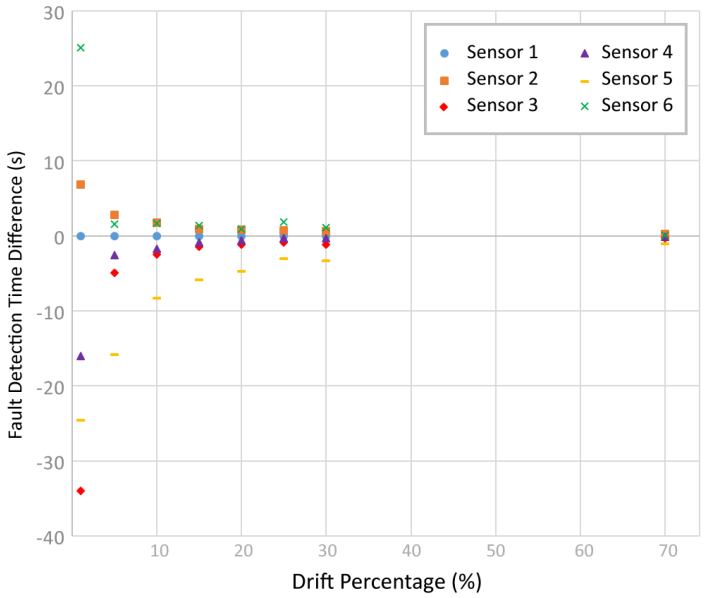

Figure 6.—Drift results, sensor 1 disqualified. Data points indicate the difference calculated for each drift percentage by subtracting the Kalman filter detection time from the hardware redundancy detection time. Negative values indicate faster detection by the Kalman filter.

TABLE 6.—DRIFT RESULTS, ONE SENSOR DISQUALIFIED, CORRECT ISOLATION PERCENTAGE

Sensor failed	Hardware redundancy, percent	Kalman filter, percent
1	93.75	100
2	100	93.75
3	100	90.63
4	100	100
5	0	100
6	0	53.13

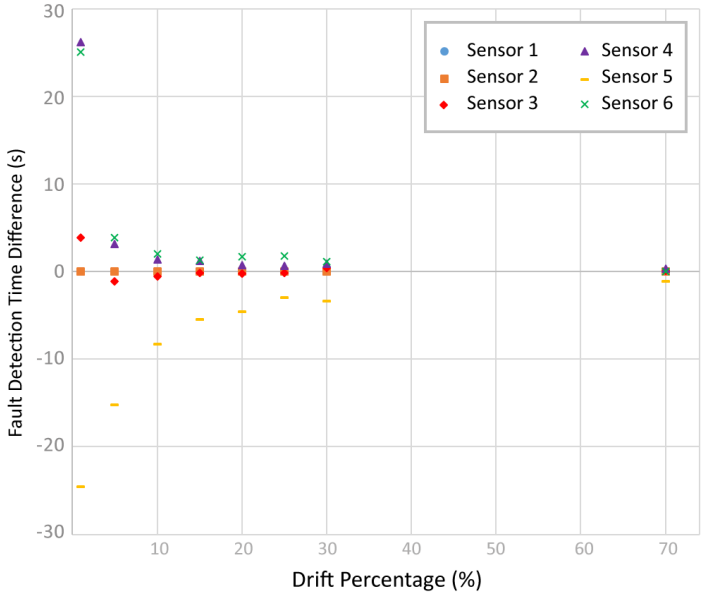

Figure 7.—Drift results, sensors 1 and 2 disqualified. Data points indicate the difference calculated for each drift percentage by subtracting the Kalman filter detection time from the hardware redundancy detection time. Negative values indicate faster detection by the Kalman filter.

Finally, drift faults were injected into the nominal engine data, assuming two MCC sensors were disqualified or unavailable. Similar to the bias fault simulations, only the two scenarios of disqualified sensors 1 and 2 or disqualified sensors 3 and 4 were examined. The fault detection time difference of the Kalman filter and hardware redundancy algorithms, respectively, are shown in Figure 7 for the sensors 1 and 2 disqualified scenario, while Figure 8 shows results for the sensor 3 and 4 disqualified scenario. In general, when sensors 1 and 2 are disqualified, the hardware redundancy algorithm is faster at detecting faults except in the case of a fault in sensor 5 or for some of the sensor 3 fault cases. However, when sensors 3 and 4 are disqualified, the Kalman filter bank is faster at detecting faults in sensors 1 and 5, but slower when sensors 4 and 6 are faulty. Though, the main advantage of the Kalman filter bank in this two sensor scenario is that it can isolate faulty sensor signals while the hardware redundancy cannot. The Kalman filter was able to correctly isolate the correct sensor fault 78.1 percent of the time again with all errors associated with faults in sensor 6, as shown by Table 7. When a drift fault was injected into sensor 6, it was often seen by the algorithm as a fault in sensor 5.

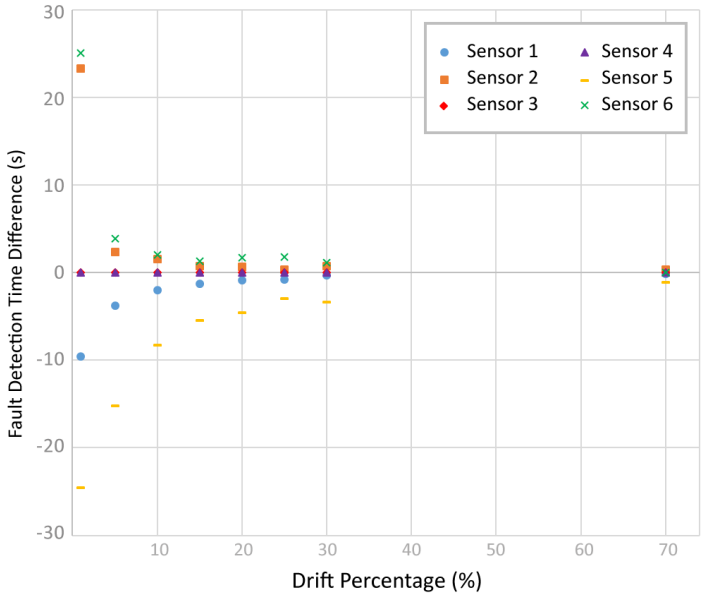

Figure 8.—Drift results, sensors 3 and 4 disqualified. Data points indicate the difference calculated for each drift percentage by subtracting the Kalman filter detection time from the hardware redundancy detection time. Negative values indicate faster detection by the Kalman filter.

TABLE 7.—DRIFT RESULTS, TWO SENSORS DISQUALIFIED, CORRECT ISOLATION PERCENTAGE

Sensor failed	Hardware redundancy, percent	Kalman filter, percent
1	0	100
2	0	100
3	0	100
4	0	100
5	0	100
6	0	78.1

In summary, results show that there are some scenarios where the Kalman filter has a faster fault detection time and others where the hardware redundancy algorithm is faster. Generally, results appear fairly similar, with the exception of the scenario of two valid sensors. The Kalman filter algorithm was able to isolate faults in both the two HPOTP discharge pressure sensors and when MCC pressure sensors were reduced to two while the hardware redundancy could not perform fault isolation in any of these scenarios.

5.0 Conclusion

In order to address the limitations of hardware redundancy-based sensor fault detection and isolation, an analytical redundancy algorithm based on a bank of Kalman filters was explored. This algorithm uses Kalman filters constructed using a set of three dynamic equations and six sensors, four MCC pressure sensors and two HPOTP discharge pressure sensors. Sensor measurements of the fuel injector pressure and two high pressure turbopump discharge temperatures were also taken as inputs into the system. The set of dynamic equations used is a subset of the RS-25 engine model given by Lozano-Tovar (Ref. 18)

and forms the basis of the Kalman filter time update. Using updates from the dynamic model and sensor measurements, the Kalman filter generates an estimate of the measured parameters that can be used to validate their values.

Fault simulations were conducted to compare the Kalman filter bank's ability to detect and isolate both bias and drift sensor faults simulated in real RS-25 test data. It was found that the Kalman filter algorithm and conventional hardware redundancy check algorithm perform similarly with respect to fault detection when presented these two types of sensor fault scenarios. In some scenarios, faults were detected faster with the hardware redundancy check, while in other scenarios, the Kalman filter bank detected the faults faster. However, the key difference between the two algorithms is their fault isolation performance. The hardware redundancy check cannot isolate a faulty sensor when there are only two valid sensors of a given type and location while the Kalman filter algorithm can. Although the bank of Kalman filter algorithm has some issues with correct isolation of faulty sensors, it is thought that further refinement of the algorithm can help address these issues. Particularly, it is recommended that future work explore optimization of the Kalman filter WSSR weights and thresholds, as well as the overall fault isolation logic applied.

References

1. Bickford, R.L., Bickmore, T.W., Meyer, C.M., and Zakrajsek, J.F., "Real-Time Flight Data Validation for Rocket Engine," *American Institute of Aeronautics and Astronautics*, 1996.
2. Wong, E., "Sensor Data Qualification and Consolidation (SDQC) for Real-Time Operation of Launch Systems," *AIAA Space Forum*, AIAA, Sept. 2016.
3. Maul, W.A., Melcher, K.J., Chicatelli, A.K., and Sowers, T.S., "Sensor Data Qualification for Autonomous Operation of Space Systems," NASA/TM—2006-214475, Nov. 2006.
4. Distler, M.E., Melcher, K.J., and Wong, E., "Investigation of Data Qualification Methods for the Two Sensor Problem," *Internship Final Report*, NASA Glenn Research Center, Aug. 2015.
5. Zarchan, P., and Musoff, H., "Fundamentals of Kalman Filtering: A Practical Approach," 2nd ed., Progress in Astronautics and Aeronautics, AIAA, Reston, VA, 2000.
6. Grewal, M.S., and Andrews, A.P., "Applications of Kalman Filtering in Aerospace 1960 to the Present," *IEEE Control Systems Magazine*, IEEE, May, 2010.
7. Merrill, W.C., Delaat, J.L., and Bruton, W.M., "Advanced Detection Isolation, and Accommodation of Sensor Failures-Real-Time Evaluation," NASA-TP-2740 19870015898, July 1987.
8. Kobayashi, T., "Aircraft Engine Sensor/Actuator/Component Fault Diagnosis Using a Bank of Kalman Filters," NASA/CR—2003-212298, March 2003.
9. Kobayashi, T., Simon, D.L., "Application of a Bank of Kalman Filters for Aircraft Engine Fault Diagnostics," NASA/TM—2003-212526, Aug. 2003.
10. Kobayashi, T., and Simon, D.L., "Evaluation of an Enhanced Bank of Kalman filters for In-Flight Aircraft Engine Sensor Fault Diagnostics," NASA/TM—2004-213203, Aug. 2004.
11. Kobayashi, T., Simon, D.L., and Litt, J.S., "Application of a Constant Gain Extended Kalman Filter for In-Flight Estimation of Aircraft Engine Performance Parameters," NASA/TM—2005-213865, Sept. 2005.
12. Simon, D.L., and Garg, S., "Optimal Tuner Selection for Kalman Filter-Based Aircraft Engine Performance Estimation," NASA/TM—2010-216076, Jan. 2010.
13. Simon, D.L., and Armstrong, J.B., "An Integrated Approach for Aircraft Engine Performance Estimation and Fault Diagnostics," *Journal of Engineering for Gas Turbines and Power*, Vol. 135, July 2013.

14. Ho, N., Lozano, P., Mangoubi, R., and Martinez-Sanchez, “Failure Detection and Isolation for the Space Shuttle Main Engine,” AIAA, 1997.
15. Taniguchi, M.H., “Failure Control Techniques for the SSME,” Rockwell International: Rocketdyne Division, Canogo Park, CA, 1988.
16. Simon, D., “Optimal State Estimation: Kalman, H_∞, and Nonlinear Approaches,” Wiley & Sons, Hoboken, New Jersey, 2006.
17. Cha, J., Ha, C., Ko, S., and Koo, J., “Application of fault factor method to fault detection and diagnosis for space shuttle main engine,” *Acta Astronautica,* Science Direct, Dec. 2015.
18. Lozano-Tozar, P.C., “Dynamic Models for Liquid Rocket Engines with Health Monitoring Applications,” Master’s Thesis, Aeronautics and Astronautics Dept., Massachusetts Institute of Technology, Cambridge, MA, 1998.