System Failure Isolation in Dynamic Systems

Dan T. Horak*
Allied-Signal Aerospace Company, Columbia, Maryland 21045

This paper presents an analysis and a systematic solution to the problem of system failure isolation in dynamic systems. System failures are related to the system matrix of a dynamic model and they change the dynamic response of the system. Failure isolation is concerned with specifying the type, size, and location of failures, following their detection. Two system failure isolation algorithms are proposed, one for failures that can be modeled as changes of parameters of the system matrix and the other for failures that do not have a simple model. The two algorithms working in parallel form a method capable of isolating system failures of either type. A flight control example and a jet engine example are used to illustrate the method.

Introduction

A FAILURE detection and isolation (FDI) system capable of providing complete coverage of failures in real systems must perform 1) failure detection, 2) sensor failure isolation, and 3) system failure isolation, all in the presence of modeling errors and noise.

Failure detection is concerned with fast and reliable detection of an anomaly without attempting to determine its details. The challenge in designing a failure detection system is detecting the smallest possible failures while preventing false alarms in the presence of noise and modeling errors that cause effects resembling failure signatures. There are several methods for averaging out the effects of noise on the failure detection process[1,2] as well as methods for dealing with modeling errors.[3]

Failure isolation is concerned with determining the details of a failure such as its type, size, and location. Failures in dynamic systems can be related to the matrices of the standard state-space representation that are affected. Sensor failures result in sensor readings that differ from the values predicted by $y = Cx$. Failures of actuators that do not have dynamics cause the system to be excited in a way different from Bu. System failures cause the system dynamics to be different from the dynamics predicted by the system matrix A.

Failure detection and isolation methods can be classified according to the type of redundancy on which their decisions are based. Sensor failures can be detected and isolated using hardware redundancy. The typical application of this method is the use of three identical sensors for the measurement of one signal. A failed sensor is detected and isolated if its reading disagrees with the other two. Hardware redundancy, however, does not apply to system failures because the replication of components other than sensors for the purpose of failure detection and isolation is usually not practical. Since the subject of this paper is system failures, we rely on analytical redundancy for failure isolation. This approach uses the system model rather than replicated hardware to isolate the failed components or subsystems.

The difficulty of isolating system, actuator, and sensor failures using analytical redundancy is directly related to the roles played by the A, B, and C matrices. The system matrix A determines the eigenstructure of the dynamic system and, therefore, its speed of response, damping, and coupling between state variables. The input matrix B and the output matrix C only determine how the inputs excite the state variables and how the outputs are formed from the state variables. If one assumes that the system matrix A is known exactly, the dynamic part of the system response can be accounted for exactly, greatly simplifying the isolation of sensor and actuator failures. This strategy does not work for system failures, however, because the dynamics of the failed system are not described by the original system matrix. Therefore, the isolation of system failures cannot be decoupled from the dynamics of the system. The fact that after the onset of failure, the most important part of the dynamic model, the system matrix, becomes invalid makes model-based system failure isolation difficult.

System failure isolation problems can be classified according to the level of coupling in the monitored systems. Chemical and power plants usually consist of many serially interconnected subsystems in which the inputs and outputs of each subsystem are measured. The determination of the health of a subsystem in these plants can be accomplished by applying range check to its outputs or by testing the energy balance across the subsystem. The challenging part of monitoring these plants is isolating the failed subsystem in minimum time without testing all of them.

This paper concentrates on the second class of isolation problems, those involving systems with strong coupling. Jet engines and flight control systems belong to this category. Although they do consist of several subsystems, these subsystems are strongly coupled as reflected in the relatively few zeros in the system matrices of their models. Additionally, because of the strong coupling, usually not all the state variables that provide the coupling between these subsystems are being measured. Isolation methods for these systems rely on analytical redundancy, which is made possible by the availability of accurate models, unlike the situation in most chemical plants.

The two main approaches to system failure isolation in systems with strong coupling that have been proposed in the past are parameter identification and failure detection filters. The first approach is based on identifying the parameters of the monitored system in real time and comparing their values to those identified during unfailed operation.[4] This approach, however, performs only as well as does the identification algorithm on which it is based. In most real systems, unmodeled effects such as parameter variations, high-order dynamics, and nonlinearities cause identification errors that are larger than the effects of the failures to be isolated. These errors limit the use of this failure isolation approach to simple cases.

Failure detection filters[5] are observers with gains selected so that the directions of the innovation vectors generated by the observers can be used to isolate failures. The advantage of

Received March 10, 1989; revision received July 5, 1989.

*Member Technical Staff, Aerospace Technology Center.

failure detection filters is their compactness; a single filter can distinguish between several failures. Their drawback is high sensitivity to model inaccuracies that interfere with the proper operation of the finely tuned observers, as concluded in recent reexaminations of this method.[6,7]

The main contribution of this paper is a method for isolating system failures in dynamic systems. The method avoids the use of finely tuned algorithms that work in idealistic simulations but fail in the presence of modeling errors that are unavoidable in real systems. The method consists of two algorithms executed in parallel corresponding to two possible cases of system failures: structured and unstructured. The isolation method uses the reachable measurement intervals (RMI) algorithm[3] for systems with modeling errors as a failure detector, further improving its robustness.

Isolation of System Failures

The majority of FDI applications and papers to date have concentrated on sensor failures. This is the simplest case of failure isolation because it can be accomplished by parallel execution of several failure detection algorithms, each driven by different groups of sensors. The failed sensors are isolated if all the algorithms using them indicate a failure, and all the algorithms not using them do not indicate a failure.[3,8]

The isolation of failures of actuators that do not have dynamics is also relatively easy because they do not change the dynamics of the system. The models of these actuators and of their failures are contained in the matrix B. A recently introduced failure isolation method is capable of handling both actuator and sensor failures in a unified way that makes their isolation very simple.[9]

Many actuation devices, however, have nonnegligible dynamics and must be included in the system matrix A. In this case only the static gains of the actuators appear in the matrix B. Even gains of actuators without dynamics appear frequently both in the A and the B matrices because their effect on the system is described by $G(u-x)$, where G is gain, x is a state variable, and u is an input. Whereas Gu is a term corresponding to the input matrix B, $-Gx$ is a term corresponding to the system matrix A.

It is convenient to include all the parameters related to the actuators in the system matrix A and thus to transform actuator failure isolation into system failure isolation. The transformation is accomplished by augmenting the state vector with additional state variables $\boldsymbol{v}$ that are proportional to the system inputs $\boldsymbol{u}$ within the bandwidth of the system. The new state variables have dynamics defined by $\dot{\boldsymbol{v}} = -K\boldsymbol{v} + K\boldsymbol{u}$, where $\boldsymbol{u}$ is the system input vector and K is a diagonal matrix of large gains. In the augmented system, terms of the type bv in the new system matrix replace the original input terms bu. The input matrix of the augmented system is K, which is known exactly and does not contain any system parameters. The system augmentation transforms the actuator failure isolation problem into system failure isolation, which we solve next.

Our goal is a system failure isolation method that works well with inexact models. Modeling errors cause two types of problems for failure isolation. First, they set the fundamental limits of failure isolability. Model-based failure isolation is based on comparing the measured response to a simulated response. Therefore, failures that cause system response changes that are smaller than the differences between responses caused by the modeling errors cannot be isolated reliably. This isolability limit is independent of the isolation method used and can be reduced only by more accurate modeling. Problems of the second type arise due to the specific characteristics of the isolation methods, such as the high sensitivity to modeling errors caused by the use of finely tuned observers in the failure detection filter case.[6] These problems further reduce the size of the smallest reliably isolable failure, but they are avoidable.

Our approach toward realizing a system failure isolation method that works well with real systems is to minimize the second type of problems by minimizing the complexity of the processing employed by the isolation method. We do so by segmenting the system failure isolation problem into multiple subproblems, each solvable via a direct comparison of measured and simulated responses.

The first step toward segmenting the general system failure isolation problem is classifying the failures as structured or unstructured. If a failure has a structure that can be anticipated and modeled as changes of parameters of the original model, it can be isolated by testing for agreement between the measurements from the monitored systems and the outputs of a simulation that includes the effects of a hypothetical failure. One such test can be performed for each failure hypothesis and thus provide the second level of problem segmentation. If a test yields agreement, the corresponding failure hypothesis is correct, and the failure has been isolated. In this case the structured failure has also been identified; i.e., its size has been determined.

Failures that do not have a specific structure cannot be identified because even a single component could fail in infinitely many random, nonlinear, and time-varying ways. The goal in this case is isolating the failure to within a subsystem of the model of the unfailed system. This can be accomplished by partitioning the system into two subsystems, one assumed to include the failure and the other assumed to exclude it. A failure is isolated to the subsystem assumed failed if the overall system tests failed and the subsystem assumed unfailed tests unfailed.

A complete failure isolation system must execute algorithms for isolation of structured and unstructured failures simultaneously because the failure type is not known prior to its isolation. We trigger the failure isolation algorithms with a failure detection algorithm in order to enhance their robustness with respect to modeling errors and noise. Therefore, we structure the complete FDI system as shown in Fig. 1. Note that prior to applying the isolation algorithms, the failure must be classified as a sensor failure or a system failure. A failure that affects more than one measurement can be classified as a system failure because the simultaneous failure of several sensors is rare. A failure that affects only one sensor can usually be classified as a sensor failure. In some systems, however, a localized system failure may affect only one measurement significantly, opening the possibility for erroneous classification. The response to failures of such systems must be investigated, and case-specific logic that prevents such errors must be built into the failure-type classifier.

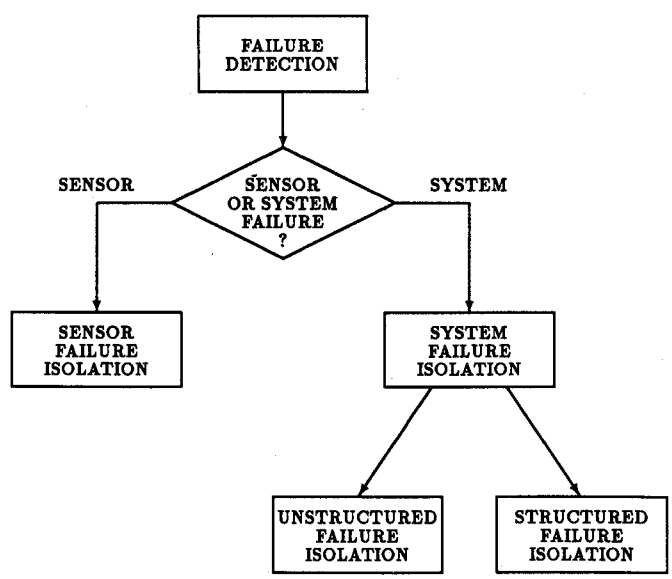

Fig. 1 A complete FDI system.

Isolation of Structured System Failures

Our method for isolating structured system failures does so by matching the measurements from the monitored system to simulated measurements corrected for the effect of the failure.[10] The failure is isolated if the corrected simulation matches the measured response.

Let the unfailed system be described by the nth order model:

$$\dot{x}_0 = Ax_0 + Bu \tag{1a}$$

$$y_0 = Cx_0 \tag{1b}$$

and the failed monitored system be described by

$$\dot{z} = (A + rF)z + Bu \tag{2a}$$

$$y = Cz \tag{2b}$$

where y is a vector of q linearly independent measurements. It is useful to define an n-vector y_m in which the ith component is the ith state variable if it is being measured, or zero if it is not being measured. For example, in a fifth-order system with the first and fourth state variables measured, it is $y_m = [z_1\ 0\ 0\ z_4\ 0]^T$. Note that both the number of measurements and the number of measured state variables can be smaller than n.

In Eq. (2), F is an $n \times n$ matrix that specifies the failure structure. If the failure can be modeled as a change in the value of one entry of the matrix A, the matrix F has a one at that location, and all its other entries are zero. If the failure of a component affects more than one entry of the matrix, numbers reflecting the relative effects of the failure on the entries are placed at the corresponding locations.

The constant r determines the size of the failure. Failure detection filters[5] attempt to isolate failures with the size r allowed to vary arbitrarily with time. Our failure model is much more restrictive, but it still covers most practical situations. The benefit derived from using a constant failure size is an isolation method that is robust with respect to modeling errors, unlike detection filters. Note that neither F nor r are known prior to the isolation of the failure. However, a set of p failure structure matrices F that describe the possible failure modes of the system is available. The failure isolation is based on the following p simulations:

$$\dot{x}_i = Ax_i + Bu + r_iF_iy_m, \qquad i = 1,...,p \tag{3}$$

in which the nominal model is excited by an input corresponding to the effect of the ith hypothetical failure. If the values of r_i and F_i correctly describe the failure, i.e., $r_i = r$ and $F_i = F$, then the outputs of the ith simulation must equal the corresponding measurements. Note that only failures that affect parameters in columns corresponding to the measured state variables can be isolated using this method. Fortunately, in many systems of interest including jet engines and flight controls, many state variables are routinely measured.

The determination of the failure size r is facilitated by dealing with deviations of state variables from their nominal values rather than with the variables themselves. Consider the difference between the state vector of the monitored system and the nominal state vector obtained by subtracting Eq. (1) from Eq. (2):

$$\dot{e}_z = Ae_z + rFz \tag{4a}$$

$$y_z = Ce_z \tag{4b}$$

where $e_z = z - x_0$. Also consider the differences between the state vectors of the simulations given by Eq. (3) and the nominal state vector, Eq. (1):

$$\dot{e}_i = Ae_i + r_iF_iy_m, \qquad i = 1,...,p \tag{5a}$$

$$y_i = Ce_i \tag{5b}$$

where $e_i = x_i - x_0$. Since x_0 is a known quantity, an agreement between Eqs. (4) and (5) implies an agreement between Eqs. (2) and (3).

The selection of the correct failure hypothesis is made as follows. The unfailed system simulation and the p failure hypothesis simulations with $r_i = 1$ are executed continuously using zero-order hold approximations of Eqs. (1) and (5), respectively. The p simulations produce y_{i1}, which is y_i computed with $r_i = 1$. The values of y_z are computed by subtracting the simulated nominal outputs y_0 from the measurements y. For each failure hypothesis, there is a failure size r_i that minimizes the sum of the squares of $r_iy_{i1} - y_z$ in the processing window. After N sampling periods, this failure size is given by least squares as

$$r_i = \sum_{k=1}^{q} w_k^2 \sum_{j=1}^{N} y_{i1kj}y_{zkj} \Big/ \sum_{k=1}^{q} w_k^2 \sum_{j=1}^{N} y_{i1kj}^2 \tag{6}$$

where y_{i1kj} is the kth output of the ith hypothesis simulation with $r_i = 1$ at the jth point in the processing window. The equation is written for the case where one r_i is found that minimizes the weighted sum of errors of all the measurements. The weights w_k make the consideration of measurements of low magnitudes possible as well as the selective amplification of the influence the measurements have on r_i. One useful set of weights is the inverses of the rms values of the measurements. These weights assign equal influence to all measurements.

The failure isolation is based on matching the measured response to the simulated response of the failed system. The average matching error for the ith hypothesis is

$$g_i = \left[(1/Nq) \sum_{k=1}^{q} w_k^2 \sum_{j=1}^{N} (r_iy_{i1kj} - y_{zkj})^2 \right]^{1/2} \tag{7}$$

The hypothesis with the smallest g_i reconstructs the failure best, and its F_i and r_i specify the isolated failure structure and size, respectively.

It is also possible to apply least squares to each measurement individually by using q times Eq. (6) with $k = 1$. When this is done, the average deviation of the q estimated failure sizes is another indicator of hypothesis correctness. The average deviation is given by $(\Sigma|r - \bar{r}|)/(q|\bar{r}|)$ where q is the number of the individually computed failure sizes and $\bar{r}$ is their average. In the ideal case, if the matching error for a hypothesis is much smaller than those for other failure modes and the average deviation of the r is small, the algorithm produces a clear and reliable isolation decision. In the worst case, when the smallest matching error is only slightly smaller than the other errors and the average deviation is large, a clear failure isolation decision cannot be made. Such a situation develops if the correct failure structure is not being considered or if the failure is more complex than a parameter change.

The failure isolation method is based on testing if one out of several hypothetical failure modes can be scaled to match the effects of an actual failure. The determination of the scaling factor, the failure size r, is done by a one-step application of Eq. (6). Although technically speaking this process can be called identification, it is much simpler and much more robust to unmodeled effects such as nonlinearities, high-order dynamics, and parameter variations than failure isolation via parameter identification. In the latter method all the parameters of the model are estimated in real time and compared to their values estimated during unfailed operation of the system, which is a much more difficult problem than estimating a single size for a failure with a known structure.

The failure isolation method is not a reliable failure detection method. It makes isolation decisions even when failures are not present and interprets modeling errors and noise as failures. Therefore, it must operate together with a reliable

failure detection method. We use the RMI algorithm for systems with modeling errors[3,11] as a failure detector because of its optimal handling of model inaccuracies. Initially only the RMI algorithm runs. As soon as RMI suspects a failure, which happens when a measurement crosses its RMI bounds, it triggers the isolation process. The RMI algorithm continues running even after it issues the alarm and it re-evaluates its initial decision as more data become available. If the initial alarm is not supported by the additional data, the isolation method is stopped and reset. This mode of operation yields the fastest possible failure isolation.

The RMI concept can also be used to estimate failure isolability in the presence of modeling errors. RMI represents modeling errors by parameter tolerance matrices that can be estimated experimentally.[12] If all the entries of the tolerance matrix that correspond to the system matrix are larger than the entries of the failure matrix rF in Eq. (2), that failure cannot be isolated reliably. The mismatch between measurements and their estimates in this case can be due to unmodeled parameter variations that are not failures. When the failure isolation process is being triggered by the RMI failure detection algorithm, however, false isolation due to this effect is automatically avoided. The RMI algorithm will not start the isolation process if the mismatch can be attributed to modeling errors.

Once the isolation process has been triggered by the failure detection algorithm, the length of the processing window increases with time. Therefore, the isolation decision becomes more accurate with time. However, this is only true for time-invariant failures. Failures with time-varying sizes will not be isolated correctly because Eq. (6) will attempt to describe them by a fixed size. An alternative mode of operation uses a finite-length processing window. Once the number of samples collected since the onset of failure reaches a predetermined value, the sample size becomes fixed. For each additional sample added to the processing window, the oldest sample is removed from it.

The size of the finite-length processing window is determined so as to achieve a reasonable compromise between failure size tracking capability and isolation accuracy. Tracking the size of time-varying failures requires a short window. The algorithm cannot track parameter changes that vary significantly during a period equal to the length of the processing window. Isolation accuracy, on the other hand, requires the processing window to be long, in order to reduce the effect of noise.

The number of measurements also influences the selection of the processing window length. If only one is available for use in Eqs. (6) and (7), a longer window is required because an incorrect failure mode with an appropriate size can resemble the failure effect on a single measurement. With more measurements a shorter window can be used, because an incorrect failure mode is unlikely to affect several measurements the same way as the actual failure.

All the computations related to the failure isolation algorithm can be performed recursively by updating the summation terms in Eqs. (6) and (7) during each sampling period. Therefore, the computational load of the isolation algorithm per failure hypothesis is only slightly more than that required to simulate Eq. (5).

The structured failure isolation method is illustrated using a seventh-order longitudinal dynamics model of the AFTI F-16 aircraft[11,13] described in the Appendix. The example demonstrates the capability of the algorithm to distinguish between failures of the elevator and the flaperon, which have a similar effect on the aircraft. The failure modes are

Battle-damaged elevator: The decrease in the elevator area is modeled by a proportional decrease of the corresponding stability derivatives. The failure structure matrix F_1 is all zeros except for $f_1(2,4) = -17.25$ and $f_1(3,4) = -0.169$.

Battle-damaged flaperon: The failure structure matrix F_2 is all zeros except for $f_2(2,5) = -1.577$ and $f_2(3,5) = -0.2518$.

In the simulation of the aircraft, we applied parameter variations of $\pm 10\%$ to the seven aerodynamic stability derivatives of the model, to account for the uncertainty associated with these parameters. These are terms 2–5 in row 2 and terms 3–5 in row 3 of the matrix A from Eq. (A1). These variations caused differences of about 30% between the measurements and the measurements simulated with the nominal model used for failure isolation. We also excited rows 2–5 of Eq. (A1) by random inputs to simulate the effects of wind gusts and structural vibrations and added noise to the measurements. The magnitudes of the independent Gaussian noise sources were selected so as to produce rms noise/signal ratios of 10% in all the measurements.

During the simulation the aircraft performs pitch-pointing maneuvers and vertical translation maneuvers. The sampling period of the simulation and of the FDI algorithms is 20 ms. We first simulated the case of a battle-damaged elevator. The failure is $-0.3F_1$, representing a 30% reduction in the area of the elevator. It is introduced at time $t = 1$ s. The RMI algorithms[3,11] that monitor the pitch angle and the angle of attack first suspect a failure at time $t = 1.04$ s, and they trigger the isolation process. This initial diagnosis of the RMI algorithms is supported by subsequent measurements. The isolation method uses a fixed-length processing window that stays at 4 s after 4 s of data are collected. The isolation is based on the pitch angle and the angle of attack measurements, both weighted with $w = 1$.

The matching errors of the two failure hypotheses, computed using Eq. (7), are shown in Fig. 2. The failure mode is correctly isolated as F_1 in about 0.25 s. Prior to that time, the startup transient caused by the short processing window prevents reliable isolation. The correct failure hypothesis produces matching errors that are about 2.5 times smaller than those produced by the incorrect hypothesis 0.5 s after the onset of failure. The ratio increases to six 1.5 s after the onset

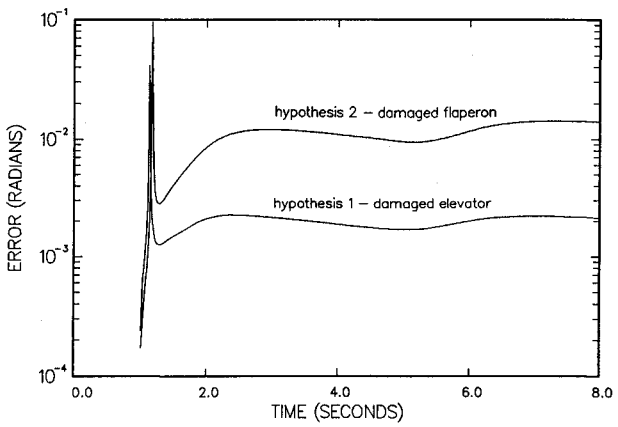

Fig. 2 Matching errors for two failure hypotheses (battle-damaged elevator).

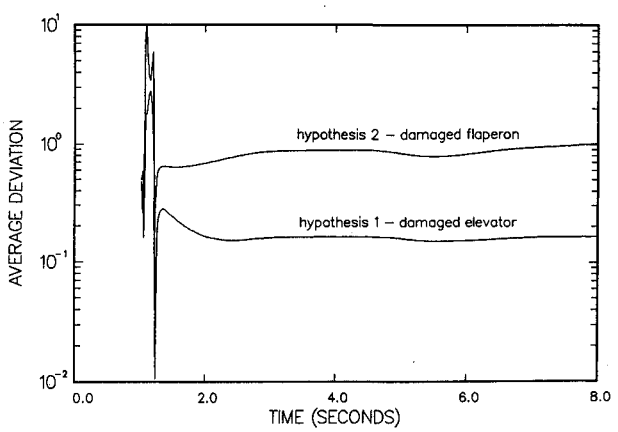

Fig. 3 Average deviations of elevator failure size estimates.

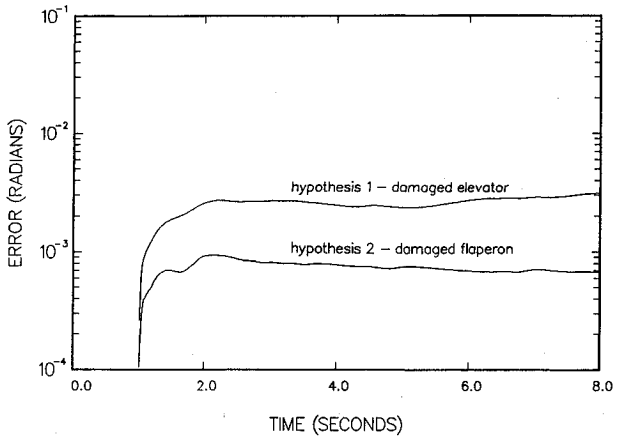

Fig. 4 Matching errors for two failure hypotheses (battle-damaged flaperon).

of failure. Similar ratios exist between the average deviations of the correct failure hypothesis and those corresponding to the incorrect hypothesis, as seen in Fig. 3.

The estimated size for the isolated failure mode (hypothesis no. 1) converges to -0.25 in about 1.5 s after the onset of failure. The error with respect to the actual size of $r_1 = -0.3$ is due to the modeling errors. As these are reduced, the estimated failure size approaches the actual value.

We then simulated a battle-damaged flaperon specified by $-0.3F_2$, i.e., a 30% reduction in flaperon effectiveness. Figure 4 shows the matching errors for the two hypotheses. The errors become two times smaller for the correct failure hypothesis in about 0.2 s. The ratio becomes about three 0.5 s after the onset of failure. The estimated failure size converges to $r_2 = -0.32$ in about 1.5 s. Note that the flaperon failure affects the aircraft less than the elevator failure, making its isolation more difficult. Therefore, the matching errors ratio is smaller in the flaperon failure case. The battle-damaged elevator and flaperon examples demonstrate the accuracy and the speed of the failure isolation method in the presence of modeling errors and noise.

In the examples we have assumed that the vector of measured state variables is $y_m = [0\ 0\ 0\ z_4\ z_5\ 0\ 0]^T$. These measurements allow the isolation of both failures. The elevator failure is reconstructed using z_4, and the flaperon failure is reconstructed using z_5. The matching errors and the average deviations were computed using the pitch angle and the angle-of-attack measurements. These measurements happen to be state variables in this case, but do not have to be state variables in general.

We also simulated failures of the elevator and flaperon actuators.[10] These failures were isolated as fast and reliably as battle-damaged control surfaces.

Isolation of Unstructured System Failures

System failures that do not have simple models cannot be isolated to the point of specifying their details. They can only be isolated to the extent of specifying their location in the monitored system.

We have developed an effective method for isolating unstructured system failures.[14] The method is capable of determining the subsystem that contains the failure. It is based on partitioning the system into two subsystems, one of which is assumed to include the failure and the other is assumed to be unfailed. The failure is isolated if these assumptions are shown to be correct. The test correctness is based on a simulation of the unfailed subsystem, for which a model is available even after the onset of failure. The simulation is excited by measurements from the monitored system to account for the effects the failure has on the unfailed subsystem.

The failure isolation method is based on the following partitioning of the standard state-space model of the system:

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\end{bmatrix} = \begin{bmatrix}A_{11} & A_{12}\\ A_{21} & A_{22}\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} + \begin{bmatrix}B_1\\ B_2\end{bmatrix}u \tag{8}$$

$$\begin{bmatrix}y_R\\ y_T\\ y_F\end{bmatrix} = \begin{bmatrix}C_{R1} & C_{R2}\\ C_{T1} & C_{T2}\\ C_F & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix} \tag{9}$$

For clarity of presentation and without loss of generality, it is assumed that the system is partitioned so that its last p state variables x_2 correspond to the subsystem that contains the failure. The state variables x_1 correspond to the unfailed part of the system. Three types of outputs are shown in Eq. (9). Outputs with subscript R are used for deriving unmeasured state variables. Outputs with subscript T are used for testing of failure location hypotheses. Finally, subscript F designates outputs used to shape the response of the isolation algorithm.

The simplest possible situation occurs when the entire state vector of the failed subsystem is being measured. In that case the unfailed subsystem can be simulated using

$$\dot{x}_1 = A_{11}x_1 + A_{12}x_2 + B_1u \tag{10}$$

where x_2, the measured output of the failed subsystem, is being treated as an additional input to the unfailed system. This situation corresponds to chemical and power plants in which the inputs and outputs to all the subsystems are usually measured and the subsystems are stable. The testing of each subsystem for failures in this case is straightforward.

In most systems, however, not all the state variables in x_2 are being measured, and there are not enough outputs to compute the state variables by inverting the output matrix C. State estimation by observers cannot be used either, because the subsystem contains an unknown failure. In this case x_2 can be derived from the measurements y_R as follows. Let the number of measurements in y_R be equal to the number of state variables in x_2, the failed subsystem, and let C_{R2} be invertible. Then the first row in Eq. (9) can be solved for x_2:

$$x_2 = C_{R2}^{-1}(y_R - C_{R1}x_1) \tag{11}$$

Eq. (11) can be substituted into Eq. (10) to get

$$\dot{x}_1 = (A_{11} - A_{12}C_{R2}^{-1}C_{R1})x_1 + B_1u + A_{12}C_{R2}^{-1}y_R \tag{12}$$

A simulation based on Eq. (12) accounts for the failure effects due to the failed subsystem via the measurements y_R. Therefore, it produces estimates of the state variables of the unfailed subsystem that agree with the corresponding state variables in the monitored system, if the failure is indeed contained in x_2. Since the individual state variables are not measured in general, y_T, a subset of the measurements, is used for testing the assumption. The measurements from the monitored system are compared to the reconstructed measurements obtained by combining the second row of Eq. (9) and Eq. (11). The reconstructed measurements are

$$y_T = (C_{T1} - C_{T2}C_{R2}^{-1}C_{R1})x_1 + C_{T2}C_{R2}^{-1}y_R \tag{13}$$

The dynamic system formed to isolate the failure, Eq. (12), may have undesirable dynamics or even be unstable. Its eigenvalues are determined by the selection of the subsystems and the outputs used for isolation. The eigenvalues can be controlled to some extent by selecting y_R, but not to the point of guaranteeing stability. To make this failure isolation algorithm usable for all systems, it is desirable to add a stabilizing term to Eq. (12). This term allows the arbitrary placement of the eigenvalues without affecting the performance of the algorithm as a failure isolator.

The stabilizing term is $K_F(y_F - C_F x_1)$, which is nominally zero according to the third row of Eq. (9). The modified equation is

$$\dot{x}_1 = (A_{11} - A_{12}C_{R2}^{-1}C_{R1} - K_F C_F)x_1 + B_1 u + A_{12}C_{R2}^{-1}y_R + K_F y_F \quad (14)$$

where K_F is a matrix of gains designed so as to produce a stable and well-behaved algorithm. The determination of the gains can be done with pole-placement algorithms used for observer design. This stabilizing term can also be used in Eq. (10) if the matrix A_{11} is unstable or too oscillatory.

The isolation algorithm relies on the RMI failure detection algorithm to detect the failure in the presence of modeling errors and noise, similarly to the algorithm for structured failures. It executes several simulations based on Eqs. (14) and (13) all the time, each one corresponding to a failure location hypothesis. The measurements y_T produced by these simulations are not used until the detection of a failure. Immediately following the detection, the algorithm starts computing the relative error between the measured and the estimated outputs y_T for each hypothesis. We define the relative error as the ratio between the rms of the output differences and the rms of the corresponding measurements. After N sampling periods, the relative error for a hypothesis is given by

$$h = \frac{\left[\sum_{k=1}^{m} w_k^2 \sum_{j=1}^{N} (y_{Tkj} - y_{TMkj})^2\right]^{1/2}}{\left[\sum_{k=1}^{m} w_k^2 \sum_{j=1}^{N} y_{TMkj}^2\right]^{1/2}} \quad (15)$$

where m is the number of measurements in y_T, y_{TM} is the measured counterpart of y_T, and subscript kj designates the kth output at the jth point in the data window. The weights w_k scale the measurements to allow equal influence of all of them on the combined error.

The hypothesis with the smallest relative error h isolates the failure to within the subsystem it assumes to be failed. To prevent incorrect isolation in cases when none of the hypotheses is correct, the smallest relative matching error must be a small fraction, such as 0.2 or less, before a failure is declared isolated. An alternative, less compact, but frequently more reliable isolation criterion compares the individual relative errors for all the measurements rather than their weighted sum. This approach avoids the danger that, because of incorrect weights w_k, some measurements may not contribute to the isolation decision. The individual relative errors are given by Eq. (15) with $m = 1$.

When the system is unfailed, all the hypotheses show small relative errors. After the onset of failure, the errors of the incorrect hypotheses increase, whereas the error of the correct one remains small. We start the computation of the errors only after the detection of a failure in order to include in the data window only points that carry the failure information. Otherwise the incorrect failure hypotheses are biased toward showing small errors and may cause incorrect isolation.

The data window is N points wide to prevent incorrect isolation due to momentary effects of noise. After the detection of a failure, one can start computing the errors even before the window fills, or one can wait until N points are available. The first approach yields a faster isolation but is susceptible to noise. The second approach is robust with respect to noise but has an isolation delay of N sampling periods. Since Eq. (15) can be evaluated recursively, it is possible to let the data window grow indefinitely without increasing the computational load. A long window reduces the sensitivity to noise, but it hinders the detection of failures of duration shorter than the window. The most appropriate window length for a specific application is determined by the response of the system and its failure modes.

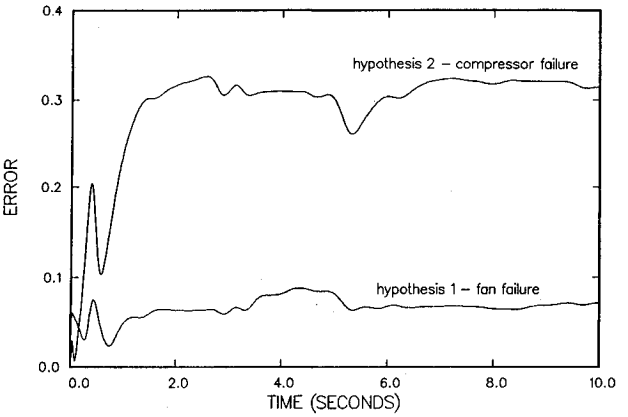

Fig. 5 Relative errors (fan failure).

The relative error concept of Eq. (15) is used here as a simple failure detection algorithm. This failure detection algorithm is applied to subsystems of the monitored system in order to isolate the failure. The isolation method, however, is not limited to such simple failure detection schemes. Virtually any failure detection algorithm can be used instead. If the RMI failure detection algorithm[3] is applied to the subsystems, the selection of the failed subsystem becomes robust with respect to modeling errors. This application of RMI is in addition to the RMI algorithms that monitor the entire system and trigger the isolation process.

The isolation method for unstructured system failures has been presented in a form suitable for linear, time-invariant systems. However, since the method is based on straightforward simulations of subsystems, it can be extended to handle nonlinear and time-varying systems that can be partitioned into subsystems. In the extended form, the simulations described by Eqs. (12) and (14) become nonlinear and time varying but still computationally straightforward if the appropriate models are available.

The failure isolation method is illustrated using a simulation of a hypothetical turbofan engine.[15] We use a linearized model in this example and thus neglect the effects of the failures on the steady-state characteristics of the engine. The model state vector is fan speed, compressor speed, burner-exit slow-response temperature, and fan-turbine inlet slow-response temperature. The measurements are fan speed, compressor speed, and burner pressure. The details of the linearized model of the engine are in Ref. 3.

The engine input is main-burner fuel flow approximated by colored noise, which represents the inputs generated by the engine controller during flight. The response of the unfailed engine to this input is shown in Ref. 3. The failures we considered were minor mechanical damage to the fan and the compressor. They were simulated by time-varying parameter changes and additive random inputs in the first and second rows of the engine model, respectively. The failures caused variations of up to 20% in the fan and compressor speeds compared to the response of the unfailed engine. Note that since we use a linear model, these speeds are the deviations from steady-state values and not the absolute speeds.

To isolate the fan failure, which is related to the first row of the model equations, we use the burner pressure measurement for the generation of the fan-speed state variable via Eq. (11). Since the burner pressure measurement has a nonzero direct transmission term, Du is subtracted from y_R before applying Eq. (11). The correctness of the failure location hypothesis is tested on the compressor speed measurement. Compressor failures are isolated by simulating the fan-speed measurement using the burner pressure measurement for the generation of the compressor speed state variable. The eigenvalues of the compressor failure isolator are all real and do not require augmentation. A pair of eigenvalues of the fan failure isolator have a damping ratio of 0.22. A stabilizing term, Eq. (14), is

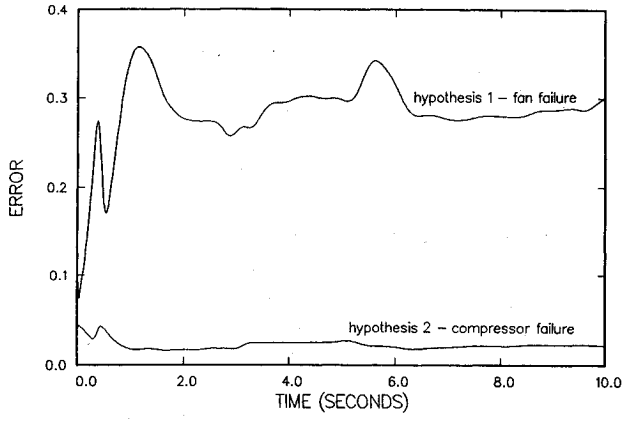

Fig. 6 Relative errors (compressor failure).

used to place the isolator poles at -6, -7, and -8 to prevent oscillatory response.

Figure 5 shows the relative errors for hypothesis no. 1, fan failure, and hypothesis no. 2, compressor failure, with a fan failure present. The isolation process starts immediately after the failure has been detected, and the algorithm uses a data window that increases indefinitely, starting with one point. The sampling time is 0.02 s. It can be seen that the algorithm reaches a correct decision in 0.2 s and after 0.7 s shows an error about four times smaller for the correct hypothesis (no. 1) than for the incorrect one.

Figure 6 shows the results of a compressor failure simulation. It shows that the compressor failure hypothesis, no. 2, produces smaller relative errors starting at the first time step following failure detection. Its error is more than ten times smaller than the error of the incorrect hypothesis after 0.8 s. The relative error settles to about 0.025.

Further simulations showed that the isolation accuracy and speed are independent of the details of the failures as long as their effect on the engine response has similar magnitude. Failures modeled as parameter variations, random additive signals, and products of state variables were all isolated as fast and accurately as the failures in the examples.

In a complete FDI system, both structured and unstructured failure hypotheses must be considered simultaneously because the failure type is not known prior to its isolation. If a structured failure hypothesis yields a good agreement, its assumed mode and estimated size identify the failure. If an unstructured failure hypothesis that agrees with the location of the isolated structured failure is also being tested, it can provide further support for the isolation decision.

In cases when none of the structured failure hypotheses yield reasonable agreement, the details of the failure cannot be determined. In these cases the unstructured failure isolation algorithm can be used to isolate the subsystem which contains the failure.

Neither one of the algorithms will isolate the failure if it cannot be anticipated by the FDI system designer and its effects are spread throughout the system. Failure isolation in these cases is impossible with our approach, or with any other approach, because the failure changes the system so drastically that the model of the unfailed system cannot be used as a reference for model-based failure isolation.

Conclusions

The system failure isolation problem in a dynamic system has been analyzed. Two effective methods for system failure isolation have been developed corresponding to two classes of system failures: structured and unstructured. The isolation method for structured system failures selects the most likely failure mode out of a supplied menu of possible failures and estimates the size of the failure. The method for isolation of unstructured system failures isolates the failure to a subsystem of the system model. It is based on testing failure location hypotheses using simulations of subsystems that are assumed to exclude the failure.

Both isolation methods employ multiple versions of simple algorithms for the testing of their failure hypotheses. These algorithms are based on direct comparisons of measured and simulated responses and are not sensitive to modeling errors. Furthermore, the isolation methods are triggered by the reachable measurement intervals failure detection algorithm for systems with imperfect models, thus further improving their robustness with respect to modeling errors.

The computational load required for the testing of each failure mode hypothesis or failure location hypothesis is approximately equivalent to that required for simulating the system. Therefore, when implemented in real time on today's microprocessors they can be used to isolate failures in large aerospace and industrial systems. The performance of the system failure isolation methods has been demonstrated on a simulated aircraft flight control system and a simulated turbofan engine, with excellent isolation accuracy and speed results.

Appendix

The seventh-order model of the longitudinal dynamics of the AFTI F-16 aircraft[10,15] we use is specified by

$$A = \begin{bmatrix} 0.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.0 & -0.8693 & 43.22 & -17.25 & -1.577 & 0.0 & 0.0 \\ 0.0 & 0.9934 & -1.341 & -0.169 & -0.2518 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & -20.0 & 0.0 & 20.0 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.0 & -20.0 & 0.0 & 20.0 \\ 2110.0 & 890.6 & 4910.7 & -534.3 & -100.8 & -1000.0 & 0.0 \\ -5320.0 & -898.0 & -4661.8 & 428.0 & 109.9 & 0.0 & -1000.0 \end{bmatrix}$$

$$B = \begin{bmatrix} 0.0 & 0.0 \\ 0.0 & 0.0 \\ 0.0 & 0.0 \\ 0.0 & 0.0 \\ 0.0 & 0.0 \\ 1000.0 & 0.0 \\ 0.0 & 1000.0 \end{bmatrix} \tag{A1}$$

The seven state variables are pitch angle, pitch rate, angle of attack, elevator angle, flaperon angle, elevator actuator input, and flaperon actuator input. The two system inputs are elevator angle command and flaperon angle command. The first five rows of Eq. (A1) represent the open-loop model of the aircraft. The last two rows form the combined inputs to the actuators due to pilot commands and feedback that is

required to stabilize this open-loop unstable aircraft. This model has been derived from the fifth-order model (design no. 1) of Ref. 13 by including all of the parameters of the actuators in the system matrix using the method described earlier in this paper. The elevator and flaperon actuator dynamics, both described by $\dot{x} = -20x + 20u$, appear in the fourth and the fifth rows of Eq. (A1), respectively.

References

[1]Gertler, J. J., "Survey of Model-Based Failure Detection and Isolation in Complex Plants," *IEEE Control Systems Magazine*, Vol. 8, No. 6, 1988, pp. 3–11.

[2]Willsky, A. S., "A Survey of Design Methods for Failure Detection in Dynamic Systems," *Automatica*, Vol. 12, Nov. 1976, pp. 601–611.

[3]Horak, D. T., "Failure Detection in Dynamic Systems with Modeling Errors," *Journal of Guidance, Control, and Dynamics*, Vol. 11, No. 6, 1988, pp. 508–516.

[4]Isermann, R., "Process Fault Detection Based on Modelling and Estimation Methods," *Automatica*, Vol. 20, July, 1984, pp. 387–404.

[5]Jones, H. L., "Failure Detection in Linear Systems," Ph.D. Dissertation, Dept. of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA, 1974.

[6]Massoumnia, M.-A., "A Geometric Approach to the Synthesis of Failure Detection Filters," *IEEE Transactions on Automatic Control*, Vol. AC-31, No. 9, 1986, pp. 839–846.

[7]Massoumnia, M.-A., Verghese, G. C., and Willsky, A. S., "Failure Detection and Identification," *IEEE Transactions on Automatic Control*, Vol. 34, No. 3, 1989, pp. 316–321.

[8]Clark, R. N., "Detecting Instrument Malfunctions in Control Systems," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. AES-11, No. 4, 1975, pp. 465–473.

[9]Massoumnia M.-A., and Vander Velde, W. E., "Generating Parity Relations for Detecting and Identifying Control System Component Failures," *Journal of Guidance, Navigation, and Control*, Vol. 11, No. 1, 1988, pp. 60–65.

[10]Horak, D. T., and Allison, B. H., "Isolation of Parametric Failures in Dynamic Systems," *Proceedings of the 1989 American Control Conference*, American Automatic Control Council, Green Valley, AZ, June 1989, pp. 868–873.

[11]Horak, D. T., and Allison, B. H., "Experimental Implementation and Evaluation of the RMI Failure Detection Algorithm," *Proceedings of the 1987 American Control Conference*, American Automatic Control Council, Green Valley, AZ, June 1987, pp. 1803–1810.

[12]Horak, D. T., "Experimental Estimation of Modeling Errors in Dynamic Systems," *Journal of Guidance, Control, and Dynamics*, Vol. 12, No. 5, 1989, pp. 653–658.

[13]Sobel, K. M., and Shapiro, E. Y., "A Design Methodology for Pitch Pointing Flight Control Systems," *Journal of Guidance, Control, and Dynamics*, Vol. 8, No. 2, 1985, pp. 181–187.

[14]Horak, D. T., "Isolation of Unstructured System Failures in Dynamic Systems," *Proceedings of the AIAA Guidance, Navigation and Control Conference*, AIAA, Washington, DC, 1989, pp. 728–732.

[15]Merrill, W. C., "HYTESS—A Hypothetical Turbofan Engine Simplified Simulation," NASA TM-83561, Jan. 1984.