Published at 2010 AIAA Infotech @ Aerospace Conference, Atlanta, GA, April 2010, AIAA-2010-3307

Detecting Aircraft Performance Anomalies from Cruise Flight Data

Eric Chu* and Dimitry Gorinevsky†

Stanford University, Stanford, CA 94305; Mitek Analytics LLC, Palo Alto, CA 94306

Stephen P. Boyd‡

Stanford University, Stanford, CA 94305

We propose an approach to detecting anomalies from aircraft cruise flight data. The detection is based on a model learned from the historical data of a fleet of aircraft. For a variety of cruise flight conditions with and without turbulence, we validate the approach using a FOQA dataset generated by a NASA flight simulator. We identify a regression model that maps the flight conditions and aircraft control inputs into accelerations (linear and rotational). Anomalies are detected as outliers that exceed the scatter caused by turbulence and the modeling error. The detection method is related to multivariable statistical process control. Sensor offset anomalies that are a fraction of a degree in flight surface position and a small percentage of the experienced sensor range are reliably detected from the data collected under light turbulence conditions.

Nomenclature

x	Input data vector
y	Output data vector
ϕ	Regressor applied to input data vector
A	Learned model
W	Residual covariance
i	Flight number index
t	Time index
k	Fault index
$\{x_i\}_{i=1}^N$	The set of elements $x_1, \ldots, x_N$

I. Introduction

I.A. Goals

In fleets of aircraft used by airlines and other operators, flight performance is monitored to ensure optimal operation and also to detect anomalies. Small offsets in aircraft flight surfaces or biases in sensors may translate into increased fuel consumption or indicate a safety issue. It is desirable to detect these anomalies as early as possible and fix these problems.

In this paper, we consider a fleet of aircraft (vehicles) of the same type. Flight performance data for each vehicle is collected during the flight and stored in a ground database. Airlines usually collect such data as part of a FOQA (Flight Operations Quality Assurance) program. Using historical fleet data, we recover an average, data-driven model of the nominal operation for vehicles in the fleet. Anomalies can be detected as deviations from the model.

*Graduate Student, Stanford University, echu508@stanford.edu; Research Engineer, Mitek Analytics LLC, AIAA Student Member

†Managing Partner, Mitek Analytics LLC, dimitry@mitekan.com; Consulting Professor, Stanford University, AIAA Member

‡Professor, Department of Electrical Engineering, Stanford University, boyd@stanford.edu

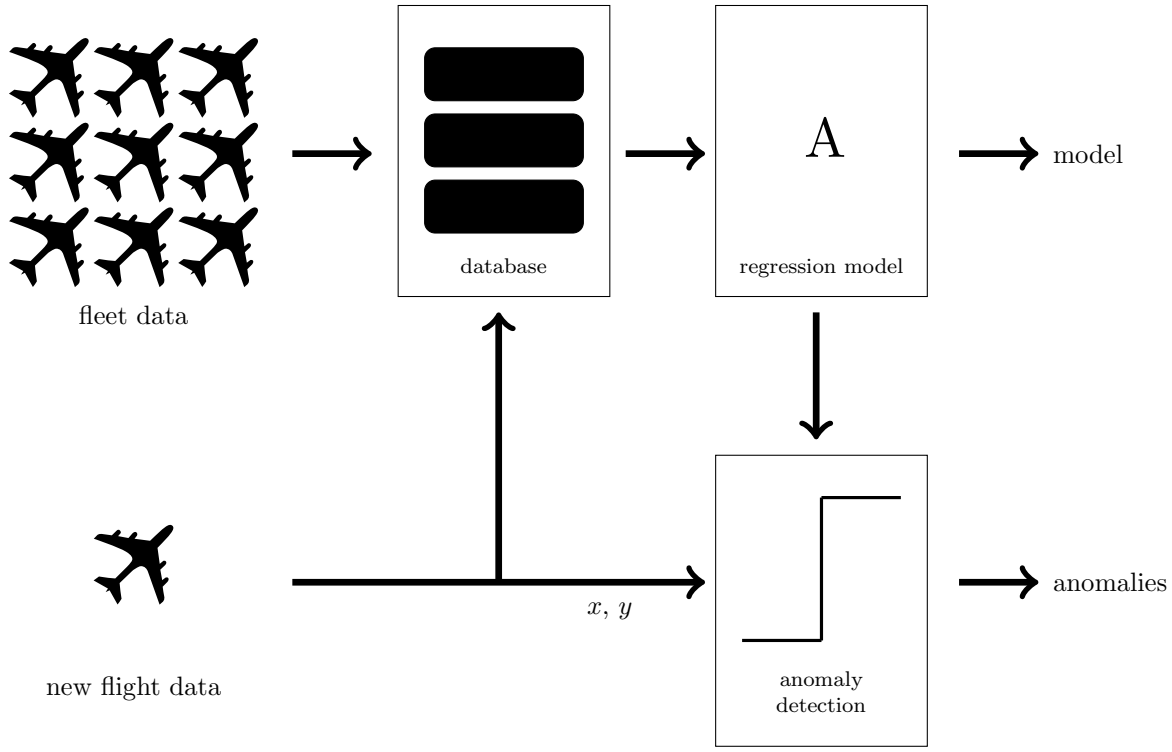

Figure 1: Flow chart demonstrating the flow of information in our assumptions.

Figure 1 shows the overall flow of information in the proposed data processing system. The aircraft fleet data are stored in a database. We mine the data to produce a model, A. The model can then be used on new flight data records to detect anomalies in flight performance. This paper describes a method for building a model from a large dataset; we also discuss how to use this model for anomaly detection.We use the term "anomaly detection" to refer to the act of determining whether or not a fault exists; we say that there is an anomaly if a fault exists.

For aircraft data, we make some additional assumptions. First, we assume that the data was collected in a range of cruise flight conditions. That is, the aircraft might fly at various mach numbers, altitudes, trims, and turbulence conditions. Secondly, we select a model structure that is informed by the structure of aircraft flight dynamics equations. Finally, we justify using a data-driven approach by noting that aircraft operators often do not have access to performance models. Our approach is roughly similar to that used to build aerodynamical models of aircraft in a flight test program; however, we use a less detailed model structure and cover a smaller portion of the flight envelope (*i.e.*, only cruise flight). Section III.A describes these assumptions in more detail.

To handle large scale fleet data, we need a method that is scalable and a model that is quickly computable. The requirements for our method are being able to

- process large amounts (terabytes) of vehicle data,
- work with a fleet of vehicles of the same kind (*i.e.*, a fleet of 757's, *etc.*),
- use known structure of the dynamical models (input and output variables), and
- assume that aircraft operate in a well-defined range of conditions (*e.g.*, cruise flight).

I.B. Previous Work

The proposed approach is related to multivariate statistical process control (SPC)[1] methods. SPC is typically used in manufacturing and process industries for systems operating at a fixed setpoint. The main multivariable SPC methods are partial least squares (PLS)[2] and principal component analysis (PCA).[1] Neither of these methods have been previously applied to data from aircraft operating at a range of flight conditions

(a range of setpoints). The regression approach we use can be considered as a generalization of full least squares, which, in turn, is a special case of PLS.

We use a data-driven least squares (kernel) regression method that can detect faults in a fleet of aircraft using data stored in a large, historical database. To the authors' best knowledge, there is no prior work where the dynamics of an aircraft are identified from a (scalable) linear regression on historical aircraft fleet data and then used for anomaly detection and fault diagnostics.

Related work by Dimogianopoulos, *et al.* uses a nonlinear autoregressive model with exogenous inputs to model flight dynamics from incomplete sensor data and perform anomaly detection via statistical tests.[3] The model is constructed offline using ordinary (or weighted) least-squares, and fault detection is performed online. The paper does not discuss scaling the algorithm for use on large, historical flight data sets.

There is some related work that uses data to build models and use them to detect anomalies. These include NASA's inductive monitoring system (IMS)[4] and the multivariable state estimation technique (MSET)[5] originally developed at Argonne National Labs. There is not much detail surrounding these systems because of their proprietary nature. These monitoring systems do not focus on fleets of devices and little is known about whether or not they scale to large datasets. Nonetheless, these systems have been deployed in several process monitoring applications (*e.g.*, monitoring nuclear power plants[6]). Some other model-based work use neural networks to identify a model and diagnose faults in engine data.[7] MSET has also been licensed to the company SmartSignals where it was used to detect the onset of engine failures in aircraft.[8]

A scalable approach that uses aviation safety data for evaluating the performance of aircraft operators is described in the work by Cheng, *et al.*[9] While able to handle large quantities of safety data, their work mostly focuses on operational safety.

Related work focused on monitoring a fleet of devices, in particular, buses, have been pursued at Volvo. They construct self-organizing maps as compact representations of nominal vehicle operation and pass these to other buses in the fleet.[10] The on-board computers in the buses can then determine possible faults. While this work deals with fleet monitoring, technically, it is very different from ours and focused on a different application.

Aircraft flight dynamics models are built in work on adaptive, reconfigurable flight control, such as NASA's work on integrated resilient aircraft control (IRAC).[11] Though is it focused on a related application problem, this work is very different from ours in the goal and scope. They perform in-flight fault diagnostics from data collected during a single aircraft. The diagnostic results warn pilots and adjust the flight controller in order to ensure safe flight.

I.C. Contribution and Outline

The contributions of this work are twofold:

1. We propose an approach that allows accurate detection of aircraft performance anomalies in cruise flight data. Our method is able to detect flight surface offsets on the order of 0.1 degrees and sensor offsets that are at least 5% of the range of values experienced during cruise flight. This has been demonstrated using a NASA flight simulator.

2. Our method is completely data-driven. No prior knowledge of the aircraft model is used except knowledge of the inputs and outputs of the dynamics model. The flight dynamics model is determined empirically and identified from a historical dataset. The method we present can handle large scale datasets that do not fit in memory (on the order of terabytes).

The remainder of this paper is divided in to two main sections.

In Section II, we discuss how to formulate the problem in a general setting. We also describe a method to efficiently compute the model from our large dataset. And in Section II.E, we discuss how to use the model to detect anomalies. We present two different statistics that can be used for anomaly detection.

In Section III, we discuss how we set up our simulation and the data we collected. We then present the results of model identification and anomaly detection. Our conclusion states that despite the existence of more sophisticated neural network-based methods, clustering methods, or self-organizing maps, an affine least-squares model will detect faults sufficiently well.

II. Problem Formulation and Setup

II.A. Model Structure

From flight dynamics, we know that the forces acting on an aircraft and, consequently, the experienced accelerations are a function of the aircraft control surface positions, velocities and rates, dynamic pressure, thrust, and some other flight condition variables. So, we can write loosely

$$y = F(x) + e, \tag{1}$$

where y is a vector of six accelerations (three linear accelerations at the aircraft center-of-gravity and three rotational accelerations), x is a vector of inputs, and e is some noise that is a consequence of unaccounted flight condition variables and turbulence. The nonlinear map $F(x)$ describes the dependence of the accelerations on vehicle mass, rotational inertia, forces, and moments acting on the vehicle. The components of the input vector x include s, v, $\bar{q}$, and τ, where s is a vector of flight surface angles, v is a vector of velocities and rotational rates, $\bar{q}$ is dynamic pressure, and τ is thrust. The components of the input x and accelerations y are available in the aircraft avionics systems and are collected as part of FOQA datasets. The goal is to (approximately) solve for the function F and use this model to detect subsequent faults.

II.B. Problem Statement

The problem is divided into two parts: the first is to identify the model F; the second is to use the identified model to detect and diagnose faults. These correspond to the two major blocks in Figure 1.

First, however, we describe our database of flights more formally. For a single flight, we might have a number of sensors (airspeed sensors, flight surface sensors, *etc.*) and their recorded values during flight. For example, an airspeed sensor for a single flight would contain the speed of the aircraft during the entire flight. The dataset collected during the flight can be represented as

$$\{x_t, y_t\}_{t=1}^{M}, \tag{2}$$

where M is the number of samples for the flight, x_t and y_t are the input and output vectors at time sample t, respectively. The input-output pairs are independent and obey (1).

The database consists of a collection of these flights. That is, it is the set of sets

$$\{\, \{x_{i,t}, y_{i,t}\}_{t=1}^{M_i} \,\}_{i=1}^{N}, \tag{3}$$

where i indexes the flight in the database. We do not distinguish between the aircraft in the fleet (they are assumed to be the same). Note that M_i suggests that the amount of data collected per flight may vary from flight to flight.

Now, the first problem is to identify the map F from the data in our database. We approach this as a linear regression problem and replace (1) with

$$y = A\phi(x) + \epsilon, \tag{4}$$

where $\phi(x)$ is a regressor, A is a matrix that contains the model parameters, and ϵ combines the noise, e, in (1) and the error of the regression fit. In what follows, we denote the regressor by ϕ and drop the dependency on the input x to simplify notation. Section II.C discusses the choice of regressors.

Our first problem of identifying F then becomes a problem of identifying A from a large amount of data given in the set (3). While aircraft manufacturers may have access to aircraft flight models, these models tend to be complex and expensive to construct. Model identification gives us an empirical model while circumventing these costs. These (simple) empirical models can then be used for anomaly detection. We describe how to identify the model from our large dataset in Section II.D. Figure 2 shows a more detailed flow chart for model identification.

The second problem is to use this model A to detect anomalies. For that, we consider a modification of (4) and write it as

$$y = A\phi(x) + \epsilon + f, \tag{5}$$

where f represents the impact of the fault condition on the output y. Assuming that x, y, and A are known in (5), we want to determine if f is nonzero. We call this problem the anomaly detection problem: we

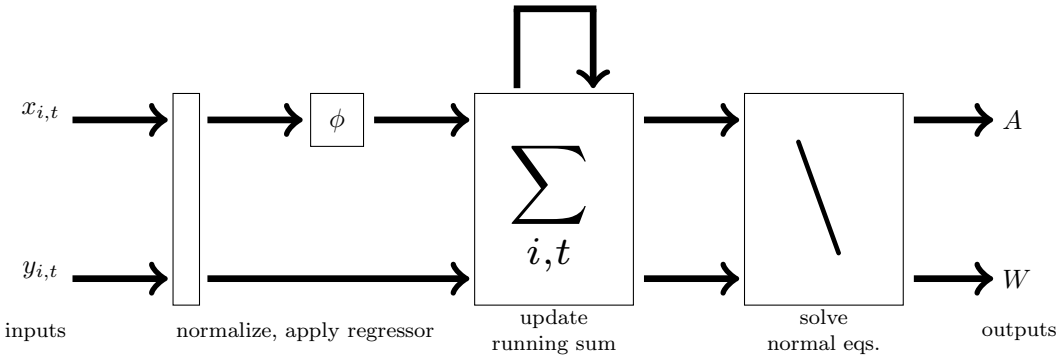

Figure 2: Detailed flow chart showing how to build our least-squares model.

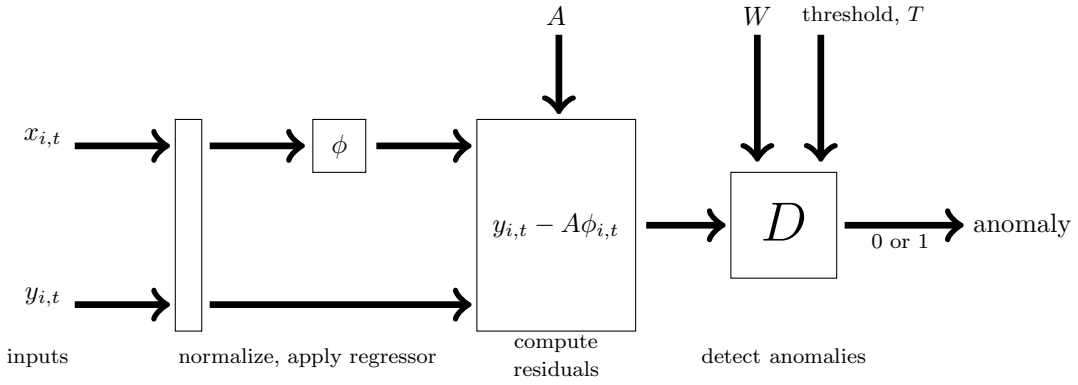

Figure 3: Detailed flow chart showing how to perform anomaly detection.

want to identify when there exists a fault in an entire flight record given by (2). We describe how to detect anomalies for an entire flight in Section II.E. Figure 3 shows a flow chart for anomaly detection.

Succinctly, the problem can be summarized: Find a model that holds across a range of operational parameters, environmental conditions, and the entire fleet of vehicles; use this model to detect anomalies in vehicle data.

II.C. Choice of Regressors

We can use different functions ϕ to map the inputs to regressors; a simple one might be the affine (linear) regressor

$$\phi_{\text{affine}}(x) = \begin{bmatrix} x \\ 1 \end{bmatrix}. \tag{6}$$

Using this regressor, the derived model A would describe an affine (linear) relationship between the input x and the output y.

An alternative regressor to use would be a quadratic regressor. Instead of the affine regressor (6), we can use the quadratic regressor that is constructed in the following manner

$$\phi_{\text{quadratic}}(x) = \begin{bmatrix} \{x_j x_k\}_{j \leq k} \\ x \\ 1 \end{bmatrix}, \tag{7}$$

where the set $\{x_j x_k\}_{j \leq k}$ is the set of all distinct products of the elements of the vector x. Here, the subscript refers to the elements of x as opposed to a distinct flight or time. We may also use a subset of all quadratic terms. The derived model would describe a quadratic relationship between the input x and output y.

Some other regressors might be a cubic regressor or a sinusoidal regressor. The choice of regressors affects the interpretation and performance of the regression model, but not how the model is constructed. Choosing a high-dimensional regressor requires that we properly regularize our model identification problem.

What we have presented in (4), (5), (6), and (7) are special cases of what are better known as kernel methods.[12] The MSET algorithm is equivalent to a specific kernel method as well.[13]

A full dynamic model of the aircraft used for control and performance analysis can be also represented in the form (4) where the model is given by the aerodynamic database coefficients (such a lift and drag coefficients in different aircraft), inverse mass matrix parameters, and so on. The regressor coefficients include dynamic pressure, B-splines used for interpolation between the aerodynamic database nodes, and geometric parameters defined by sines and cosines of various kinematic angles.

II.D. Model Identification with Large Amounts of Data

In this section, we describe how to obtain a model A from a large database. Using data without faults (such that $f = 0$) and without heavy turbulence (so that ϵ is small), we can construct a model from the data. The solution can be constructed by finding the (regularized) least squares approximate solution for A by solving

$$\text{minimize} \quad \sum_{i=1}^{N} \sum_{t=1}^{M_i} \|y_{i,t} - A\phi_{i,t}\|^2 + \lambda \|A\|_F^2, \tag{8}$$

where $\phi_{i,t}$ and $y_{i,t}$ represent the regressor and output vector, respectively, at time t from the i^{th} vehicle, and $\|\cdot\|_F$ denotes the Frobenius norm of a matrix. The number N represents the number of flights, and M_i is the number of time samples for the i^{th} flight. The parameter λ is chosen to keep the coefficients in the model "small" and prevent model overfit. If we use an affine regressor for the model, we call the least-squares model obtained this way the affine model. A different regressor would be similarly described (*e.g.*, a quadratic model).

The solution to (8) can be computed quickly and in a distributed fashion for extremely large data sets that do not fit into memory and are stored in large historical databases. By taking the derivative with respect to the matrix A in (8) and setting the result equal to 0, we obtain the normal equations that provide the solution to (8),

$$\left(\lambda I + \sum_{i=1}^{N} \sum_{t=1}^{M_i} \left(\phi_{i,t} \phi_{i,t}^T \right) \right) A^T = \sum_{i=1}^{N} \sum_{t=1}^{M_i} \phi_{i,t} y_{i,t}^T. \tag{9}$$

Note that (9) includes running sums of the data over all vehicles and all samples. We can easily handle large datasets and also construct our model recursively by updating the two matrices containing these running sums,

$$\sum_{i=1}^{N}\sum_{t=1}^{M_i}\left(\phi_{i,t}\phi_{i,t}^T\right) \quad \text{and} \quad \sum_{i=1}^{N}\sum_{t=1}^{M_i}\phi_{i,t}y_{i,t}^T.$$

This update corresponds to the update block of our model identification flow chart in Figure 2.

To see how this can be used for a large scale database (one that does not fit in computer memory), suppose that we have obtained M_i data points each from N flights, and we want to compute the updated model when we obtain flight data for the $N+1$ flight. We now have M_{N+1} more data points. Then, we perform the two updates

$$\begin{aligned}\sum_{i=1}^{N+1}\sum_{t=1}^{M_i}\left(\phi_{i,t}\phi_{i,t}^T\right) &= \sum_{t=1}^{M_{N+1}}\left(\phi_{(N+1),t}\,\phi_{(N+1),t}^T\right)+\sum_{i=1}^{N}\sum_{t=1}^{M_i}\left(\phi_{i,t}\phi_{i,t}^T\right)\\ \sum_{i=1}^{N+1}\sum_{t=1}^{M_i}\phi_{i,t}y_{i,t}^T &= \sum_{t=1}^{M_{N+1}}\phi_{(N+1),t}\,y_{(N+1),t}^T+\sum_{i=1}^{N}\sum_{t=1}^{M_i}\phi_{i,t}y_{i,t}^T\,,\end{aligned} \tag{10}$$

where the double sums on the right-hand side have already been computed at the previous update. Updating the model only requires that we compute the outer product of the new M_{N+1} regressors with themselves and the outputs. At each step we sum data for the new flight only; then, we add the sum to the existing running sum matrix. To prevent numbers from overflowing, we can always divide both sides by the total number of data entires.

Note that the size of the matrices in (10) depends only on the number of regressors and output variables; it does not depend on the size of the data. This data-independence is what allows us to scale our model identification method up to infinite data. To give a sense of the scale we can handle, the data in Section III contains 2660 flights with 600 time samples each. That is, we have $N = 2660$ flights and $M_i = 600$ for all flights; there are over 1 million input data points in $\mathbf{R}^{19}$ and output data points in $\mathbf{R}^6$. If we assume each number is stored as a 64-bit floating point number in memory, just storing the inputs alone would occupy approximately 200Mb of memory, well outside the range of modern-day processor caches. Using (6), our regressors are vectors in $\mathbf{R}^{20}$; the running sum matrices are therefore matrices in $\mathbf{R}^{20\times 20}$ and $\mathbf{R}^{20\times 6}$. Instead of requiring 200Mb of memory to build our model, we would only need 4kb of memory on the L1 cache. Now, if we were to obtain data for the 2661^{th} flight, we would compute the outer product of 600 vectors from the new flight and add it to our matrices—adding new flight data does not cause us to occupy more memory. We can then solve for A by adding the regularization term λI and by using any least-squares package—all while occupying only 4kb of memory.

Finally, we also compute the empirical covariance for the residuals as a combination of the running sums

$$\begin{aligned}W &= \frac{1}{K-1}\sum_{i,t}(y_{i,t}-A\phi_{i,t})(y_{i,t}-A\phi_{i,t})^T\\ &= \frac{1}{K-1}\left(\sum_{i,t}(y_{i,t}y_{i,t}^T)-2A\sum_{i,t}(\phi_{i,t}y_{i,t}^T)+A\sum_{i,t}(\phi_{i,t}\phi_{i,t}^T)A^T\right),\end{aligned} \tag{11}$$

where A is our learned model, and $K=\sum_{i=1}^{N}M_i$ is the total number of data points. Note that we have already computed the matrices, $\sum(\phi_{i,t}y_{i,t}^T)$ and $\sum(\phi_{i,t}\phi_{i,t}^T)$. We can compute the matrix $\sum(y_{i,t}y_{i,t}^T)$ in a similar manner, as in (10).

II.E. Anomaly Detection

To detect anomalies, we use the identified model and compute the residual

$$r_{i,t} = y_{i,t} - A\phi_{i,t}. \tag{12}$$

The motivation for using this method comes from statistical process control methods that are used to detect (persistent) changes in the mean, such as cumulative sums (CUSUM).

Given the residuals (12), we can

1. threshold the norm of the residual at a given time for a given flight, or

2. compute the mean residual over an entire flight and then threshold its norm.

These two methods correspond to different methods of anomaly detection. The first method is similar to computing the Hotelling T^2 statistic for all time and for all flights. The second method is a Hotelling T^2 statistic for an average residual over one flight. These methods can be thought of as "drop-in" versions of the detector in Figure 3.

The first detector labels every sample as anomalous or not by comparing the Mahalanobis norm of the residual to a threshold. We use the covariance, W in (11), to compute the Mahalanobis norm. It can be summarized as a detector D_0, which is a function of the individual residuals, $r_{i,t}$. The detector D_0 is defined as

$$D_0(r_{i,t}) = \begin{cases} 1 & \text{if } \|W^{-1/2} r_{i,t}\|^2 \geq T_0 \\ 0 & \text{otherwise,} \end{cases} \tag{13}$$

where a 1 corresponds to an anomalous sample; T_0 is the chosen threshold; and W is the covariance in (11).

We call this detector the "single sample" anomaly detector. We cannot use D_0 directly to decide whether aircraft dynamics are anomalous because D_0 works for a single sample and each flight record has many samples. Instead, we want to detect whether an anomaly occurred during an entire flight record.

The second method computes Hotelling T^2 statistics for the average residual across an entire flight, $\frac{1}{M_i}\sum_{t=1}^{M_i} r_{i,t}$. This number is computed for each flight i where M_i is the number of samples for that flight. We compute its Mahalanobis norm, assuming that the residuals are independent, identically distributed (IID). Since anomalies are thought to cause a shift in the mean, anomalous data would have a mean that is very different from zero. By averaging across the entire flight, we can reduce the effects of noise (caused by turbulence). This method produces a single indicator of whether or not a flight is anomalous. We summarize the second method by

$$D_1(r_i) = \begin{cases} 1 & \text{if } \frac{1}{M_i^2}\|W_{\text{ave}}^{-1/2}\sum_{t=1}^{M_i} r_{i,t}\|^2 \geq T_1 \\ 0 & \text{otherwise.} \end{cases} \tag{14}$$

Assuming IID residuals, $W_{\text{ave},i} = W/M_i$ where W is given by (11). We call this detector the "single flight" anomaly detector.

III. Experimental Results

III.A. Simulation Setup

We verified our approach in a simulation experiment. We have generated about 1Gb of simulated FOQA data using NASA's flight simulator, FLTz.[15] FLTz is a medium fidelity flight simulator widely used by NASA and its collaborators. It has been used to develop adaptive reconfigurable flight control, planning emergency maneuvers, and in-flight fault detection for fixed-wing aircraft. FLTz is capable of simulating various types of transport aircraft. For our simulation, we used the advanced concepts flight simulator (ACFS) aircraft—it is a generic transport model, roughly equivalent to the L-1011. FLTz can also simulate flights with several levels of turbulence severity; we alternate between the lightest turbulence and no turbulence in our experiments.

For our purposes, we configured FLTz as follows. We used a traditional controller for our simulated aircraft. We also only simulated the cruise flight regime. For each flight, we had an autopilot fly the plane in cruise for 10 minutes. We also modified the FLTz source code to add functionality. We added sensor offset faults to FLTz and used them in our simulations. These faults occur when the sensor produces a value that is statically offset from the actual value (biased). We modeled actuator faults as flight surface offsets introduced in to the actuator loop.

We simulated flights across a wide range of flight parameters. The altitude of flight was randomly chosen between $30,000$ and $40,000$ feet, the Mach number from between 0.78 and 0.84, and the turbulence levels from none to light turbulence, as previously mentioned. Using the same spread of parameters, we were also able to simulate varying fuel load and payload by varying the weight of our plane from between $150,000$ lbs and $185,000$ lbs. In our simulations, we also adjusted the moment of inertia proportionally to the mass. We did not explicitly simulate fuel burn; instead, because our data is collected over a short time interval of flight, we assume that the mass is constant during this interval. However, between different cruise flight

segments, the plane might have a different mass depending on the amount of fuel consumption up to that point. Section III.D considers data that has varying mass. Sections III.B to III.C only consider data from aircraft with constant mass.

Using the simulator, we generated 2660, non-faulted flights with varying Mach numbers, altitudes, and turbulence conditions to use as our training data. We use a sampling rate of 1Hz to simulate the collection of FOQA data and collect data for 10 minutes of cruise flight. Therefore, we can think of the dataset as having 2660 flights and 600 samples per flight—we have $N = 2660$ and $M_i = M = 600$ for all 2660 flights.

For each of the 2660 flights and 600 samples, we have an input $x_{i,t} \in \mathbf{R}^{19}$ and an output $y_{i,t} \in \mathbf{R}^6$. There are 19 sensed variables in the $x_{i,t}$'s and 6 output variables in the $y_{i,t}$'s as listed in the appendix.

To avoid numerical problems and neutralize the effect of large variables, we normalized our data and apply the regressor ϕ to the normalized data. We compute

$$\phi_{i,t} = \phi\left(2\left(\frac{x_{i,t} - x_{\min}}{x_{\max} - x_{\min}}\right) - 1\right), \tag{15}$$

where $x_{\min}$ and $x_{\max}$ are the maximums and minimums set in the appendix. This normalization forces most of our raw data to lie in $[-1, 1]$ and prevents large numbers (such as forward velocity) from dominating. We also normalize the outputs $y_{i,t}$ in the same way (but do not apply the regressor, ϕ).

We concatenate the resulting $\phi_{i,t}$'s to form the matrix Φ, and we concatenate $y_{i,t}$'s to form Y; that is, we form

$$\Phi = \begin{bmatrix} \phi_{1,1} & \phi_{1,2} & \cdots & \phi_{N,M} \end{bmatrix}$$
$$Y = \begin{bmatrix} y_{1,1} & y_{1,2} & \cdots & y_{N,M} \end{bmatrix}.$$

Note that $N \times M = 1,596,000$. These 2660 nonfaulted flights produce the training data, Φ and Y which we call Φ_{train} and Y_{train} to differentiate from our test data.

In addition to the described training set, we have generated a test set of 100 nonfaulted flights, Φ_{nofault} and Y_{nofault}, and 100 faulted flights, Φ_{fault} and Y_{fault}, with identical flight conditions. Each flight also records 600 data points over 10 minutes of flight (a 1Hz sampling rate). So for the test set, $N = 100$ and $M_i = M = 600$ for all 100 flights.

Some faults are flight surface actuator offset faults, where the actual position of the flight surface is offset (by a constant) from the actuator command. We also have sensor offset faults, where the sensors measuring airspeed, pitch, and so on may be offset. In total, we simulated eight different types of faults for our faulted data. These are

1. left aileron sensor offset, labeled AIL
2. left elevator sensor offset, labeled ELEV
3. rudder sensor offset, labeled RUD
4. stabilizer sensor offset, labeled STAB
5. measured roll acceleration offset, labeled ROLLA
6. measured vertical acceleration offset, labeled VERTA
7. measured dynamic pressure offset, labeled DYNP
8. measured pitch angle offset, labeled PITCH.

In Table 1, we list the seeded fault magnitudes for each of the offset faults. These magnitudes were chosen so that the affine model produces a receiver operating curve (ROC) that is close to the 5% false positive rate and 95% true positive rate (see Section III.C for an explanation of ROC curves).

	Fault Magnitude	Range Experienced in Simulation
AIL	0.3°	−5° to 5°
ELEV	0.1°	−2° to 2°
RUD	0.4°	−5° to 5°
STAB	0.02°	−2° to −1°
ROLLA	0.05 rad/s^2	−0.25 to 0.25 rad/s^2
VERTA	0.15 m/s^2	−3 to 3 m/s^2
DYNP	5 Pa	200 to 300 Pa
PITCH	0.17°	0° to 3°

Table 1: The seeded fault magnitudes and their experienced ranges in simulation.

III.B. Goodness of Model Fit for Fixed Mass Aircraft

In Section II.B, we wanted to obtain a model (of our choice) that could replace the mapping F given in Figure 2. To measure how good our model is, we want to know how much of the output variation the model explains. We compute

$$p = 1 - \frac{\sum_{i,t} \|y_{i,t} - A\phi_{i,t}\|^2}{\sum_{i,t} \|y_{i,t}\|^2} \tag{16}$$

and refer to p as the "predictive power" of the model or the "variance explained" by our model. If $p = 1$, then the numerator must have been 0. This only occurs if $A\phi_{i,t}$ is a perfect predictor of $y_{i,t}$. If $p = 0$, then we know that $A\phi_{i,t}$ explains none of the variance in $y_{i,t}$.

For fixed mass aircraft and the flight parameters described in Section III.A, we use the data in Φ_{train} and Y_{train} to solve (8) and obtain a model A. Then, using the data in Φ_{nofault} and Y_{nofault}, we computed p for two kinds of model—an affine model and a quadratic model. For both models, we use no regularization ($\lambda = 0$). Cross-validation showed that $\lambda = 0$ gave the best fit results over the regularization path for both these models.

Table 2 shows the results. Note that the quadratic model leaves about 15% less variation unexplained.

	Affine Regressor	Quadratic Regressor
Predictive Power	0.817	0.842

Table 2: The predictive power of the affine model and the quadratic model.

III.C. Anomaly Detection for Fixed Mass Aircraft

Using the anomaly detection methods described in Section II.E and in Figure 3, we test the anomaly detection abilities of the linear and quadratic models obtained in Section III.B for fixed mass aircraft.

The model A and the covariances W and W_{ave} were computed from our training data, contained in the matrices Φ_{train} and Y_{train}. By applying (8) to this data, we were able to compute the model and obtain empirical, residual covariances. The detector D_0 was not used in our experiments because it labels data samples as anomalous instead of entire flight records as anomalous. The threshold for D_1 (the single flight anomaly detector) was determined empirically from our results.

To compare the different regression models, we plot the receiver operating curve (ROC) in Figure 4. Each curve shows the dependence of the false-positive (no anomaly seeded, but anomaly detected) and true-positive (anomaly seeded, and anomaly detected) rates on the threshold T_1 in (14). The curves were empirically obtained using the test set data, Φ_{fault} and Y_{fault}. Different subplots in Figure 4 correspond to the detection of one of the eight faults listed in Table 1 that are seeded in our simulation experiment. We also show the 5% false-positive rate and 95% true-positive rate in our figure. Our goal is to have the ROC curve above the intersection of these two lines, achieving at worst a 5% false-positive rate and a 95% true-positive rate. The figures also show the magnitude of the sensor offset fault.

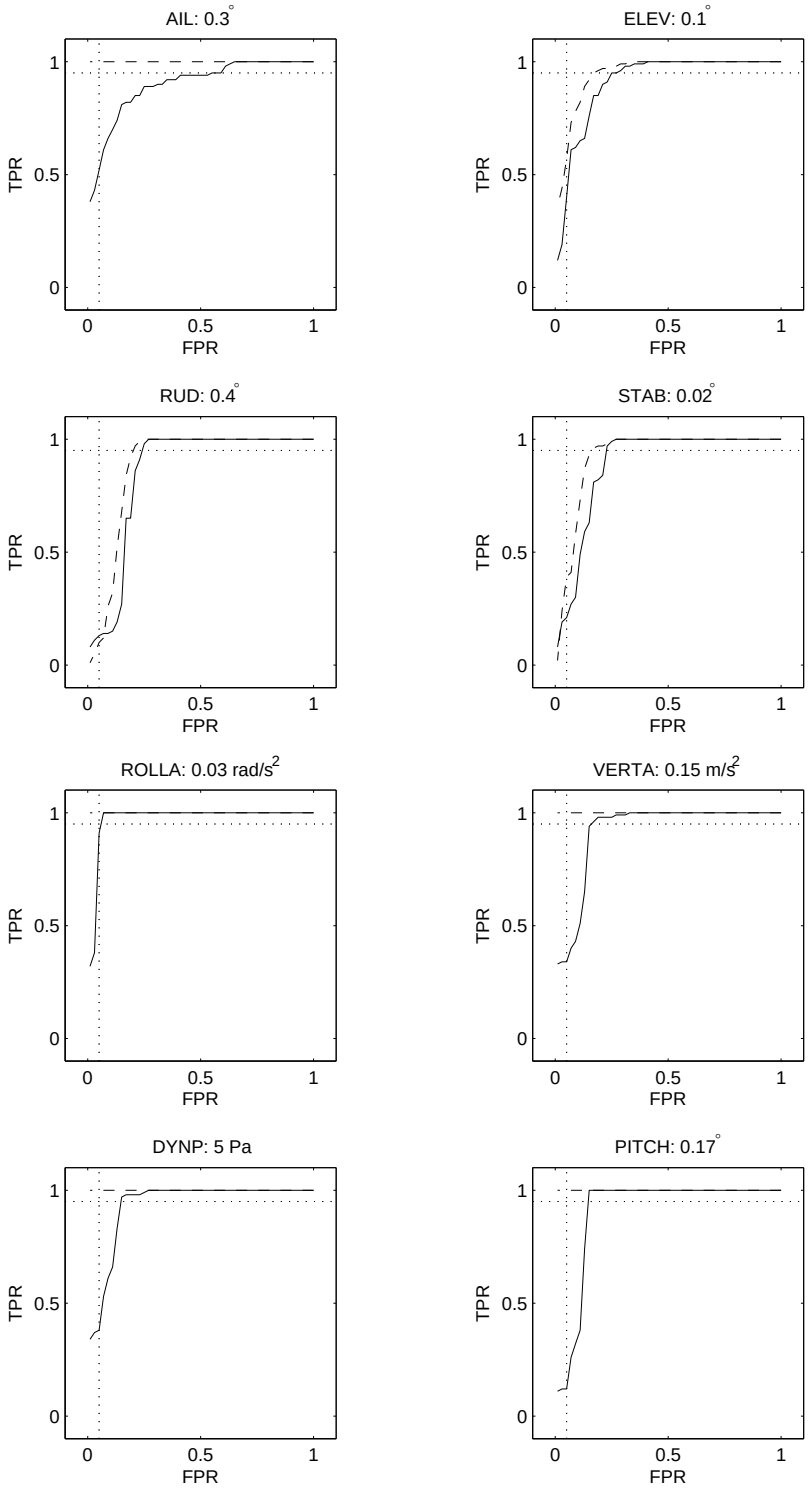

Figure 4: ROC curves for the affine model using D_1 (solid line) and the quadratic model using D_1 (dashed line). D_1 is the single flight anomaly detector.

We also show the area under the ROC curve in Table 3. It is clear that the affine model can perform just as well as the quadratic model. Since the affine model is the simplest model and performs just as well as a more complex model, the affine model is adequate for anomaly detection. We evaluate the performance of the affine model in more detail.

Area Under ROC Curve		
Detector	**Affine Regressor**	**Quadratic Regressor**
D_1	0.914	0.971

Table 3: The area under the ROC curve for the two models using the single flight anomaly detector, D_1, described in Section II.E.

III.D. Results with Varying Mass

While aircraft aerodynamics remain the same between flights and between different aircraft in the fleet, the aircraft mass varies with payload and fuel load. Therefore, we would like to be able to handle data induced by variable mass.

In this section, we assume that the aircraft mass is known. This assumption is true with certain accuracy for FOQA data, since the takeoff mass is known and the fuel burn is tracked during flight. To deal with varying mass, we note that our model roughly corresponds to a mapping between the inputs and the forces acting on the aircraft; that is, $y = A\phi(x)$ is roughly equivalent to

$$y = M^{-1}\tilde{A}\phi(x),$$

where M is the mass and inertial matrix for the aircraft and the learned model $A \approx M^{-1}\tilde{A}$.

Now, if the mass varies, we have approximately

$$(M_0 + M_1\Delta m)y = \tilde{A}\phi(x), \tag{17}$$

where M_0 is the average mass and rotational inertia of the aircraft across all time and all flights, and M_1 describes how the deviation from the average mass, Δm, affects the mass and rotational inertia of the aircraft currently.

We can build a model that takes the mass into account by manipulating (17) and obtaining

$$y = M_0^{-1}\tilde{A}\phi(x) - M_0^{-1}M_1\, y\Delta m. \tag{18}$$

Equation 18 contains the usual regressor $\phi(x)$ and also the added term $y\Delta m$. We can write (18) as

$$y = \begin{bmatrix} M_0^{-1}\tilde{A} \\ M_0^{-1}M_1 \end{bmatrix} \begin{bmatrix} \phi(x) \\ y\Delta m \end{bmatrix}.$$

Therefore, we define a new regressor,

$$\phi_{\Delta m}(x, y) = \begin{bmatrix} \phi_{\text{affine}}(x) \\ y\Delta m \end{bmatrix}$$

and use this regressor in (8) to solve for the model A. This regressor is an affine model with cross-terms.

Table 4 shows how well this regressor predicts the accelerations in the presence of varying mass as compared to a regular affine regressor. Note that this regressor achieves three times less unexplained variance than the regular affine regressor.

We compared how this regressor performs in fault detection. There appears to be no appreciable gain in performance when using the varying mass regressor over the affine regressor. This result suggests that a fixed mass affine model might have worse predictive performance, but may actually be suited for fault detection. One explanation might be related to the directionality of the residuals, where the computation for predictive power is dominated by one direction—given by the large singular vectors of the covariance—while fault detection ability is a function of the scatter along a different direction—given by the small singular vectors. This issue may be worth looking into in the future.

	Affine Regressor	**Varying Mass Regressor**
Predictive Power	0.787	0.929

Table 4: The predictive power of the affine model and the varying mass model.

III.E. Summary

In our simulation experiments, the affine model has roughly 80% predictive ability (see Section III.B). We found that though the quadratic model provides a better fit ($\approx 15\%$ better), its ability to detect faults is not drastically superior to the affine model. Therefore, the affine model can be recommended for practical use in fault detection.

To deal with varying aircraft mass, we added cross terms to the affine regressors. Such a regressor showed great improvement in the model fit. What is surprising, though, is that even though the affine model assumed a fixed aircraft mass and has an inferior fit to the data, when it comes to fault detection, it worked as well as the model with the mass cross-terms.

IV. Conclusions

We believe that the proposed data processing approach can be used for actual aircraft data because it is simple and efficient. The only information needed to run this algorithm is a large database consisting of input and output data collected during a number of past cruise flights. This information can be used to construct a flight dynamics model which can then be used to detect anomalies.

Some future extensions to this work might include a fault diagnostics algorithm that identifies the root cause of the anomaly; we would also like to have the model identification process be robust to faults that may be present in the training data; and lastly, we would like to be able to take individual vehicle or flight offsets in to account. These offsets might encode things such as slightly different center-of-gravities or individual variations in aircraft characteristics that lead to biases in the model.

Appendix: Data Normalization

These numbers were obtained by generating 100 flights with varying flight conditions—different altitudes, mach numbers, and turbulences—in the FLTz flight simulator. The turbulence levels vary from none to mild. The empirical maximum and minimum were computed and rounded before picking the normalization ranges.

- Parameters 1-19 are inputs; 20-25 are outputs. Parameter 26 is the mass used in our varying mass model.
- Comments give justification for range values. These range values were used for data normalization.

	Parameter	**Range**	**Units**	**Comments**
1.	Aileron	$[-0.5, 0.0]$	degrees	from mild turbulence data
2.	Differential Aileron	$[-5.0, 5.0]$	degrees	from mild turbulence data
3.	Elevator	$[-2.0, 2.0]$	degrees	from mild turbulence data
4.	Rudder	$[-5.0, 5.0]$	degrees	from mild turbulence data
5.	Stabilizer	$[-2.0, -1.0]$	degrees	from mild turbulence data
6.	Roll Angle	$[-0.1, 0.1]$	radians	about 5°; more in strong turb.
7.	Yaw Angle	$[-3.0, 3.0]$	radians	relative to earth-axis, between $[-\pi, \pi]$
8.	Pitch Angle	$[0.0, 0.05]$	radians	plane doesn't pitch much in cruise
9.	Angle-of-Attack	$[0.0, 0.05]$	radians	greater than 0; steady in cruise
10.	Sideslip Angle	$[-0.02, 0.02]$	radians	about 1°; more in strong turb.
11.	Mach Number	$[0.7, 0.9]$	—	less than 1; range from simulation
12.	Dynamic Pressure	$[200, 300]$	pascals	approx. from min, max mach
13.	Engine Thrust	$[20000, 33000]$	lbs	from mild turbulence data
14.	Long. Velocity	$[700, 900]$	ft / s	from mild turbulence data
15.	Lat. Velocity	$[-10, 10]$	ft / s	from mild turbulence data
16.	Vertical Velocity	$[0.0, 40]$	ft / s	from mild turbulence data
17.	Roll Rate	$[-0.1, 0.1]$	rad / s	from mild turbulence data
18.	Pitch Rate	$[-0.005, 0.005]$	rad / s	from mild turbulence data
19.	Yaw Rate	$[-0.01, 0.01]$	rad / s	from mild turbulence data
20.	Forward Accel	$[-0.5, 0.5]$	ft / s^2	from mild turbulence data
21.	Lateral Accel	$[-10, 10]$	ft / s^2	from mild turbulence data
22.	Vertical Accel	$[-10, 10]$	ft / s^2	from mild turbulence data
23.	Roll Accel	$[-0.25, 0.25]$	rad / s^2	from mild turbulence data
24.	Pitch Accel	$[-0.05, 0.05]$	rad / s^2	from mild turbulence data
25.	Yaw Accel	$[-0.05, 0.05]$	rad / s^2	from mild turbulence data
26.	Mass	$[4680, 5800]$	slugs	from simulation setup

Acknowledgments

This work was supported by NASA Grant NNX07AEIIA as a part of the IVHM Project in NASA Aviation Safety Program. The authors would like to acknowledge the help of Dr. Rodney Martin of NASA who brought the aircraft performance monitoring problem considered in the paper to their attention. E. Chu would also like to thank Scott Poll and Stefan Campell at NASA for assistance with FLTz.

References

[1]Qin, S. J., "Statistical Process Monitoring: Basics and Beyond," *Journal of Chemometrics*, Vol. 17, No. 8-9, 2003, pp. 480–502.

[2]Wold, S., "PLS-regression: a basic tool of chemometrics," *Chemometrics and Intelligent Laboratory Systems*, Vol. 58, No. 2, 2001, pp. 109–130.

[3]Dimogianopoulos, D. G., Hios, J. D., and Fassois, S. D., *Fault Detection and Isolation in Aircraft Systems Using Stochastic Nonlinear Modelling of Flight Data Dependencies*, IEEE, 2006.

[4]Iverson, D. L., "Inductive System Health Monitoring," The 2004 International Conference on Artificial Intelligence (IC-AI04), Las Vegas, 2004.

[5]Gross, K., Singer, R., Wegerich, S., Herzog, J., VanAlstine, R., and Bockhorst, F., "Application of a Model-based Fault Detection System to Nuclear Plant Signals," *International conference on intelligent systems applications to power systems*, Argonne National Lab, Seoul, Korea, 1997, p. 6.

[6]Zavaljevski, N. and Gross, K. C., "Sensor fault detection in nuclear power plants using multivariate state estimation technique and support vector machines," *3rd International Conference of the Yugoslav Nuclear Society*, Belgrade, 2000.

[7]Lu, P.-J., Zhang, M.-C., Hsu, T.-C., and Zhang, J., "An Evaluation of Engine Faults Diagnostics Using Artificial Neural Networks," *Journal of Engineering for Gas Turbines and Power*, Vol. 123, No. 2, April 2001, pp. 340–346.

[8]Herzog, J. P., Hanlin, J., Wegerich, S. W., and Wilks, A. D., "High Performance Condition Monitoring of Aircraft Engines," *ASME Turbo Expo 2005: Power for Land, Sea, and Air*, Reno-Tahoe, Nevada, 2005.

[9]Cheng, A. Y., Liu, R. Y., and Luxhoj, J. T., "Monitoring Multivariate Aviation Safety Data by Data Depth: Control Charts and Threshold Systems," *IIE Transactions*, Vol. 32, 2000, pp. 861–872.

[10]Svensson, M., Byttner, S., and Rognvaldsson, T., "Vehicle Diagnostics Method by Anomaly Detection and Fault Identification Software," *SAE Int. J. Passeng. Cars Electron. Electr. Syst.*, Vol. 2, No. 1, October 2009, pp. 352–358.

[11]Guo, T.-H. and Litt, J. S., "Resilient Propulsion Control Research for the NASA Integrated Resilient Aircraft Control (IRAC) Project ," 2007.

[12]Shawe-Taylor, J. and Cristianini, N., *Kernel Methods for Pattern Analysis*, Cambridge University Press, 2004.

[13]Gribok, A. V., Urmanov, A. M., and Hines, J. W., "Uncertainty Analysis of Memory Based Sensor Validation Techniques," *Real-Time Systems*, Vol. 27, No. 1, 2004.

[14]Hastie, T., Tibshirani, R., and Friedman, J. H., *The Elements of Statistical Learning*, Springer, 2003.

[15]Kaneshige, J., Bull, J., and Totah, J., "Generic Neural Flight Control and Autopilot System," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Vol. 4281, 2000.

[16]Duda, R. O., Hart, P. E., and Stork, D. G., *Pattern Classification (2nd Edition)*, Wiley-Interscience, 2000.