Discovering Anomalous Aviation Safety Events using Scalable Data Mining Algorithms

Bryan Matthews* Santanu Das† Kanishka Bhaduri‡ Kamalika Das* Rodney Martin§
Nikunj Oza§ Ashok N. Srivastava§ and John Stutz§

NASA Ames Research Center, Moffett Field, CA 94035, USA

The world-wide civilian aviation system is one of the most complex dynamical systems ever created. Most modern commercial aircraft have onboard flight data recorders (FDR) that record several hundred discrete and continuous parameters at approximately 1 Hz for the entire duration of the flight. This data contains information about the flight control systems, actuators, engines, landing gear, avionics, and pilot commands. In this paper we discuss recent advances in the development of a novel knowledge discovery process consisting of a suite of data mining techniques for identifying precursors to aviation safety incidents. The data mining techniques include scalable multiple kernel learning for large-scale distributed anomaly detection. A novel multivariate time series search algorithm is used to search for signatures of discovered anomalies on massive data sets. The process can identify operationally significant events due to environmental, mechanical, and human factors issues in the high dimensional Flight Operations Quality Assurance (FOQA) data. All discovered anomalies are validated by a team of independent domain experts. This novel automated knowledge discovery process is aimed at complimenting the state-of-the-art human-generated exceedance-based analysis that fails to discover previously unknown aviation safety incidents. In this paper we discuss the discovery pipeline, the methods used, and some of the significant anomalies detected on real-world commercial aviation data.

I. Introduction

As the complexity and traffic density in the the world-wide air transportation system increases, it is going to become more important than ever to develop advanced techniques to help keep the fatal accident rate at the extremely low levels that they are today. A study conducted by Boeing in 2006 shows that the fatality rate in that year was less than 1 per 1 million departures. While this rate is exceedingly small, many aviation experts are concerned that the unprecedented growth in air travel may create a situation where the current safety levels cannot be sustained. This paper demonstrates a new knowledge discovery process combining novel data mining algorithms developed specifically for detecting precursors to aviation safety incidents that represent a significant advancement in the state of the art.

Current methods rely on human experts to create massive rule-based systems that detect known safety issues, based on whether a small set of parameters exceed some predefined thresholds. This approach, based on known 'exceedances', allows analysts to quickly search large databases for predefined issues. Based on the result of this search, the analyst can recommend new training or operational procedures, maintenance action, or another type of intervention to appropriately address the issue.

The approach that we are discussing here is fundamentally different because, rather than using human-generated rules to detect potential known issues, we use anomaly detection techniques that scan large multivariate time series databases to uncover statistical anomalies. Our knowledge discovery process outputs a set of statistically significant anomalies from historical observed data. Not all of these anomalies are necessarily significant from the aviation operations perspective. Further input from domain experts such

*SGT, Inc., NASA Ames
†UCSC, NASA Ames
‡MCT Inc., NASA Ames
§NASA, NASA Ames

as pilots and airlines safety management teams help us categorize these anomalies, sometimes base on additional contextual information. Once these unique anomalies are discovered, there is a need to determine the frequency of these occurrences in massive historical time series databases. We have designed and adapted multivariate time series search algorithms to efficiently look for high-dimensional anomalies in these massive time series data sets to address this need.

II. Related work

Anomaly detection is an active area of research in data mining and is widely used in many application areas such as finance, social networks, e-commerce, earth science, and aeronautics. Broadly speaking, outliers can be detected using unsupervised, supervised, or semi-supervised techniques.[7,14] Unsupervised techniques, as the name suggests, do not require labeled points for detecting outliers. In this category, the most popular methods are distance-based and density based techniques, in which outliers are points in low density regions or points farthest away from other points. In their seminal work, Knorr *et al.*[17] proposed a distance-based outlier detection technique based on the idea of nearest neighbors. The naïve solution has a quadratic time complexity since every data point needs to be compared to every other, in order to find the nearest neighbors. To overcome this, researchers have proposed several techniques such as the work by Angiulli and Pizzuti,[1] Ramaswamy *et al.*,[20] Bay and Schwabacher,[3] and Bhaduri *et al.*[4] Density based outlier detection methods, on the other hand, flag a point as an outlier if the point is in a low density region. Using the ratio of training and test data densities as an outlier score, Hido *et al.*[13] have proposed a new inlier-based outlier detection technique. Supervised techniques require labeled points of both normal and abnormal data for first building a model (*e.g.* a classifier) and then testing if an unknown data point is an outlier. The model can be probabilistic, based on Bayesian inferencing[9] or deterministic, such as decision trees, support vector machines and neural networks.[15] Very recently, Srivastava[23] have used supervised methods for detecting aircraft having abnormal fuel consumption in a fleet commercial aircraft. Semi-supervised techniques only require labeled points of normal data. Therefore, they are more widely applicable than the fully supervised ones. These techniques build models of normal data and then flag as outliers all those points which do not fit the model. The applications we are interested have very few labeled data points and we therefore resort to unsupervised outlier detection techniques.

In the context of outlier detection from aviation data, several papers that use support vector machines have been published. Das *et al.*[10] present a technique for speeding up 1-class SVM using a sampling strategy. The authors show that the proposed technique is 15 times faster than the traditional 1-class SVM while maintaining accuracy. Das *et al.*[12] have also developed an anomaly detection method which can work with both continuous and discrete sequences to demonstrate how some significant anomalies can be detected from real aviation operational data. Auto pilot systems allow an aircraft to be flown using the flight computers and thereby may minimize pilot errors during flight. However, the auto pilot systems are controlled using switches in the cockpit which are manipulated by the pilot. SequenceMiner,[6] has been shown to detect abnormal switching from a learned model of normal switching. It helps an analyst to identify abnormal pilot inputs used to control the auto pilot system. Usefulness of these techniques for aviation safety has been studied in[11] in which the authors have applied these data mining algorithms on real aviation data sets to get meaningful results. It is important to note that the choice of algorithms for anomaly detection in aviation data is based on a number of factors such as the data set at hand, the types of anomalies we are interested in finding, and the scale of the problem. What is proposed in this paper is a systematic way of combining the strengths of these algorithms in a way that helps an analyst identify new and existing precursors to aviation safety incidents. The algorithms that are part of our system will be discussed in details in Section IV.

The ultimate goal is develop a process that can compliment the existing state-of-the-art human-generated exceedance-based method to uncovers some operationally significant events due to environmental, mechanical, and human factors issues in high dimensional, multivariate Flight Operations Quality Assurance (FOQA) database. Although there exists a number of algorithms and publications that discuss the use of anomaly detection methods for aviation safety, to the best of our knowledge, this is the first paper that describes the entire knowledge discovery pipeline for aviation data using scalable data mining algorithms. This paper highlights the value of the proposed framework in unearthing safety critical events in aviation, particularly those that are related to human automation interaction: a class of events that are becoming increasing important due to the increasing complexity of automation on board the aircraft. Our framework is the first in aviation analytics that allows automated discovery of such precursors to aviation safety incidents.

III. Background

The world's collection of aircraft and the airspace in which they operate, which we will collectively refer to as "the airspace" from now on, is a complex system that generates a large amount of data, and as such is a challenging domain for data mining. The airspace contains many elements that are in the critical path of operation such as aircrafts, airports, people (flight crew), weather events, and routes. Each contains many subcomponents. For example, each aircraft contains many subsystems and components, each of which may have experienced a variety of stresses and maintenance actions. Each airport has multiple runways. These elements that make up the airspace interact in many complex ways. For example, each flight is an aircraft operated by a flight crew to go from one airport to another airport via a route that may be set up or may need to be changed to avoid weather events. Of course, multiple flights operate in the airspace at any given time, and the flights need to coordinate to avoid adverse events (incidents or accidents) while maximizing throughput and minimizing delays.

The airspace provides a very rich and challenging application area for data mining. The complexity described above manifests itself in a substantial amount and variety of data. There are many data types, only some of which are standard independent, identically distributed data that most methods assume. Data can be temporal, spatial, or spatiotemporal. It can have different sampling rates across different dimensions. Some of it is structured, such as the FOQA data, whereas other kinds of data, such as safety reports and maintenance reports, are unstructured. The data has varying levels of fidelity and accuracy. Methods currently used in the aviation industry to analyze this data are unable to cope with most of this complexity and are typically restricted to simple threshold checks over a small number of variables. While these methods run relatively quickly and are easy to understand, we hypothesize that data mining methods that are more capable of coping with the complexity of airspace data, will find operationally significant anomalies that are not revealed by current methods.

We fortunately have an air transportation system that is extremely safe. Even though ostensibly the safety and efficiency of the air transportation system seems like a solved problem, there is still room for improvement due to the potential threefold increase in air traffic in the next few decades as well as room for cost reduction through green aviation measures. Furthermore, adverse events, when they happen, can have devastating consequences. Therefore, improved safety and efficiency based upon the use of various growing data sources remain important objectives. Data mining methods play a critical role in understanding the airspace, not only to correct problems that may be causing "near-misses" now, but also to anticipate problems that are likely to get worse as the Next Generation Air Transportation System (NextGen) comes about due to increasing air travel, increasing automation in aircraft, and other unforseen factors. We expect that there are sequences of events hidden in airspace data that lead to or could lead to an adverse event, and call these sequences precursors. It is critical to identify these precursors, so that an on-time detection of such a precursor in flight configuration or flight manoeuvre can alert the concerned authority to take necessary action to avoid the safety incident in future flights.

IV. Aviation safety knowledge discovery process

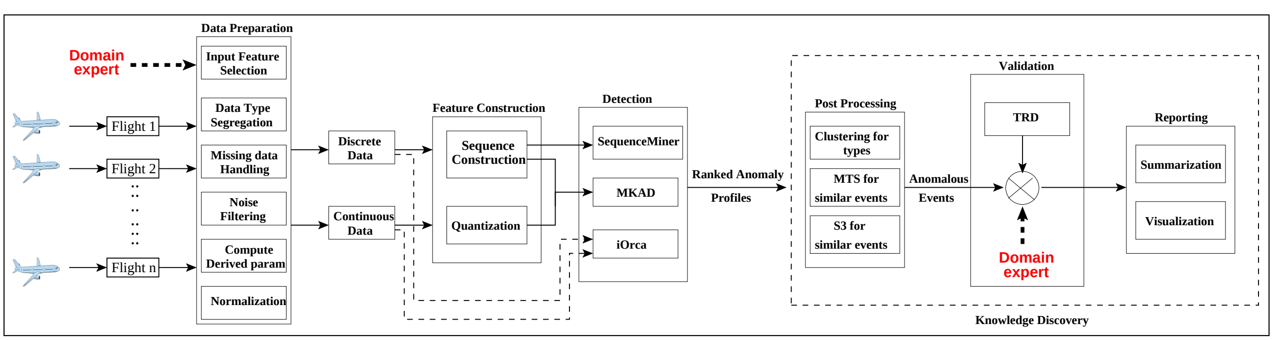


Figure 1. Aviation Safety Knowledge Discovery (AvSKD) process for identifying abnormal aviation events.

In this section we describe the entire proposed knowledge discovery process for analyzing aviation data, that we will refer to as the Aviation Safety Knowledge Discovery (AvSKD) process. Figure 1 presents a flowchart for this framework. We discuss each block in details in the following sections.

A. Data and preprocessing

Input data: Raw FOQA data from each aircraft is collected and input to the AvSKD process. Each flight record is a matrix with rows corresponding to time samples and columns to observed parameters. The samples are collected at frequency of 1 Hz and a typical flight from takeoff to landing consists of 5000-6000 samples and 350 parameters. The data is heterogeneous in the sense that there are both discrete (binary and categorical) and continuous parameters.

Data preparation: This raw data for each flight is then passed through the data preparation module which performs several functions: feature selection, data type segregation, missing data processing, noise filtering, and normalization. Depending on the type of safety study that is undertaken, aviation experts help us choose the variables (feature selection) which are most relevant for the analysis. Once the variables are selected, the next sub block separates the discrete and continuous variables so that different data summarization techniques can be applied to them independently. Flight recorded data is often contaminated with missing data, out of bounds variables, noisy recordings, amplitude spikes etc. caused by sensor malfunctions or recording medium errors. Our next sub block applies several data quality filters specifically developed for this purpose, to clean the data. Derived parameters may help to understand the hidden state of the aircraft based on some set of observed parameters *e.g.* estimated aircraft speed margin above stall speed based on flap settings, gross weight, and velocity. These parameters have specific advantages for tracking particular events. We construct some derived parameters depending on our study and expert inputs. In the last step of the data preparation phase, we either normalize the continuous features using z-score, or 0-1 normalization or convert categorial features into binary ones. The outputs of this block are two time series corresponding to selected discrete and continuous variables for each flight.

Feature construction: Once the discrete and continuous parameters are separated, different techniques are applied on them for feature construction. The continuous data is quantized over a window length and in amplitude to convert them into a SAX representation,[19 12] The discrete parameters are handled by marking the on and off transitions between switch states with unique symbols and concatenating the symbols into a sequence vector while preserving the time ordering.

B. Anomaly detection algorithms

Once the data is cleaned and formatted as desired, knowledge discovery algorithms are applied. AvSKD is a very flexible platform and many anomaly detection algorithms can be easily plugged in. To keep the discussion short, we discuss a couple of algorithms that we have used in our example discovery process. [a].

(1) **MKAD**: The Multiple Kernel Anomaly Detection (MKAD) algorithm is designed to run on heterogeneous data sets. Heterogeneity in data may result from the presence of multiple attribute types *e.g.* discrete and continuous. MKAD[12] is a "multiple kernel"[2, 18] based approach where the major advantage is the method's ability to combine information from multiple data sources. Multiple Kernel Learning (MKL) takes advantage of the mathematics of kernels allowing users to derive new kernels from existing kernels, provided each kernel satisfies the "Mercer condition", that the kernel function must be continuous, symmetric, and positive definite. In particular, the resultant kernel K can be a convex combination of all kernels computed over multiple features i.e. $K(\vec{x}_i, \vec{x}_j) = \sum_{p=1}^{n} \eta_p \hat{K}_p(\vec{x}_i, \vec{x}_j)$, with $\eta_p \geq 0$ and $\sum_{i=1}^{n} \eta_p = 1$. Here $\hat{K}_p(\vec{x}_i, \vec{x}_j)$ represents the p^{th} kernel computed for either discrete or continuous parts of data points x_i and x_j, and η_p are to weight individual kernels (in this paper, we always use $\eta_p = 0.5$ and n=2 **unless otherwise specified**). There exist several classes of kernel which coincide with the Mercer kernel, such as radial basis function (RBF), polynomial, bag-of-words, sigmoid, spline, graph based, tree based, mismatch based functions etc .[22, 16, 8] Although the current MKAD approach is not limited to the choice of kernel functions, for the purposes of this study, an $nLCS$[6] based kernel function is used to model switching sequences for the process, where the order of the switching is important from operational perspective. Normalized Longest Common Subsequences (nLCS) can be defined as $nLCS(X, Y)$ where X and Y are two sequences of discrete

[a]Some or all these algorithms can be downloaded from (Left blank for blind review)

symbols and

$$nLCS(X,Y) = \frac{|LCS(X,Y)|}{\sqrt{|X||Y|}}. \tag{1}$$

Given two sequences X and Z, Z is a subsequence of X if removing some symbols from X produces Z. Z is a common subsequence of sequences X and Y if Z is a subsequence of both. The longest such subsequence is called the longest common subsequence (LCS) and is denoted by $LCS(X,Y)$ and $|LCS(X,Y)|$ is its length. This kernel is used because there exists a standard operating procedures for flying the aircraft and the sequence of the pilot inputs (or actions) along with the measured quantities or parameters are extremely meaningful. $nLCS$ is a useful metric for measuring similarity between discrete sequences. These sequence features can be generated directly from discrete parameters or from SAX representations[19] of continuous variables as described in.[12]

The heart of MKAD is a one-class SVM model that constructs an optimal hyperplane in the high dimensional feature space to separate the abnormal (or unseen) patterns from the normal (or frequently seen) ones. This is done by solving the following optimization problem:[21]

$$\begin{array}{ll} \min & Q = \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j K(x_i, x_j) \\ \text{subject to} & 0 \le \alpha_i \le \frac{1}{\ell\nu}, \sum_i \alpha_i = 1, \rho \ge 0, \quad 0 < \nu < 1 \end{array} \tag{2}$$

where α_i's are Lagrange multipliers, ℓ is the number of examples, ν is a user specified parameter that defines the upper bound on the training error, and also the lower bound on the fraction of training points that are support vectors, ρ is a bias term and K is the kernel matrix. Once this optimization problem is solved at least $\nu\ell$ training points with non-zero Lagrangian multipliers ($\vec{\alpha}$) are obtained and the points for which $\{x_i : i \in [\ell], \alpha_i > 0\}$ are called support vectors. Once $\vec{\alpha}$ is obtained, we can compute the decision function:

$$f(\vec{z}) = sign\left(\sum_i \alpha_i \sum_p \eta_p \hat{K}_p(\vec{z}, \vec{x}_i) - \rho\right)$$

to predict positive or negative label for a given test vector $\vec{z}$. Examples with negative labels are classified as outliers.

MKAD is highly scalable and it can process large data sets very fast. We conducted a scalability study where we were able to process data of 940,000 flights (equivalent to over 5 billion data points) in 12.5 hours using distributed computing.

(2) **iOrca**: Another unsupervised anomaly detection method based on distance to nearest neighbors as the measure of "outlierness"[43] is also used in this framework. The basic idea of distance-based outlier detection is to find, for each point x in the data set, k other points which are closest to x by computing the Euclidean distance between x and all the other points in the data set. Then the points are ranked (from highest to lowest) according to the average distance (or maximum distance) to the k-nearest neighbors. This average distance is known as the *outlier ranking function* or *outlier score*. The ranked list becomes the set of outliers in the order specified by the score. It is easy to verify that this computation has a computational complexity of $O(n^2)$ where n is the number of points in the data set. In order to speed up this computation the authors of,[3] have proposed a state-of-the-art algorithm Orca which uses a pruning strategy to quickly remove the normal points which are the bulk of the population. To implement pruning, one needs to keep track of the smallest outlier score of the outlier set (call it the cutoff c). Then, while testing a new point x, its nearest neighbors are searched and whenever a neighbor of x is found whose distance to x is less than c, then x is discarded from the pool of outlier candidates.

More recently, Bhaduri *et al.*[4] have proposed a new method Index-Orca (iOrca) which is at least an order of magnitude faster than Orca, while still guaranteeing correct results. Inefficiency of Orca is due to its slow update of cutoff (hence pruning). To overcome this,[4] proposes a data reordering scheme such that: (1) the cutoff is updated faster, and (2) for every point, the nearest neighbors are found quickly, often in constant time.

In order to update the cutoff faster, iOrca processes the outliers as early as possible. To achieve this, a random point R is first selected as a reference point from D and then the distance of all the other points in D from R is calculated. This array of distances is the *index* which is then used to reorder the points in D with increasing distance from R. Instead of traversing through the data in the original order (or random order[3]), iOrca tests the points along this index order. The rationale for this traversal is as follows. Since R is chosen at random, it is likely that R will be one of the inlier points if there are more inliers than outliers. Therefore, the points farthest from R will be more likely be the outliers. Since the points in D are processed in decreasing distance to R, it is very likely that the outliers will be processed first, leading to a faster increase in the cutoff threshold. Re-ordering the database for testing also allows iOrca to terminate without ever needing to process all the data:

Lemma IV.1 *[Stopping rule] Let L be the index as described in this section. Let R be the reference point used to build L. Let x_t be any test point currently being tested by iOrca. If*

$$\|x_t - R\| + \|R - x_{n-k+1}\| < c,$$

then iOrca can terminate the execution immediately, where c is the current cutoff threshold, x_{n-k+1} is the true k-th nearest neighbor of R and $\|\cdot\|$ is the 2-norm of a vector.

Finally, nearest neighbors of a test point are searched along the index order. Instead of starting the search from the beginning of the database for every test point, iOrca starts the search from the location it lies along the index and then gradually "spirals" out. Since the index prescribes a total ordering of the points projected along one dimension, it is likely that, for a test point x_t, if x_i is closer to x_t than x_j along the index, then it will be the same even when the actual distances are also computed *i.e.* if

$$|\|x_i - R\| - \|x_t - R\|| < |\|x_j - R\| - \|x_t - R\||$$

then it is expected that

$$\|x_t - x_i\| < \|x_t - x_j\| .$$

As shown in,[4] iOrca is highly scalable and often produces near-linear running time due to the index and early stopping criterion.

Both algorithms discover anomalies. They also decompose the overall anomaly score to reflect the contribution of each parameter towards that score, thus providing a rough estimate of the most faulty parameter. These two methods described above are not intended to solve the same problem, but rather to detect anomalies at different levels of abstraction. For instance, one important distinction between MKAD and iOrca is that MKAD works with the data compressed to a single vector for each flight (using SAX and sequencing), while the iOrca ingests all the raw time series data coming from the flight data recorder. Since MKAD is working with the data at a lower resolution, it reports anomalies at the fleet level, *i.e.* it identifies anomalous flights, whereas iOrca is more tuned to detect anomalies within a flight because it analyzes the uncompressed time series directly.

C. Knowledge discovery

Post processing: Both these outlier detection algorithms output anomaly scores and the location of the anomalies. Depending on the type of algorithm used, we can either identify an entire flight as abnormal (*e.g.* MKAD) or point out at which point in time the abnormality has been found (*e.g.* iOrca). Depending on the size of the input database, the algorithms may discover hundreds to thousands of ranked anomalies, making it difficult for domain experts to validate all of them. We have adopted a variant of k-means algorithm that helps us to cluster the found anomalies based on the contribution score of each parameter. Then, instead of presenting all the anomalies to the domain expert, we only show them some representative examples from each group.

Once we find some interesting anomalies using the anomaly detection techniques, we are interested to know the frequency and severity of those events in the entire data set. To do this, we use a tool called Multivariate Time-Series Search (MTS).[5] For each anomaly found earlier, this tool searches for similar anomaly evidences in the entire database. The MTS tool takes as input the data files (which can be very large), a query in the form of a multivariate time series defined over a small subset of variables, and a threshold ϵ specifying the radius of the search with respect to the query. It returns all the location pairs in the form of (*File number, Time instance*) where a match has been found. In the preliminary phase, an index is built on the data set. This consists of selecting a random point from the data set (called the reference point) and then rearranging all the other points in the data set according to distance to this fixed point. This process is repeated separately for all the variables in the data set, even before the query is provided.

When the query q arrives in the form of a multivariate waveform, first it is mapped to the 'index space' by subtracting the reference point and then a binary search is executed to find the location of the transformed query q' on the index. Using triangle inequality it is easy to show that the only candidates of interest are the tuples in the range $q' \pm \epsilon$ which guarantees no false dismissals. These candidates are then fetched from the data set and false positives are removed by doing an exact calculation with q.

Similarly, for anomalies in discrete sequences, such as those found by MKAD with contributions from the discrete kernel, we pass the examples to our tool Sequence Similarity Search (S3). All examples are converted into sequences using the preprocessing step described earlier. The query sequence is then compared against all sequences in the data using the nLCS metric as the distance function described in Eqn. (1). The distances are sorted and a user defined cutoff is applied to determine the matches.

Validation: Once we find anomalous events using the knowledge discovery algorithms described above, we validate them. There are three sources of validation that are generally followed in the aviation industry. The first level of validation is with the help of experts in the field. For our case study we have sought help from four aviation experts with differing backgrounds. The first domain expert has over 30 years of experience in human factors and aviation, text analysis and statistical methodologies. The second expert is a retired commercial airline pilot of a major US carrier with over 35 years of experience in flying Boeing 777 and 747 aircrafts. Our third expert is a researcher in human factors with special emphasis on aircraft automation and its effects on human performance. He is also a commercial pilot and flight instructor for single and multi-engine aircrafts including the Airbus aircraft. Our fourth domain expert is also a researcher in the human factors with emphasis in cognitive science and psychology as applicable to spatial reasoning, decision making, risk assessment, communication, and skill acquisition and retention for air traffic controllers, airline pilots, space mission controllers, and astronauts. He has over 35 years of experience in flying single engine and multi engine commercial aircraft including certifications for Airbus A320, A330, and Boeing B737.

As further validation, a repository of text reports, known as the Text Reports Database (TRD) written by a member of the flight crew can potentially be used to confirm the anomaly with the pilots depiction of the event. While the numeric FOQA data gives us information as to *what* happened, the text reports, *if available*, provide information as to *why* the event happened. A third way of validation is by interviewing the crew themselves once an event is found. However, in many cases we do not have access to either the reports database or the crew, and so we mainly rely on experts in the field for validation.

Reporting system: The final step in the AvSKD process is the report generation phase. We have developed a web browser compatible reporting system [b] which displays each anomalous flight and the top few parameters due to which it is deemed anomalous. For a given anomalous flight ID it reports the following:

- Percentage contribution: Discrete, Continuous
- Graphical plot: Contributing continuous parameters
- Percentage contribution of each continuous parameter
- Missing and extra discrete switches

V. Discoveries

This section will discuss three examples of anomalies found by each of the detection algorithms. It is important to note that the choice of the examples presented in this paper is based on the fact that both the domain experts and airline agreed that they are compelling incidents. Since this is a discovery problem on unlabeled data, the results can be validated for ground truth only by domain experts. It can be argued that the current threshold based methods can be used to look at false positive and false negatives rates to generate ROC curves, and therefore tune the algorithms' performance. However, this approach will only effect the algorithms' ability to detect what is currently being detected and not improve the algorithms ability to detect unknown operationally significant events. Two of the examples identified are due to pilot controlled manoeuvers (see Sec. 1 & 1). The other example involves a human-machine interaction scenario that is a significant challenge for the pilots to maintain awareness of the modes of the flight computer (see Sec. 2). All three examples are important safety events. They appeared within the top ranked anomalies, however the rank is determined by the statistical severity of the identified anomalous flights. In other words, anomalies with operational significance may not always end up on the top of the list. In FOQA data there are several

[b] A sample of the report can be found here: (Left blank for blind review)

natural sources of homogeneity, for example: flights that have common origin or destination airports, city pair routes, tail numbers, aircraft models, as well as seasonal aspects, such as flights within a month, and can be grouped accordingly. Even within a flight there exist several phases such as take-off, cruise, and landing. In this study we partition the data based on departure or destination airport and analyze the take-off and landing respectively. The set of continuous parameters considered include: altitude, airspeed, roll, pitch, engine RPM, wind speed, estimated airspeed above stall, and various control surface positions. The binary state parameters considered include: landing gear, landing flap position, whether the autopilot and/or flight director was engaged, and various vertical and lateral guidance system modes.

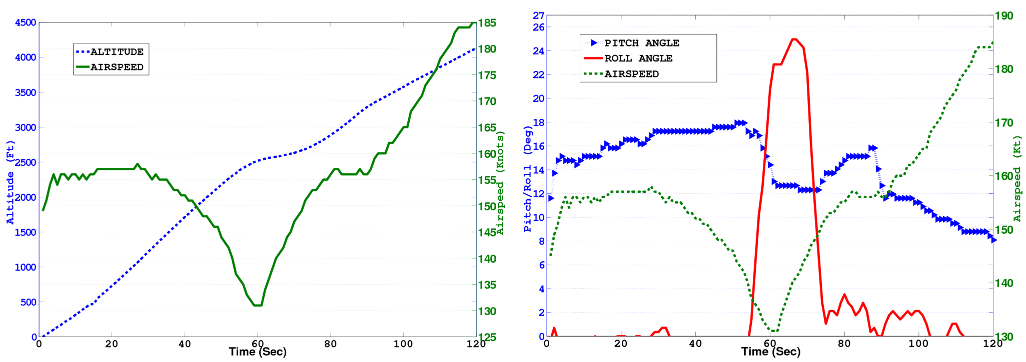
Continuous: `<Altitude, Airspeed, Roll, Pitch, Angle of attack, Engine, RPM, Wind speed, Computed airspeed above stall, Aileron, Rudder, Stabilizer, Elevator, etc.>`

Discrete: `<Landing gear, Landing flap position, Autopilot and/or flight director status Various vertical and lateral guidance system modes`[c]`>`

A. Fleet level anomalies

The following two anomalies were identified at the fleet level, meaning individual flights were labeled anomalous by the algorithm and not the samples within the flight. Further analysis was performed in the post processing block to look within the flight to determine what characteristics of the flight were abnormal.

1. *Drop in airspeed*



(a) Drop in airspeed with the corresponding altitude at which it occurred

(b) Airspeed recovery as the pilot performed a roll maneuver and simultaneously began correcting the pitch to begin increasing airspeed

Figure 2. Drop in airspeed

An anomaly was found in a flight by MKAD, using both the continuous and the discrete parameters listed at the beginning of Section V as input, and formatted as described in Section B. The algorithm was run on all available flights departing from a particular airport, identifying this flight as an anomaly, which was ranked 11th out of 439 flights, with a runtime of 3 seconds. The higher ranked anomalies were examined and found to have various statistically anomalous characteristics (such as lower airspeeds, high wind speed, or an uncommon departure direction), however the behavior in this flight was deemed to be of elevated interest by the domain experts.

Takeoff procedures typically allow aircraft to maintain an airspeed of $V_2 + 10$ knots (Kts), where V_2 is the minimum speed that needs to be maintained up to acceleration altitude. Acceleration altitude is the altitude (typically 3,000 ft above ground level [AGL]) at which pitch is reduced slightly and positive climb maintained, allowing acceleration through flap retraction speeds to 250 Kts. In the flight shown in Figure 2(a), approximately 30 seconds after takeoff, the aircraft began to experience a drop in airspeed. Given the gross weight and flap settings on the aircraft at the time, the aircraft decelerated to only 12 Kts above the estimated stall speed. The experts agreed that this flight was of interest, since it is highly unusual at this point in the flight to experience such a drop in airspeed. This drop in airspeed was a direct result

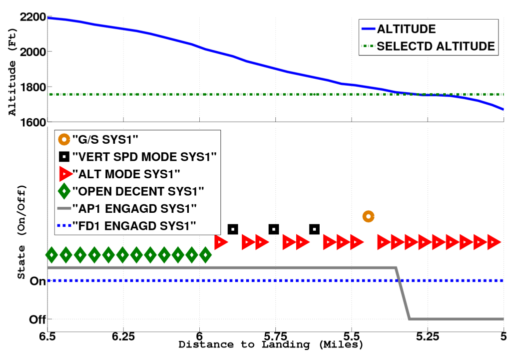
[c]For further explanation of automated modes please refer to the Airbus/Boeing flight crew training manuals here: http://www.737ng.co.uk/docs.htm.

of the pitch angle being held too high for too long, under the conditions that both engines were running at max climb thrust, and the autopilot was not yet engaged. The drop in airspeed continued for another 30 seconds. At that time a turn maneuver was executed with a simultaneous correction of the pitch angle (see Figure 2(b)), allowing the aircraft to start increasing airspeed again.

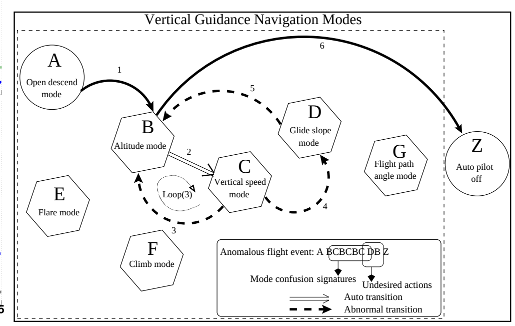
To find similar incidents, we ran MTS search tool on a particular airport which is known to the domain experts to have unusual takeoff procedures. The airport had 9543 flights consisting of roughly 1 million data points. We have used only 2 parameters for search: altitude and airspeed, and the above example as the query. Building the index took 21 mins (only takeoff phase). Search for other similar events took 0.08, 0.09, and 0.1 secs for thresholds of 0.5, 1, and 2, returning 4, 17 and 69 hits respectively. While visual inspection reveals similarity with the query, we are currently validating them with our experts.

2. Mode confusion

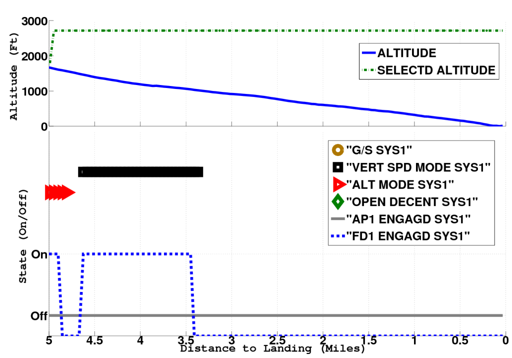
An anomaly was found in a flight by **MKAD**, using only the discrete parameters listed at the beginning of Section V as input, 100% weights on the discrete kernel, and formatted using the sequence preprocessing steps described in Section B. An additional step was taken to ignore the flight computer modes when both the autopilot and flight director were disengaged, since no action would be taken by the autopilot or pilot in response to these mode states and therefore they would not be relevant to this study. The algorithm was run on all available flights landing at a particular airport with a runtime of **210** seconds. This anomaly ranked **13th** out of 19,243 flights. Again the higher ranked anomalies were investigated and found to be higher ranked due to modes in the data that are infrequently used (such as flight path angle mode and altitude mode late in approach) however; this particular event was found to be of high interest by the domain experts. The anomaly description follows:



(a) Evidence of mode confusion with the unusual mode switching



(b) Conceptual diagram of the state space and path taken for the vertical guidance system in this example



(c) Confirmation of mode confusion with the recycling of the flight director

Figure 3. Mode confusion

On descent for approach the aircraft was slightly above, yet on track to intercept the glide slope. At approximately 2000 ft AGL the auto pilot vertical guidance system was manually switched from Open Descent mode into Altitude mode. The auto pilot immediately reverted to the default Vertical Speed mode. Manual selection of altitude mode was attempted repeatedly with the same result (see Figure 3(a)). Glide Slope mode was engaged momentarily just before the autopilot was manually disengaged, at which time Altitude mode was manually reselected. Figure 3(b) presents the corpus of vertical guidance modes for this particular flight, and helps to identify the potential anomalous sequences of switching initiated by the human. The anomalous mode transitions are shown in dashed lines in Figure 3(b). There are two important observations. First, we see a signature of mode confusion where the operator repeatedly attempted to override mode C with B (see Figure 3(b)), however the flight computer did not accept the command and auto transitioned back to C. If the aircraft had been closer to the selected altitude the flight computer would have accepted the command and the unnecessary mode switching could have been avoided. Secondly, this was followed by a sequence of undesired actions (C $\rightarrow$ D $\rightarrow$ B) briefly before the pilot removed himself/herself from the undesired automation by manually disengaging the autopilot resulting in mode Z. The domain experts identified these sequences of actions as potential evidence of mode confusion. At this point the aircraft was hand flown to level off at the selected altitude. A new altitude of 2700 ft AGL was then selected, presumably to program the flight computer for the missed approach maneuver in case of a go around. At this point, the flight director was manually reset, which resulted in the default Vertical Speed mode being selected (see Figure 3(c)). The aircraft was then manually flown using glide slope indicators and also by using the flight director in Vertical Speed mode. At 1000 ft AGL the flight director was disengaged for the final time, and this was followed by a visual landing. Repeated failed attempts to select the desired vertical modes, subsequent disengagement of the autopilot, and recycling of the flight director all suggest mode confusion, which added unnecessary workload to an already highly demanding situation.

A search for similar examples was performed on the entire data set, using S3. The sequence from the above example was used as the query and compared against the full set of landing sequences in the data, with a running time of 127 Mins. One other flight was found with near identical behavior, while 12 other flights were found to have either the unusual switching behavior or the recycling of the flight director but not both.

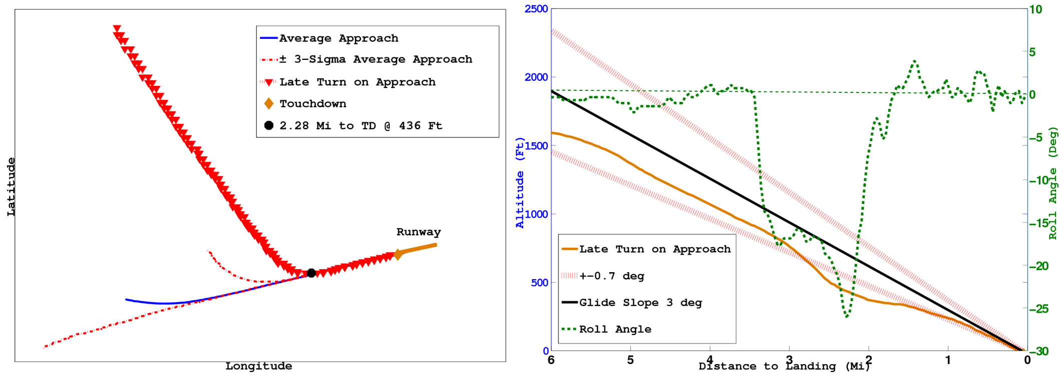
B. Flight level anomalies

The last anomaly was identified at the flight level, meaning time samples within a flight were labeled anomalous by the algorithm. One advantage in using this approach is that the algorithm can directly identify the abnormal characteristics within the flight, however there is a trade off with runtime on larger data sets.

1. Unstable approach

An anomaly was found in a flight by iOrca, using the set of continuous parameters listed at the beginning of Section V as input, and formatted as described in Section IV. The top N was chosen to be $1000 \approx 0.2\%$ of the full data and k neighbors was set to the algorithm's default value of 5. The algorithm was run on all available flights landing at a particular airport with a runtime of 622 seconds using 519,025 time samples. This flight was identified as an anomaly and ranked 22nd out of 2800 flights. Again, the higher ranked anomalies were investigated and found to be statistically unusual (higher airspeeds at lower altitudes) however, this particular event was found to be of notable interest by the domain experts. The anomaly description follows:

For this flight, an aircraft was being flown manually during the approach without the flight director, presumably for a visual approach, since the reported weather was clear. The aircraft was being flown slightly under a three degree approach path and was on a base leg (the part of the approach that is perpendicular to the runway) (see Figure 4(a) for flight track). An unusually sharp turn was executed to intercept the final approach path at 2.28 mi. During this turn the aircraft was below 500 ft AGL, and the turn required a bank angle of 26 degrees to intercept the final approach course (see Figure 4(b)). Such a turn so close to the airport is undesirable, since it does not allow much time for the aircraft to meet stable criteria for landing, which could result in a go around. However, in this case, correction of the aircraft's attitude allowed for stabilized conditions to be met before touching down safely.



(a) Late turn on approach compared to the average flight track for this runway.

(b) Glide slope compared to the 3 degree glide path showing the altitude at which the roll was performed.

Figure 4. Unstable approach

C. Value of Discoveries

The AvSKD process has helped in identifying important precursors to undesirable events. We are defining precursors as the combination of conditions that increase the likelihood of potential unsafe situations in the future. In the above example 1 the drop in airspeed may have led to a stall warning. Similarly the mode confusion event in 2 could have lead to a failure to reach stabilized landing criteria, and the unstable approach in 1 might have resulted in a go around or hard landing. Insights derived from the precursors identified by the AvSDK process can be brought to the attention of the airline's training personnel and policy makers, which may lead to operational changes to address safety related issues.

The current FOQA program is designed to provide answers to questions that the domain experts have thought to ask. Whereas, in the AvSKD process we are not focusing on any particular group of anomalies, instead we "search for the unexpected" from the entire data set and consult the domain expert to validate each of them. Exceedance-based programs are like mechanical filters that need prior knowledge of the events before they can be implemented. Looking at the entire data using exceedance-based approach is not scalable, and therefore has been implemented only on the portions of the flights as prescribed by the domain experts, leaving many unexamined portions of the data. In our approach the algorithms neither need prior knowledge of the events nor do they examine only a subset of the data. Most of the algorithms presented in this paper have the ability to handle large data sets and therefore allow us to process the full data set in a reasonable amount of time. It is also important to note that FOQA programs are dependent on the validity of the thresholds defined by the domain experts, whereas our methods use statistical properties of the data to rank the anomalies and find their severities. If multiple parameters interact in a counter-intuitive way the exceedance-based-approach might overlook a subset of those parameters if they are not predefined by the domain experts, whereas our multivariate models can take into account the complex relationships and report their contributions to the anomalies.

VI. Conclusion

In this paper, we have described a data mining and knowledge discovery process that can detect precursors to aviation safety incidents. We demonstrated this process on a large commercial aviation data set by showing that it identified three operationally significant events that were validated by experts. We plan to extend this work to identify anomalous sequences across a set of flights that have a key characteristic in common, such as the same aircraft, flight crew, or city pair. We are also working to identify and incorporate other sources of data that correspond to these flights, such as radar track data and weather data, as these data will also be useful in identifying anomalies. We are also currently working on making this framework even more scalable to facilitate larger scale analysis. Finally, it should be noted that although the methods we describe are for the aviation domain, all these techniques are domain agnostic, and can be used in any application area that exhibits similar data properties and challenges. For example, in recommender systems the task it to maximize the click through rate by good quality recommendations based on models learned from user browsing data. These models are learned on continuous data and the discrete signals may encode seasonal patterns which help in model segmentation and selecting the appropriate model for predicting user behavior. Anomaly detection in this case relates to finding unique, noisy or fraudulent users for further analysis or building better models.

Acknowledgment

This research is supported by the NASA Aviation Safety Program, System-Wide Safety and Assurance Technologies Project. We would also like to thank Irv Statler, Bob Lawrence, Mike Feary, Immanuel Barshi and the partner air carrier for providing data and expertise.

References

[1]F. Angiulli and C. Pizzuti. Outlier mining in large high-dimensional data sets. *IEEE Transactions on Knowledge and Data Engineering*, 17(2):203–215, 2005.

[2]F. Bach, G. Lanckriet, and M. Jordan. Multiple kernel learning, conic duality, and the smo algorithm. In *International Conference on Machine Learning*, 2004.

[3]S. D. Bay and M. Schwabacher. Mining distance-based outliers in near linear time with randomization and a simple pruning rule. In *KDD '03: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 29–38, 2003.

[4]K. Bhaduri, B. Matthews, and C. Giannella. Algorithms for Speeding up Distance-Based Outlier Detection. In *Proceedings of SIGKDD'11*, pages 859–867, 2011.

[5]K. Bhaduri, Q. Zhu, N. C. Oza, and A. N. Srivastava. Fast and Flexible Multivariate Time Series Subsequence Search. In *Proceedings of ICDM'10*, pages 48–57, 2010.

[6]S. Budalakoti, A. Srivastava, and M. Otey. Anomaly detection and diagnosis algorithms for discrete symbol sequences with applications to airline safety. *Trans. Sys. Man Cyber Part C*, 39:101–113, 2009.

[7]V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Computing Surveys*, 41(3):1–58, 2009.

[8]O. Chapelle and P. Haffner. Support vector machines for histogram-based classification. *IEEE Transactions on Neural Networks*, 10(5):1055–1064, 1999.

[9]K. Das and J. Schneider. Detecting anomalous records in categorical datasets. In *KDD '07: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 220–229, 2007.

[10]S. Das, K. Bhaduri, N. C. Oza, and A. N. Srivastava. ν-Anomica: A Fast Support Vector Based Novelty Detection Technique. In *IEEE Conference of Data Mining*, pages 101–109, 2009.

[11]S. Das, B. Matthews, and R. Lawrence. Fleet Level Anomaly Detection of Aviation Safety Data. In *IEEE Prognostics and Health Management Conference*, 2011.

[12]S. Das, B. L. Matthews, A. N. Srivastava, and N. C. Oza. Multiple kernel learning for heterogeneous anomaly detection: algorithm and aviation safety case study. In *Proceedings of SIGKDD'10*, pages 47–56, 2010.

[13]S. Hido, T. Shohei, K. Yuta, H. Kashima, M. Sugiyama, and T. Kanamori. Inlier-based outlier detection via direct density ratio estimation. In *ICDM '08: Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, pages 223–232, 2008.

[14]V. Hodge and J. Austin. A survey of outlier detection methodologies. *Artificial Intelligence Review*, 22(2):85–126, 2004.

[15]W. Hu, Y. Liao, and V. R. Vemuri. Robust anomaly detection using support vector machines. In *ICMLA '03: Proceedings of the International Conference on Machine Learning and Applications*, pages 168–174, 2003.

[16]H. Kashima, K. Tsuda, and A. Inokuchi. Kernels for graphs. *Kernel Methods in Computational Biology*, 39(1):101–113, 2004.

[17]E. M. Knorr, R. T. Ng, and V. Tucakov. Distance-based outliers: algorithms and applications. *The VLDB Journal*, 8(3-4):237–253, 2000.

[18]G. Lanckriet, N. Cristianini, L. E. Ghaoui, P. Bartlett, and M. Jordan. Learning the kernel matrix with semidefinite programming. *Journal of Machine Learning Research*, 5:27–72, 2004.

[19]J. Lin, E. J. Keogh, L. Wei, and S. Lonardi. Experiencing sax: a novel symbolic representation of time series. *Data Min. Knowl. Discov.*, 15(2):107–144, 2007.

[20]S. Ramaswamy, R. Rastogi, and K. Shim. Efficient algorithms for mining outliers from large data sets. *ACM SIGMOD Record*, 29(2):427–438, 2000.

[21]B. Schölkopf, J. C. Platt, J. C. Shawe-Taylor, A. J. Smola, and R. C. Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.

[22]B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, 2002.

[23]A. N. Srivastava. Greener aviation with virtual sensors: a case study. *Data Min. Knowl. Discov.*, 24(2):443–471, 2012.