An Algorithm for Classification and Outlier Detection of Time-Series Data

R. ANDREW WEEKLEY
National Center for Atmospheric Research, Boulder, Colorado

ROBERT K. GOODRICH
National Center for Atmospheric Research, and Department of Mathematics,
University of Colorado, Boulder, Colorado

LARRY B. CORNMAN
National Center for Atmospheric Research, Boulder, Colorado

(Manuscript received 20 February 2009, in final form 16 July 2009)

ABSTRACT

An algorithm to perform outlier detection on time-series data is developed, the intelligent outlier detection algorithm (IODA). This algorithm treats a time series as an image and segments the image into clusters of interest, such as “nominal data” and “failure mode” clusters. The algorithm uses density clustering techniques to identify sequences of coincident clusters in both the time domain and delay space, where the delay-space representation of the time series consists of ordered pairs of consecutive data points taken from the time series. “Optimal” clusters that contain either mostly nominal or mostly failure-mode data are identified in both the time domain and delay space. A best cluster is selected in delay space and used to construct a “feature” in the time domain from a subset of the optimal time-domain clusters. Segments of the time series and each datum in the time series are classified using decision trees. Depending on the classification of the time series, a final quality score (or quality index) for each data point is calculated by combining a number of individual indicators. The performance of the algorithm is demonstrated via analyses of real and simulated time-series data.

1. Introduction

The analysis of times-series data plays a fundamental role in science and engineering and relies on the identification and classification of various features in the data. Quality control may be viewed as a subclass of problems in general feature identification and classification (e.g., differentiating between a “good” signal and a “contaminated” signal). Existing time-series algorithms detect outliers when the assumptions inherent in the technique are reasonably well satisfied or when the percentage of outliers is relatively small. For example, an algorithm may assume that the data are nearly stationary over a given time scale and/or that a specified model (e.g., a polynomial) is a good local approximation to the data. It may be assumed that the outliers are isolated points, well separated from the true signal, or that there is additive noise with known statistical characteristics (e.g., Gaussian). Likewise, the algorithm developed in this paper, the intelligent outlier detection algorithm (IODA), performs well when the following assumptions are reasonably well satisfied:

1) the time series is highly autocorrelated, and outliers have a different correlation structure than “nominal” data;
2) the sample rate of the data is fast enough to capture the correlation structure and to measure most of the variations in the data; and
3) clusters of nominal data are approximately convex in delay space.

Time-series algorithms, such as autoregressive moving average [ARMA (p, q)], may be used to remove
isolated outliers in stationary data (Box and Jenkins 1970; Priestley 1981). The data are used to compute model coefficients and variance estimates; if a point in question is a large distance from the model prediction in terms of the estimated variance, such a point may be called an outlier. For data containing more than a small number of isolated outliers, it is necessary to use robust techniques to compute the model parameters (Barnett and Lewis 1977). These techniques are necessary, because numerous outliers may cause a large error in the parameter estimates in ARMA-type methods. Robust techniques are much less sensitive to numerous outliers in the data. However, even robust methods have breakdown points. For example, if a running median is applied to the data and more than 50% of the data in a given window are outliers, this robust technique could give incorrect results. Regardless, a quality indicator is still desired in these cases.

Human analysts are adept at feature identification and classification; however, in many applications it is desirable to have an automated algorithm that processes data. The original motivation for developing IODA1 was to provide a data quality index for anemometer data in the difficult environment of Juneau, Alaska (Weekley et al. 2003). In this paper, an algorithm that attempts to mimic the feature classification and identification processing of the human analyst is presented and applied to the quality-control problem for time-series data. This algorithm is based on image processing and cluster analysis techniques and was inspired by image processing techniques applied to Doppler wind profiler data (Cornman et al. 1998). IODA uses both the time domain and delay space for feature detection and classification of a single time series. Here, the time domain is represented by a standard time-series plot of points \( p_i = (t_i, y_i) \), and delay space is represented by a plot of points of the form \( p^*_i(n) = (y_i, y_{i+n}) \), where \( i \) and \( n \) are integers and \( n \) refers to the order of the delay denoted \( \text{delay}(n) \). Here, \( y_i \) is the data at time \( t_i \) (an asterisk is used to denote delay space throughout). Optimal clusters that contain mostly nominal data are found using density and graph techniques. These techniques have been used previously in the literature. Specifically, density maps have been used to remove outliers from points in a plane (i.e., scatter data; Wishart 1969). Density graphs have been constructed from a surface using multiple thresholds for data in a plane (Hartigan 1975). The knowledge discovery and data mining (KDD) community has employed density clustering techniques along with correlation in data analysis.

\( ^1 \) U.S. Patent Number 6735550 issued 11 May 2004. Contact the author for information concerning no-cost license agreements to obtain IODA for research and educational purposes.
FIG. 1. Examples of time-series data from various instruments including anemometers, aircraft, and lidar illustrating nominal data and five different failure modes.
on 17 m s\(^{-1}\) and the dropout data, where a chain of data consists of a subset of consecutive points in the time series.

Figure 1c shows the vertical velocity (in m s\(^{-1}\)) of the wind measured by an aircraft. This “nonstationary” case also has a large number of intermittent dropouts. Specifically, many of the data points up to 600 s are suspect, especially those from 460 to 560 s. This example also illustrates how knowledge of the data source influences conclusions about the data quality. For example, the long stretch of vertical velocities at approximately −10 m s\(^{-1}\) is clearly physically unreasonable. Nevertheless these data appear nominal in the sense that the data are well correlated. Notice that the correlated or dense data are nonstationary over the time period, changing from a mean value of −10 to 0 m s\(^{-1}\) between 570 and 600 s. A human might form a single feature to the right of 600 s, a second feature with the lower band of data from 450 to 600 s, and a third feature containing the less correlated data between 460 and 560 s. The structure in delay space is much more complicated and dramatically different than in the nominal or dropout cases; for instance, there is a faint “cross” of denser data present in delay space centered on roughly (0,0), a cluster centered roughly on (−10,−10), and clusters centered on (−10,5) and (5,−10).

Figure 1d, the “masking” case, is an anemometer time series of 1-s wind direction measurements (clockwise from due north in degrees), where the correct wind direction is ∼40° (determined by a nearby anemometer). Note that the data in the wind direction examples are treated as noncircular throughout the paper. Most of the data are contiguous and well behaved as a function of time, and the outliers are well separated from the nominal data. The band of data around ∼200° is suspicious, because it appears abnormally flat as well as being far from the majority of the data. It is believed that this failure mode is associated with a bad transistor at the anemometer site. In this case, a human might create a single feature centered on ∼40° and ignore the outliers. The majority of the data in the delay-space plot is in a cluster near the origin. The disperse data in the delay-space plot correspond to ordered pairs of points that contain either a single outlier or a pair of outliers.

It has been observed that certain wind frequencies excite normal modes of the anemometer wind direction head and can cause the device to spin erratically. Such an example may be seen in Fig. 1e, the “block” case. Again, time is measured in seconds. Between ∼500 and 950 s, the anemometer is spinning and the data become essentially a random sample of a uniform distribution between ∼50° and 360°. The true wind direction is seen intermittently at ∼225°, which is in general agreement with the value from a nearby anemometer. A human might cluster the data in the intervals from ∼100 to ∼600 s, ∼950 to ∼1000 s and ∼1450 to 1800 s. Formally, the data between ∼600 and ∼950 s and between ∼1000 and ∼1450 s do not form clusters (except for a single small cluster near 1300 s) because of the sparseness of the data. The delay-space plot in this case is dramatically different than most of the previous cases.

In Fig. 1f, the “background” case, radial velocity measurements in meters per second versus time in seconds calculated from Doppler lidar spectra at a fixed range gate are shown. The instrument was operating near its signal-to-noise limit. (Frehlich et al. 1994) The disperse background of points in the time-domain image are suspect measurements of the radial velocity. In this case, a human might cluster the higher-density data centered on 0 m s\(^{-1}\). The delay-space data in this case form a cross, which again is similar to some of the delay-space data in the aircraft plot (Fig. 1c).

2. The failure-mode decision tree and the construction of optimal clusters in time domain and delay space

IODA scores the time-series data as a function of the detected failure mode and assumes that different failure modes have different representations in the time domain and delay space. A decision tree is constructed to classify the example failure modes shown in Fig. 1 by considering the number and location of “optimal” clusters in delay space and properties of the data in relation to the optimal clusters in the time domain. Here, an optimal cluster contains either mostly nominal or mostly suspect data. Finding optimal clusters reduces the complexity of the problem in the sense that the algorithm initially considers clusters of data rather than the data itself. The notion of optimal clusters implies the existence of other less desirable clusters, either too small or too large. A cluster that is “too large” may contain both suspect and nominal data, and a cluster that is “too small” contains nominal data but not all of the nominal data available. The optimal clusters are found in the time domain and delay space using density techniques: specifically, density maps and density contour graphs.

A two-dimensional density map is calculated in each domain. In the time domain, the density map is defined as a probability density function estimated by a normalized histogram over a moving 30-s window. In the delay space, a density map is found by computing the density of points in overlapping tiles within the delay-space plot over the entire time domain. To mitigate edge effects, the tiles were allowed to overrun the edges. These density maps are extended using bilinear interpolation to define
a continuous map $I$ (in the time domain) and $I^*$ (in delay space). The continuous map in the time domain is referred to as the “histogram map.” These maps define a surface value above every point in the time–data plane and in the delay-space–data plane. Clusters are defined as contiguous regions (connected components) with density values larger than a threshold value $T$ in both the time domain and delay space. A set of nested and coincident clusters can be found by using a monotonic decreasing set of threshold values. The multiple thresholds used are the 90th, 80th, 70th . . . 10th percentiles of the continuous map (for an alternate approach, see Chi et al. 1996). Figure 2 shows a subset of the coincident clusters found in the dropout case in both the time (Fig. 2, left) and delay space (Fig. 2, right). As the threshold is lowered, more data are included in the clusters found in each domain. Cluster boundaries are shown in Figs. 2a,b as solid contours and are computed using the MATLAB contour function (the entire algorithm was implemented using MATLAB). The fact that there is a plurality of clusters with a varying degree of quality suggests a score should be created and used to find the optimal clusters.

The skill of the algorithm is a consequence of the processing done once the optimal clusters are determined. Consequently, the processing to find optimal clusters (section 3) will be discussed in greater detail after the decision tree and the construction of time-domain features have been presented. For example, in Fig. 2a, consider the clusters at the lowest threshold level (i.e., the largest clusters). These time-domain clusters centered on roughly 17 m s$^{-1}$ could be combined into a single feature spanning most of the time domain.

Data classification via a decision tree

Figure 3 shows optimal clusters found by IODA in delay space (right panels) and the optimal clusters found in the time domain (left panels) for the example time series. Notice that, in each of the cases (Figs. 3a–f), the clusters found match, on a heuristic level, the discussion presented in section 1a. In the case of Fig. 3a, the optimal time-domain clusters form a single feature that contains most of the time-domain data (with the exception of a few of the gusts) and there is a single optimal cluster found in delay space centered on roughly (17, 17). In Fig. 3b, the time-domain optimal clusters either contain mostly nominal data or mostly suspect data. The nominal data form one feature, and the dropout clusters form a second feature. Two optimal clusters are found in delay space, one centered roughly on (17, 17) and a second near the origin. In Fig. 3c, there is a single time-domain optimal cluster from roughly 560 to 620 s, multiple clusters centered on $-10$ m s$^{-1}$ from 440 to roughly 560 s, and a few small clusters centered on 0 m s$^{-1}$ between 470 and 570 s. Notice that there are two optimal clusters in delay space, one centered on (0, 0) m s$^{-1}$ and a second on $(-10, -10)$ m s$^{-1}$. In Fig. 3d, there is a single time-domain optimal cluster or feature centered on $40^\circ$ that spans nearly the entire dataset, and there are multiple
small clusters centered on roughly 200° that form a second feature. As in the previous case, there are two clusters in delay space; however, only one cluster is intersected by the line $y = x$. In Fig. 3e, there are multiple time-domain optimal clusters near 200° that form a feature separated by regions of nearly uniformly distributed data and only a single cluster in delay space. Finally, in Fig. 3f, there is a single cluster or feature centered on roughly 0 m s$^{-1}$ and a single cluster in delay space near (0, 0).

It is important to note the similarities and differences in the delay-space and time-domain plots in each of these cases. In all cases, there is at least one optimal cluster in delay space. In Fig. 3a, there is exactly one
delay-space optimal cluster. Delay-space plots for cases in Figs. 3b,c are similar in the sense that there are two delay-space clusters on axis (intersected by the line \( y = x \)); however, in Fig. 3b, the time-domain data have multiple dropouts, whereas in Fig. 3c the two delay-space clusters are caused by nonstationary data. In Figs. 3e,f, both plots have a single optimal delay-space cluster and a large number of points outside or external to the optimal delay-space cluster. The outliers in Fig. 3e fall in the gaps between the time-domain clusters, whereas in Fig. 3f the outliers are simply external to the optimal time-domain cluster (a point is external to an optimal cluster in the time domain if the point is above or below the cluster but not in the cluster); in Fig. 3d, there are two optimal clusters in delay space, but only one cluster is on axis.

The previous discussion is the motivation for the failure-mode decision tree, which can be summarized by the following rules:

1) If there is a single cluster in delay space with few external points in delay space, then classify the time series as nominal.
2) If there are multiple on-axis delay-space clusters and the features in the time domain are not connected by a cluster, then classify the data as dropout.
3) If there are multiple on-axis delay-space clusters and the features in the time domain are connected by a cluster, then classify the data as nonstationary.
4) If there is an off-axis cluster in delay space, then classify the data as masking.
5) If there is a single cluster in delay space and there are numerous external points in delay space and few external points in the time domain, then classify the data as “uniform block.” Here, a point in delay space is external if it is outside the lowest threshold contour.
6) If there is a single cluster in delay space and there are numerous external points in both the delay space and time domains, then classify the data as “uniform background.”

### 3. Density contour graph and optimal clusters

Recall that clusters are taken to be contiguous regions where the density map value is above some threshold. Given a set of nested coincident clusters, the idea is to find the thresholds that will produce optimal clusters. One way to sort out which clusters are optimal is to organize the clusters into a density contour graph or tree (Hartigan 1975). This graph has nodes corresponding to a cluster, and two nodes are connected by an arc if one of the corresponding clusters is contained in the other. Each root (a cluster that does not contain any other cluster) in the density contour graph is contained in a unique maximally connected path in the graph (a property of density graphs). This graph (possibly not connected) becomes a tree (connected) when the lowest threshold is zero. The goal is to find the optimal clusters in this graph. For a discussion of trees and graphs, see Luenberger (1984). Methods are developed to score clusters in these graphs, and the graphs are searched to find optimal clusters, first in delay space and then in the time domain.

#### a. Delay-space optimal clusters

The search for optimal clusters begins in delay space, because it is easier to identify nominal delay-space clusters than it is to find nominal time-domain clusters. Specifically, nominal delay-space clusters are convex, should be intersected by the line \( y = x \), and have a high degree of correlation. There may be multiple optimal delay-space clusters \((C^d)^*\), but a single cluster will be selected as the initial nominal data (denoted \( C^d \)). It could be that other delay-space optimal clusters result from nominal data, as in the nonstationary case. However, to resolve nonstationary data, time-domain optimal clusters are needed first and can be found using the initial guess for \( C^d \).

In Fig. 2b, the delay-space clusters for the data form sets that are approximately convex and are consistent with standard techniques such as principal component analysis (PCA). In PCA, for nominal data, one finds approximately elliptically shaped contours in delay space (Bohm et al. 2004; Jolliffe 2002). A slightly simpler approach is to look for convex contours rather than elliptical contours. A cluster is convex if any straight line connecting two points in the cluster is contained in the cluster. Specifically, the convexity of each delay-space cluster is estimated by drawing line segments that connect pairs of points on the cluster’s boundary. Multiple pairs of points are sampled, and the percentage of line segments that remain in the cluster is taken as an estimate of the cluster’s convexity. To ensure separation, a sigmoid is applied to this estimate of the convexity to give a convexity score. A weighted convexity score is calculated as the geometric mean of the convexity score and the percentage of points in the cluster. This score increases as the delay-space clusters grow in size. Once the convexity score begins to become small, because more suspect data are included in the cluster, the weighted convexity score will decrease dramatically because of the sigmoid function applied earlier. For a mathematical discussion of alternative methods for scoring convexity in Euclidean space, see Ban and Gal (2002).

In general, to find an optimal cluster in a density contour graph, a tree in the graph is selected and the root
with the highest weighted convexity score is found. The unique maximal path containing the root is searched to find the cluster with the highest weighted convexity score. This is done for each tree in the density contour graph and several optimal clusters may be found \( (C^\circ) \). For example, in Fig. 2b, there are two trees and two optimal clusters in delay space. The optimal delay-space cluster with the largest \( r^2 \) (the sample correlation coefficient) is selected to represent the nominal data or \( C^\circ_0 \).

Using the data inside the optimal delay-space cluster \( C^\circ_0 \) to estimate a sample deviation, a score that is a measure of distance from the line \( y = x \) in delay space is calculated for every point in delay space. This score, the \( C^\circ_0 \) N-sigma score, is given by

\[
s^\circ(y_i, y_{i+1}) = \frac{2}{1 + e^{k(y_{i+1} - y_i)/\sigma}},
\]

where \( k > 0 \) and sigma is the sample deviation (the square root of the sample variance) of the data in the cluster \( C^\circ_0 \). The \( C^\circ_0 \) N-sigma score is then translated to the time domain and used to find the optimal clusters in the time domain. Note that the \( C^\circ_0 \) N-sigma score is for an ordered pair of points in delay space. Consequently, a \( C^\circ_0 \) N-sigma score \( s^\circ \) is calculated for both \( (y_{i-1}, y_i) \) and \( (y_i, y_{i+1}) \). To find the optimal clusters in the time domain, the delay-space \( C^\circ_0 \) N-sigma score is translated to the time domain for a point \( p_i = (t_i, y_i) \) by

\[
s(p_i) = \sqrt{s^\circ(y_{i-1}, y_i) \cdot s^\circ(y_i, y_{i+1})}.
\]

b. Optimal clusters in the time domain and the time-domain feature

The methodology to find the optimal clusters in the time domain is slightly different than in delay space. In the time domain, an arc score is defined, where an arc connects two nodes in a graph (the clusters are the nodes in the graph). Physically, the arc between two coincident clusters is the set of points inside one cluster and not inside the coincident cluster. The idea is to start with a root that has enough well-correlated points (i.e., the median of the time domain \( C^\circ_0 \) N-sigma score is above a threshold). The cluster is allowed to grow as long as the points added to the cluster are well correlated. Once the correlation drops below a threshold value, the optimal cluster is taken as the previous cluster in the graph.

In IODA, a time-domain feature is a set of optimal time-domain clusters that share similar global statistics, as defined by an optimal delay-space cluster. Specifically, given an optimal delay-space cluster \( C^\circ \), define the associated time-domain feature \( f(C^\circ) \) to be the set of optimal clusters in the time domain such that there exist points \( (t_i, y_i) \) in \( C \) (an optimal time-domain cluster) such that \( (y_i, y_{i+1}) \) is in the cluster \( C^\circ \). See the discussion of Fig. 2 in section 2. In particular, the initial time-domain feature that is nominal is defined as \( f(C^\circ_0) \). Furthermore, let \( Y(C^\circ) \) denote the set of points \( (t_i, y_i) \) that belong to an optimal time-domain cluster in \( f(C^\circ) \). For example, in Fig. 3b, each optimal time-domain cluster containing 17 m s\(^{-1}\) has points \( y_i \) with \( (y_i, y_{i+1}) \) in \( C^\circ_0 \), the optimal cluster centered at \((17, 17)\) m s\(^{-1}\). Thus, \( f(C^\circ_0) \) is the set of these clusters and \( Y(C^\circ_0) \) is the set of all points interior to these clusters.

4. Combinations of failure modes

A time series may contain more than a single failure mode. Consequently, combinations of failure modes should be considered when classifying a time series. In the current implementation of the algorithm, 128 different combinations of failure modes are allowed. The total number of combinations depends on the types of failure modes currently found. From the decision-tree method outlined in the subsection of section 2, there are essentially five failure modes and they can be summarized as external outliers, masking, nonstationary, uniform outliers, and gap outliers. A flag is set to indicate if any one of these types of outliers is present in the data. Also, the number of clusters (at most, four: two on-axis and two off-axis clusters) in delay space can be taken as another free parameter giving an upper bound of at most \( 2^5 \times 4 = 128 \) combinations of failure modes. For the purpose of specifying a lookup table, a time series is assigned a number that corresponds to one of the 128 combinations of failure modes and the data are scored depending on the failure modes found. Currently, there are 12 distinct combinations of failure modes in IODA corresponding to the actual failure modes found in the example data shown if Fig. 1.

5. Point classification and reclassification

Once the time series has been characterized by a combination of failure modes and the optimal time-domain clusters are found and combined into nominal features and failure-mode features, it is possible to classify individual data points using a second decision tree. The point classification depends on the relationship of a point to optimal time-domain clusters and to nominal features and failure-mode features. For example, there are points inside a feature \( f(C^\circ) \), points inside optimal time-domain clusters, and chains of points that leave a feature and either return to the feature or enter another
feature. The assignment of a point type is accomplished by considering points individually, as chains, or by comparing a point to both local and global distributions of points of a known type, such as “points that belong to $f(C_i)$.” The initial classification creates a starting point for further classifications. It simply categorizes points according to combinations of important known properties or indicators, such as “a point belongs to an optimal cluster.” A number is assigned to different combinations of indicators as a convenient way to summarize the allowed combinations of point characteristics. Once the data points have been assigned a classification or a point type (29 in total), further processing checks these classifications and in some cases performs a reclassification.

The reclassification of points refines the initial point type using larger-scale information, such as the behavior of chains. For instance, consider the dropout case where a sequence of points may leave the feature $f(C_i)$, enter another feature $f(C*)$, and return to $f(C_i)$ (Fig. 3b). These points are all reclassified as elements of “chains that go through another feature.” Points can also be reclassified using local and global distributions of points that have a specific type. For instance, in the case of masking, a point may be assigned a new type according to which global distribution it most likely belongs to [i.e., the distribution of points from $Y(C_{0i})$ or the distribution of points not in $Y(C_{0i})$]. A similar comparison is done locally. In this case, rather than using all of $Y(C_{0i})$, points can be drawn locally from $Y(C_{0i})$ using a window. The probability that a test point belongs to a cluster can then be estimated using these local distributions and a new point type can be reassigned accordingly. It is possible that the classification of a point does not change. Reclassification occurs only when an appropriate test is satisfied and is failure-mode dependent. Some reclassification tests are applied only for failure modes that produce data that might correctly be reclassified using these types of tests, such as in the block case where points are classified according to their estimated probability of belonging to the feature or to the external gap points. The exact details are omitted, because there are many cases to be considered and in practice only a few points are typically reclassified.

The detected failure mode simply indicates that there are bad data present in the time series but does not determine the exact location of those bad data. Assignment of a point type is an attempt to locate the bad data in the time domain. The methods used to categorize points are the same, regardless of the failure mode; however, the assignment of a score to a point type can change as a function of failure mode and is accomplished using a lookup table, which depends on the failure mode and point type. Therefore, points of the same type may have different scores, depending on the detected failure mode.

6. Scoring points

Next, a final score is defined for each point by combining multiple scores. Recall that the $C_{0i}$ N-sigma score is a number that varies between 0 and 1 and produces a measure of how close a point is to the line $y = x$ in delay space. Another such score is the histogram map score, which is the height of the histogram map above a point. These two scores are combined using a weighted sum into a “preliminary score” by

$$q(t) = \frac{\sum_{j=1}^{2} G[s_i(t_j), a, c]w_j}{\sum_{j=1}^{2} w_j},$$  \hspace{1cm} (3)$$

where

$$G(s_j, a, c) = \frac{1}{(1 + e^{-a(s_j-c)})}$$ \hspace{1cm} (4)$$

is a weighting function for each score $s_j$ (where $s_1$ is the $C_{0i}$ N-sigma score and $s_2$ is the histogram map score), $a > 0$, $c > 0$, $w_j$ is between 0 and 1, and $t_j$ represents time. Here, $w_j$ is the relative weight of one score to another. By changing $w_j$ for a score, the influence of an individual score on the preliminary score is controlled. The weighted-sum formulation allows the inclusion of other scores beyond those indicated previously.

Once the points in the feature have been found, the feature Z-statistic score is calculated by

$$z(t) = \text{erfc}\left(\frac{y(t) - \bar{y}}{3\sigma}\right).$$ \hspace{1cm} (5)$$

Given a window of $n$ points in the time series, centered on $y(t_i)$, $\bar{y}$ and $\sigma$ are calculated from the feature points in the window. If there are too few feature points in the window, then the last value of $\bar{y}$ and $\sigma$ is used. The erfc is the complementary error function given by

$$\text{erfc}(x) = \frac{2}{\sqrt{\pi}} \int_{x}^{\infty} e^{-t^2} dt.$$ \hspace{1cm} (6)$$

Once the failure mode and point types have been determined, a lookup score $l(t)$ can be found as a function of the failure mode and the point type. The lookup score is defined to be either 0 or 1 and is specified a priori.
in a configuration file. The final quality score is calculated as

$$ q_f(t_i) = \max[\sqrt{q(t_i) \cdot z(t_i) \cdot l(t_i) \cdot s_f}]. \quad (7) $$

The feature score $s_f$ is 1 for feature points and 0 otherwise. The details of the lookup table are omitted, because the number of cases is large. Most of the assignments of 0 or 1 are obvious, such as assigning a score of 0 to points in a dropout feature or a score of 1 to points in the initial feature. If it is not obvious, then a score of 1 is assigned.

Because of the Boolean nature of the $l(t)$, $q_f(t)$ tends to be somewhat Boolean.

7. Results for real data

Figure 4 illustrates the final quality scores calculated by IODA for the cases shown in Fig. 1. Cool colors indicate a high final quality score, and warm colors indicate a low final quality score. On a gross scale, the final quality score assigned to the data correspond to what a human expert might give the points (i.e., most of the suspect points are red and most of the good points are blue).

In the nominal case, most of the data has a high quality score, except for a few points that are well separated from the bulk of the data. However, a few of the peaks (gusts) were given a lower final quality score and are candidate cases to consider for future improvements to the algorithm. These points typically have a lower density than their neighbors and hence do not reside in time-domain optimal clusters. In the dropout case, the data near 17 m s$^{-1}$ are given a high final quality score, whereas the excursions to 0 m s$^{-1}$ are given a low final quality score. In the dropout case, $C^n_0$ contains fewer points than in the nominal case and has a smaller standard deviation, hence a larger z statistic, which reduces the $C^n_0$ N-sigma score and in turn gives lower scores to some of the peaks.

In the nonstationary case, the disperse data between 460 and 560 s are assigned a low final quality score, whereas the autocorrelated data are given a high final quality score. Again, the red points intermixed with the blue data (e.g., between 520 and 530 s) indicate possible improvements to the algorithm are needed. These intermixed low final quality score points result from low density caused by the numerous outliers in this time interval. For the masking case, almost all of the suspect points are given a low final quality score, and again the data a human might consider good are given a high final quality score. Notice, however, the small bands of good points with low final quality score data at the beginning and end of the time series. The low final quality scores for these points are possible artifacts of edge effects from the image processing and indicate the need for further work. The final quality scores in the block case and the background case are consistent with how a human might score the data: the blocks of outliers have been assigned a low final quality score, and the autocorrelated data have been assigned a high final quality score. These examples show that the scoring strategies from the decision tree and the lookup tables do give appropriate final quality scores for most of the data points.

Data simulation

Simulated data were created by selecting an hour of nominal real data from an anemometer (specifically, the data shown in Fig. 1a were used) at a time when the outputs of two collocated devices were available and agreed with each other. To create a desired failure mode, points were removed at random from the nominal dataset and replaced with pseudo random numbers generated by a computer algorithm. Two failure modes were simulated: dropout outliers (similar to Fig. 1b) and data mixed with uniformly distributed outliers (similar to Fig. 1f). For the simulated dropout case, data were replaced by a pseudo-Gaussian random variable with mean 1 and standard deviation 0.2. The uniform case data were replaced with values from a uniform distribution on [0, 30]. A fixed percentage of outlier data was introduced into each time series. A total of 13 outlier percentile levels were created, ranging from 5% to 65% of the data in 5% increments, and 100 realizations at each outlier percent level were processed. Two additional scoring algorithms are defined using AR(1) and the running median and are compared to the final quality score found using IODA. The quality score for AR(1) is calculated by

$$ s_{AR} = \text{erfc} \left( \frac{y - y'}{\sigma} \right), \quad (8) $$

where $y'$ is the AR(1) estimate [i.e., ARMA(1, 0)], $y$ is the measured value, $\sigma$ is the standard deviation of $y$ over a running window, and erfc is the complimentary error function. For the running median, the quality score was calculated using a running window of $n$ points and

$$ s_M = \text{erfc} \left( \frac{y - y'}{r_1} \right), \quad (9) $$

where $r_1$ is trimmed range of $y$ (i.e., the 90th percentile of $y$ over $n$ points minus the 10th percentile of $y$ over $n$ points) and $y'$ is the median over $n$ points.

There are many ways to study the skill of an algorithm (e.g., as a function of a quality score threshold). Such
FIG. 4. Examples of time-series data with quality score assigned by IODA. A cool color indicates a high quality score and a warm color indicates a low quality score.
analysis might reveal a quality score threshold to maximize the skill of an algorithm. Specifically, a threshold was set and the points with a quality score greater than the threshold were considered to be positive (true) and the remaining values were considered to be negative (false) for each noise simulation. The true positive rate (TPR; also known as the hit rate) and false positive rate (FPR; also known as the false alarm rate) were calculated as a function of quality score threshold for all three algorithms; the running median, AR [i.e., AR(1)], and IODA at each noise level. The Peirce skill score \( PSS \) was calculated by

\[
PSS = \frac{TPR}{C_0} - FPR
\]

(Wilks 2006). The skill-score graphs are shown in Fig. 5 as a function of the quality score thresholds for all three algorithms.

\(2\) This score is also called the Hessen–Kuiper discriminate, Kuiper’s performance index, and the true skill statistic (TSS).
algorithms, the two failure modes, and all noise levels. Warm colors represent higher noise percentages in the simulations. In Fig. 5a, the PSS for the median algorithm applied to the simulated uniform noise case varies with noise level. Notice that, for a high noise level (red curve), the PSS is at a maximum to the right of 0.8, whereas for low noise the skill is at maximum at lower thresholds between 0.2 and 0.4. In Fig. 5b (the dropout case), the skill for the median is quite low at the higher noise levels (negative). In this case, there is no way to set a single threshold to give reasonable average performance (above 0.5) over all noise levels. Figures 5c,d illustrate the performance of AR applied to simulated uniform and dropout noise. AR does not perform as well as the median algorithm when applied to simulated uniform noise, and it outperforms the median in some of the dropout noise cases. AR has the interesting property that the maximum skill performance occurs at around 0.2 at each noise level in Fig. 5c. The average skill over all noise levels at this threshold is around 0.5. In Fig. 5d, there again is no single threshold that gives a reasonable average skill score over all noise levels. Figures 5e,f illustrate the skill scores for IODA applied to simulated uniform and dropout noise. In the case of simulated uniform noise, IODA outperforms the median in some of the lower noise simulations, has comparable performance in some of the moderate cases, and does not perform as well in some of the higher-percentage noise cases. If the thresholds are set at 0.7, 0.2, and 0.2, (to optimize average skill over the noise cases) in Figs. 5a,c,f respectively, IODA will slightly outperform the median. The median and IODA algorithms are comparable in this case, and both are superior to AR. In the case of dropout noise, IODA outperforms both AR and the median up to 60% noise; IODA eventually breaks down at 65% noise. At this level of noise, IODA selects the noise as the signal. For average skill over all noise levels, IODA outperforms AR and the median. Note that the PSS curve is relatively flat in Figs. 5e,f: this is because the IODA quality score is somewhat Boolean (see the discussion at the end of section 6 and Fig. 4). If selecting a threshold is desired, this insensitivity to the threshold means any reasonable threshold (e.g., 0.2) gives reasonable performance at each noise level (except for noise level of 65% in Fig. 5f). On the other hand, in Fig. 5a, if one knew the percentage of outliers and the failure mode, a high-performance threshold could be selected for each noise level and would require an adaptive version of the running median. This illustrates the importance of classifying the nature and noise level of the failure mode and was a major motivation for developing IODA.

8. Conclusions

The use of image processing and clustering techniques to classify and find outliers in time-series data has been studied. Although IODA has been applied to a small selection of sensors, these techniques should apply to other time-series datasets as well, particularly if the nominal data have a higher autocorrelation in comparison to points associated with contaminants. More work should be done to address the case of rapidly changing and nonstationary data or highly oscillatory data, although these cases may violate assumption 2 (see introduction) for IODA. Under the assumptions of the algorithm, optimal clusters identify most of the nominal data in both delay space and time domain. The number and location (i.e., on axis or off axis) of the clusters in delay space, along with the clusters in the time domain, can be used to characterize the failure-mode process. The failure modes studied have distinct representations that can be codified and uniquely identified using information from the delay space and time domains. In most cases, IODA correctly identifies the failure mode, correctly classifies the points, and returns reasonable quality scores for the time-series points. Figure 4 illustrates that the scores assigned by IODA are similar to how a human might score failure modes such as masking (Fig. 4d), dropouts (Fig. 4b), and nonstationary data (Fig. 4c). Additionally, the simulations demonstrate cases where IODA outperforms or has comparable performance to several commonly used quality-control algorithms (Figs. 5).

Many improvements could be made to the current algorithm. The final quality score calculations could be improved, additional tests could be added, a better search algorithm could be used to find the optimal clusters in the density contour graphs, and the algorithms to determine if a time series is nonstationary could be improved. Additional testing could be applied to further classify a given cluster as nominal. More example failure modes need to be studied, such as innovative outliers and cases where two competing signals cross (in the time domain). It is clear that these techniques could be generalized to study two- and higher-dimensional images. In fact, the authors have applied these ideas with some success to satellite images as well as images from a scanning lidar.

The current implementation of IODA requires extensive calculations and might not be appropriate for some applications (e.g., real-time processing). Even though there are cases where the assumptions of standard algorithms are violated, these standard algorithms do have the advantage of simplicity. On the other hand, IODA might prove useful in cases where the data quality is extremely poor and satisfies the three assumptions outlined.
in the introduction: the correct assessment of the data is critical, the data volume is too large for a human to review, or the consequences of a sensor failure might be significant.

Acknowledgments. This work was funded by opportunity funds provided by the Research Applications Laboratory of the National Center for Atmospheric Research. We would also like to acknowledge Carol Park and Carol Makowski for their editorial assistance, Barb Brown and Marcia Politovich for their technical review and comments, and Rod Frehlich for the LIDAR data set. Also we thank a reviewer for detailed comments and the suggestion to study the calibrations for the median and AR algorithms. We would also like to thank Jeff Cole for sharing video footage of a spinning anemometer wind direction head.

REFERENCES