Environmental Modelling & Software 25 (2010) 1014–1022

Contents lists available at ScienceDirect

Environmental Modelling & Software

journal homepage: www.elsevier.com/locate/envsoft

Anomaly detection in streaming environmental sensor data: A data-driven modeling approach

David J. Hill [a,*], Barbara S. Minsker [b]

[a] Department of Civil and Environmental Engineering, Rutgers University, 623 Bowser Rd, Piscataway, NJ 08854, USA
[b] Department of Civil and Environmental Engineering, University of Illinois Urbana-Champaign, 205 N. Mathews Ave., Urbana, IL 61801, USA

ARTICLE INFO

Article history:
Received 9 March 2009
Received in revised form
25 August 2009
Accepted 25 August 2009
Available online 24 October 2009

Keywords:
Coastal environment
Data-driven modeling
Anomaly detection
Machine learning
Real-time data
Sensor networks
Data quality control
Artificial intelligence

ABSTRACT

The deployment of environmental sensors has generated an interest in real-time applications of the data they collect. This research develops a real-time anomaly detection method for environmental data streams that can be used to identify data that deviate from historical patterns. The method is based on an autoregressive data-driven model of the data stream and its corresponding prediction interval. It performs fast, incremental evaluation of data as it becomes available, scales to large quantities of data, and requires no pre-classification of anomalies. Furthermore, this method can be easily deployed on a large heterogeneous sensor network. Sixteen instantiations of this method are compared based on their ability to identify measurement errors in a windspeed data stream from Corpus Christi, Texas. The results indicate that a multilayer perceptron model of the data stream, coupled with replacement of anomalous data points, performs well at identifying erroneous data in this data stream.

1. Introduction

In-situ environmental sensors are sensors that are physically located in the environment they are monitoring. Through telemetry, the time-series data collected by these sensors can be transmitted continuously to a repository as a data stream. Recently, there have been efforts to make use of streaming data for real-time applications (e.g., Bonner et al., 2002). For example, draft plans for the Water and Environmental Research Systems (WATERS) Network, a proposed national environmental observatory network, have identified real-time analysis and modeling as a significant priority (NRC 2006).

Because *in-situ* sensors operate under harsh conditions, and because the data they collect must be transmitted across communication networks, the data can easily become corrupted. Undetected errors can significantly affect the data's value for real-time applications. Thus, the NSF (National Science Foundation), 2005 has indicated a need for automated data quality assurance and control (QA/QC). Anomaly detection is the process of identifying data that deviate markedly from historical patterns (Hodge and Austin, 2004). Anomalous data can be caused by sensor or data transmission errors or by infrequent system behaviors that are often of interest to scientific and regulatory communities. In addition to data QA/QC, where data anomalies may be the result of sensor or telemetry errors, anomaly detection has many other practical applications, such as adaptive monitoring, where anomalous data indicate phenomena that researchers may wish to investigate further through increased sampling, and anomalous event detection, where anomalous data signal system behaviors that require other actions to be taken, for example in the case of a natural disaster. These applications require that data anomalies be identified in near-real time; thus, the anomaly detection method must be rapid and be performed incrementally to ensure that detection keeps up with the rate of data collection.

Traditionally, anomaly detection has been carried out manually with the assistance of data visualization tools (Mourad and Bertrand-Krajewski, 2002), but manual methods are unsuitable for real-time detection in streaming data, since they necessitate an operator to be performing analysis 24 h a day, 7 days a week. More recently, researchers have suggested automated statistical and machine learning approaches, such as minimum volume ellipsoid (Rousseeuw and Leroy, 1996), convex pealing (Rousseeuw and Leroy, 1996), nearest neighbor (Tang et al., 2002; Ramaswamy et al., 2000), clustering (Bolton and Hand, 2001), neural network classifier

* Corresponding author. Tel.: +1 217 714 3490.
E-mail addresses: ecodavid@rci.rutgers.edu (D.J. Hill), minsker@illinois.edu (B.S. Minsker).

1364-8152/$ – see front matter © 2009 Published by Elsevier Ltd.
doi:10.1016/j.envsoft.2009.08.010

(Kozuma et al., 1994), support vector machine classifier (Bulut et al., 2005), and decision tree (John, 1995). These methods are faster than manual methods, but they have drawbacks that make them unsuitable for real-time anomaly detection in streaming data. Minimum volume ellipsoid and convex peeling require all of the data to have accumulated before anomalies can be identified. Nearest neighbor, clustering and support vector machines are computationally intractable for large quantities of data and neural network classifier, support vector machine classifier and decision tree require pre-classified (anomalous/non-anomalous) data, which characterize all anomalies that may be encountered. Since real-time sensors collect data continuously, and the data are to be used in real time, decisions based on the totality of the data cannot be made. Instead, anomaly classifications must be made based on the data that have been collected up to the current point in time.

Several researchers have suggested anomaly detection methods specifically designed for real-time detection in streaming data. These methods are often referred to as analytical redundancy methods because they employ a model of the sensor data stream as a simulated redundant sensor whose measurements can be compared with those of the actual sensor. The classification of a measurement as anomalous is based on the difference between the model prediction and the sensor measurement. Early work on such methods (e.g., Upadhyaya et al., 1990; Belle et al., 1983) employed multivariate autoregressive (MAR) models to predict the next measurement in the sensor data stream, using historical values. More recently, other regression approaches, such as artificial neural networks (ANNs), have been suggested (Nairac et al., 1999; Fantoni and Mazzola, 1996; Silvestri et al., 1994). These methods were designed for use in manufacturing/power plants, and the only researchers to have suggested a method of threshold selection (Belle et al., 1983) for classifying data as anomalous/non-anomalous required detailed process knowledge that is not generally available for natural systems. Krajewski and Krajewski (1989) present an analytical redundancy method for streamflow data that employs model error standard deviations to set the threshold. This method, however, relies on a physically-based real-time model of the natural system – a tool which may not always be readily available.

This study develops a real-time anomaly detection method that employs a data-driven univariate autoregressive model of the data stream and a prediction interval (PI) calculated from recent historical data to identify streaming data anomalies. The method used to calculate the PI in this study accounts for uncertainty in the data and in the parameters of the data-driven model. A data-driven time-series model is employed because it is simpler to develop than a physics-based time-series model and it can rapidly produce accurate short forecast horizon predictions. Data are classified as anomalous/non-anomalous based on whether or not they fall outside a given PI. Thus, the method provides a principled framework for selecting a threshold. This method does not require any pre-classified examples of data, scales well to large volumes of data, and allows for fast incremental evaluation of data as it becomes available. The Section 2 describes this method in detail. Next, it is tested through a case study, in which several types of data-driven models are used to identify erroneous measurements in a windspeed data stream from the WATERS Network Corpus Christi Bay testbed, provided by the Shoreline Environmental Research Facility (SERF) (http://www.serf.tamus.edu). Finally, the results of the different instantiations of the anomaly detection method are compared, and implications of the different modeling strategies are discussed.

2. Methods

This study proposes a new analytical redundancy method for anomaly detection that uses a moving window of q sensor measurements (or their expected values as explained shortly) $D^t = \{x_{t-q+1}, \ldots, x_t\}$ to classify the next chronologically sequential sensor measurement x_{t+1}. A measurement is classified as anomalous if it deviates significantly from the one-step-ahead prediction of its value calculated using D^t as input. Upon initialization, the method fills the window with the q most recent sensor measurements and commences classification with the next measurement taken by the sensor.

In brief, the method consists of the following steps beginning at time t: (1) use a one-step-ahead prediction model that takes D^t as input to predict $\bar{x}_{t+1}$, the expected value of the sensor measurement at time $t+1$; (2) calculate the upper and lower bounds of the range within which the sensor measurement should lie (i.e., the prediction interval) with probability p; (3) when the measurement at time $t+1$ arrives from the sensor, compare the sensor measurement with this range, and if it falls outside the range, classify it as anomalous, otherwise classify it is non-anomalous; (4a) under the anomaly detection and mitigation (ADAM) strategy, if the measurement is classified as anomalous modify D^t by removing x_{t-q+1} from the back of the window and adding $\bar{x}_{t+1}$ to the front of the window to create D^{t+1}; (4b) under the anomaly detection (AD) only strategy, modify D^t by removing x_{t-q+1} from the back of the window and adding x_{t+1} to the front of the window to create D^{t+1}; (4) repeat steps 1–4. This process is illustrated with a flow chart in Fig. 1. The remainder of this section describes these steps in detail.

2.1. Step 1: One-step-ahead prediction

In the first step of the anomaly detection process, a univariate autoregressive model of the sensor data stream is used to predict the next measurement in the data stream. Univariate autoregressive models are models that predict future measurements in a sensor data stream using only a specified set of previous measurements from the same sensor; they are used because they simplify the anomaly detection process in several ways. First, using only previous values of the same data stream avoids complications caused by different sampling frequencies that can arise if a heterogeneous set of sensor data streams were used. Second, because of the frequency with which embedded environmental sensors go offline, a significant number of gaps of undefined duration exist within each of the sensor data streams, and when comparing the streams, these gaps usually do not correspond with the same time periods. Since time-series models require a defined set of input variables, it is unclear how to make predictions if one or more of the input variables is not available; thus, the use of autoregressive models reduces the number of predictions that cannot be made due to insufficient data. Finally, since anomalous data cannot be expected to produce reasonable predictions when used as inputs into a model, if data from more than one sensor are used for prediction of a particular data stream, then anomalous data from different sensors must be detected in a particular order, such that all of the independent variables of a model have already been processed. For example, if the model for data stream A requires the most recent measurements from data stream B, then anomaly detection must first be performed on data stream B, before it can be performed on data stream A. The use of autoregressive models allows anomaly detection of several sensor data streams to take place rapidly and in an arbitrary order.

The moving window $D^t = \{x_{t-q+1}, \ldots, x_t\}$ is used as input to the autoregressive model of the data stream to predict the next (i.e., one-step-ahead) sensor measurement:

$$\bar{x}_{t+1} = M(D^t) + \varepsilon \tag{1}$$

where $M()$ is the model and ε is a random variable representing the model error. This method assumes that the behavior of the

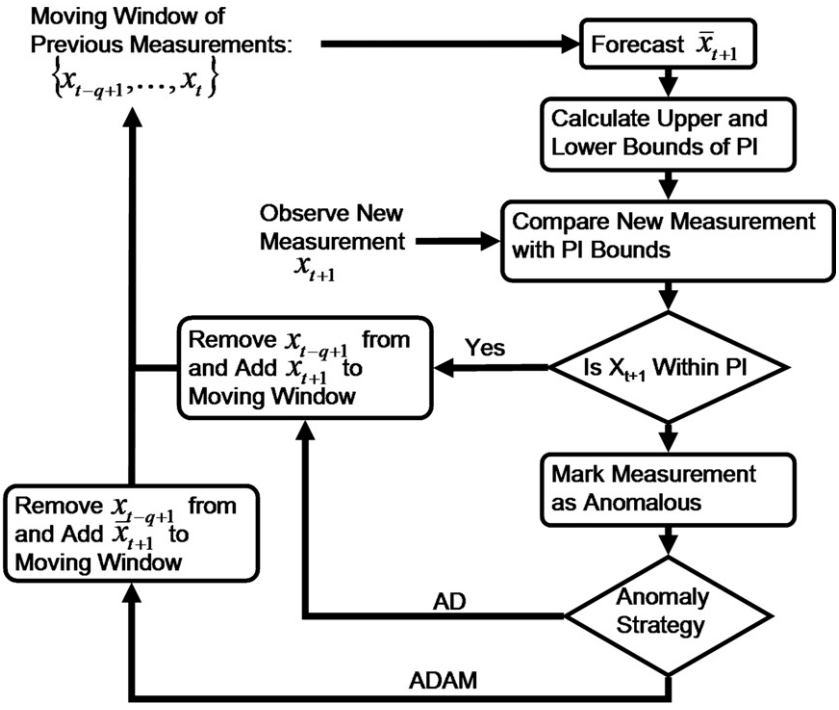

Fig. 1. Schematic of proposed anomaly detection method.

processes at time $t+1$ can be described by a finite set of q previous measurements, thus it implicitly assumes that the data stream is an order q Markov process. This study compares four data-driven methods for creating the one-step-ahead prediction model used in Eq. (1): naïve, nearest cluster (NC), single-layer linear network (LN), and multilayer perceptron (MLP). It should be noted that the MLP in this study performs time-series regression, rather than classification, and it is thus fundamentally different from the MLP classifier-based methods discussed in the introduction. These modeling methods were selected because they have all been demonstrated to successfully model different types of time-series data and are related to traditional statistical time-series modeling, as indicated below.

These types of data-driven methods use sets of training examples to tune the model parameters. Each example contains an input vector (i.e., the set of variables used to make a prediction, in this case D^t) and an output vector (i.e., the desired model prediction, in this case $\bar{x}_{t+1}$). Training involves fitting the model parameters to minimize error on the training examples, without overfitting to the training data that can lead to poor predictions. The PI requires an estimate of the standard deviation of the modeling error, for which 10-fold cross-validation (Duda et al., 2001; Han and Kamber, 2006) is used because it has been shown to quantify error caused by uncertainty in the data measurement and in the model parameter estimation (Han and Kamber, 2006). In 10-fold cross-validation, the training set is divided into 10 non-intersecting subsets of equal size, chosen by random sampling. The model is then trained 10 times, each time reserving one of the subsets as a validation set on which to evaluate the model error while fitting the model parameters using the remaining nine subsets. The model parameters with the lowest mean squared error among the 10 training models are then selected for the final model.

The error mean and standard deviation of the selected model are then calculated as the average mean and standard deviation of the prediction errors from each of the 10 trainings. Thus, the distribution of the final model errors is specified as:

$$\mu_r = \frac{1}{N}\sum_{i=1}^{10}\left(\sum_{j=1}^{N/10} r_j^i\right)$$

$$\sigma_r = \left[\frac{1}{N-1}\sum_{i=1}^{10}\left(\sum_{j=1}^{N/10}\left(r_j^i-\mu_r^i\right)^2\right)\right]^{1/2} \quad (2)$$

where μ_r and σ_r are the mean and standard deviation of the final model, N is the total number of training examples, r_j^i is the model residual on the jth training example during the ith training, and μ_r^i is the residual mean during the ith training.

The 10-fold cross-validation approach described above is used in lieu of standard single-fold cross-validation because it captures training errors that can occur with stochastic training algorithms. For example, the MLP training procedure uses a gradient-based method that may result in different model parameters depending on where in the parameter space the method was started because it may find a local optimum.

The remainder of this section describes the implementation of the four data-driven modeling methods used in this study and indicates their suitability for modeling time-series data.

2.1.1. Naïve predictor

The naïve predictor is a popular benchmark for evaluating time-series forecasting (Gjølberg and Bengtsson, 1998), which takes the form of a random walk of the original data (i.e., from one time step to the next, the original time series merely takes a random jump from its last recorded position). Mathematically, this model can be expressed as:

$$\bar{x}_{t+1} = x_t \quad (3)$$

Since this model has no parameters, neither a training procedure nor 10-fold cross-validation is necessary to estimate the distribution of the error term ε in Eq. (1); instead, it can be calculated

directly from the full training set as the mean and variance of $(x_{t+1} - x_t)$. This model does not require that the time series is stationary, but does make the assumption that the time series represents a first-order Markov process, since the measurement at time $t+1$ depends only on the measurement at time t. Not only is this model equivalent to a type of autoregressive integrated moving average (ARIMA) model, specifically the ARIMA(0,1,0) (see Box and Jenkins, 1970), but it also can be viewed as a type of nearest-neighbor (NN) algorithm (Duda et al., 2001; Hastie et al., 2001) that bases its prediction of an unseen event on the response of the system to the most similar historical event and defines similarity in terms of temporal distance. Because this nearest-neighbor approach uses temporal proximity, and because time-series data are inherently chronologically ordered, this approach, unlike many nearest-neighbor approaches, scales well to large quantities of data. This model has found wide use in economic time-series prediction (e.g., Evans, 2002; Thomakos and Guerard, 2004) and Amenu et al., 2007 demonstrated the utility of this method for modeling stream flow time-series data.

2.1.2. Nearest cluster (NC) predictor

The NC predictor is a modification of the k-nearest-neighbor (kNN) method in which the predicted value of a sensor measurement is calculated as the average of the k most similar sensor measurements in the training data, where measurement similarity is based on the Euclidean distance between the moving windows associated with each measurement (the use of k as the number of nearest neighbors is for consistency with the literature and is not related to the variable k in the k-means algorithm described shortly). This type of model has been shown to represent a wide variety of stationary and non-stationary time series with both linear and non-linear dynamics (Illa et al., 2004). Since finding the nearest neighbors requires calculating the Euclidean distance between the unseen sensor measurement that is being predicted and each training example, the time complexity of this model increases linearly with the number of training data. For this reason, an approximation of the NN method is often used, where clustering is used to group similar training data into k_c clusters based on their moving windows and the Euclidean distance metric (as in the NN described above). The predicted value of an unseen sensor measurement is then calculated as the average of all the sensor measurements in the cluster that the unseen measurement maps into. Bannayan and Hoogenboom (2008) demonstrated a similar NN approximation for predicting daily meteorological time series (including radiation, precipitation, and temperature). If the popular k-means (Duda et al., 2001; Hastie et al., 2001; Han and Kamber, 2006) clustering algorithm is used to cluster the training data, this approximation reduces the computational complexity of the prediction task to be linear in the number of clusters (Schütze and Silverstein, 1997), which is often a significant saving. Thus, given the moving window D^t of a yet unseen measurement x_{t+1}, the NC method finds the cluster that D^t falls into and predicts the value of the measurement $\bar{x}_{t+1}$ to be the average of all the sensor measurements in the training set that also fall in that cluster. In this study, the training data are partitioned using the k-means clustering algorithm, which requires that the number of clusters be specified by the user. This parameter can be inferred from the training data by performing the k-means algorithm iteratively. For each iteration, the number of clusters is increased and the within-cluster scatter (Hastie et al., 2001), which indicates the similarity of the points to their assigned cluster center, is evaluated to determine when no further clusters are necessary. This method of determining the number of clusters is costly, but since it only has to be performed once, it does not significantly affect the utility of the clustering predictor for real-time applications. The k-means partitioning algorithm is stochastic (i.e., cluster centers may depend on initial guesses of the cluster centers), hence the distribution of the model error ε in Eq. (1) is determined using 10-fold cross-validation.

2.1.3. Single-layer linear network (LN) predictor

The LN approach predicts the sensor measurement at time $t+1$ as a linear combination of the q previous measurements:

$$\bar{x}_{t+1} = b + \sum_{i=0}^{q-1} w_i x_{t-i} \tag{4}$$

where b and $\{w_0, \ldots, w_{q-2}\}$ are the constant and set of weights, respectively, that define the relationship between the moving window $\{x_{t-q}, \ldots, x_t\}$ and the expected value of the next sensor measurement $\bar{x}_{t+1}$. The weight vector $\vec{w}$ is learned iteratively by applying a small correction to each element w_i in the weight vector proportional to both the model error on an input set as well as the product $w_i x_{t-i}$, where the constant of proportionality is called the learning rate and the constant b is learned in a similar manner. This learning algorithm, called the delta learning rule (Rummelhart et al., 1986), traverses the error surface to find the point where the model error on the training set is minimized; thus, the delta rule performs a linear least-squares regression of the training data. In this study, the delta rule is augmented with a momentum factor, which adds a fraction of the previous weight update to the current one. Momentum has been shown to speed up convergence of the delta method and reduce the likelihood that it will become stuck in a local minimum on the error surface (Bishop, 1995). Both the learning rate and momentum factor are selected using a trial-and-error approach. In our case study, we tested learning rates between 0.001 and 0.1 in increments of 0.005 and momentum factors of 0, 0.001, and 0.1, and selected the learning rate/momentum factor combination that resulted in the model with the best performance, as quantified by 10-fold cross-validation.

The LN model has the same mathematical form as the traditional autoregressive (AR) model of Box and Jenkins (1970), and thus has similar capabilities. However, unlike many popular methods to determine the parameters of this model (i.e., b and $\{w_0, \ldots, w_{q-2}\}$), such as the Yule–Walker equations (Brockwell and Davis, 2002), the delta rule used in this study does not require the assumption of stationarity.

2.1.4. Multilayer perceptron (MLP) predictor

MLPs (Bishop, 1995; Duda et al., 2001; Hastie et al., 2001) create models of a system state using non-linear combinations of the input variables. The ability of MLPs to model time-series data is well documented (e.g., Bishop, 1995) and recent studies have shown that MLPs can be superior to traditional ARIMA time-series models (Dellana and West, 2009; Trapletti et al., 2000). The MLP considered in this study is a feed-forward network with sigmoid activation functions in the hidden layers and a linear activation in the output node. The MLP is trained using the backpropagation algorithm with gradient descent and momentum described above. Training, via 10-fold cross-validation, was terminated when further training caused a decrease in the model performance on the validation set (Bishop, 1995; Hastie et al., 2001). This latter condition discourages overtraining of the network. MLPs require that a learning rate, momentum factor, number of hidden layers, number of nodes in each hidden layer, and maximum number of training epochs be specified. These values were selected using a trial-and-error approach (Bishop, 1995; Hecht-Nielsen, 1990) that searched combinations of architectures and learning rates and momentum factors for best performance as quantified by the 10-fold cross-validation. Since Bishop (1995) indicates that more than one hidden layer is often not necessary, only architectures with one hidden layer were examined in this approach. The number of nodes

in each hidden layer was set through exhaustive search of architectures with between $2\sqrt{n}+m$ and $2n+1$ nodes in the hidden layer (where n is the number of inputs and m is the number of outputs) as suggested by Hecht-Nielsen (1990). As with the LN model, for this case study the learning rate was varied between 0.001 and 0.1 in increments of 0.005 for each architecture.

2.2. Steps 2 and 3: anomaly classification

Given the model prediction from step 1, the sensor measurement can then be classified as anomalous using a constant threshold calculated via the PI. The PI gives the range of plausible values that the next measurement can take, and the prediction level ($p = 100(1-\alpha)$) indicates the expected frequency with which measurements will actually fall in this range. If it is assumed that the model residuals have a zero-mean Gaussian distribution, the p% PI can be calculated as:

$$\overline{x} \pm t_{\alpha/2,n-1} \times s\sqrt{1+\frac{1}{n}} \tag{5}$$

where $\overline{x}$ is the one-step-ahead prediction of the sensor measurement, $t_{\alpha/2,n-1}$ is the pth percentile of a Student's t-distribution with $n-1$ degrees of freedom, s is the standard deviation of the model residual, and n is the sample size used to calculate s. This type of PI is a type of t-interval, because it relies on Student's t-distribution. If the new measurement falls within the bounds of the PI, then the measurement is classified as non-anomalous; otherwise, it is classified as anomalous. Thus, the PI represents a threshold for acceptance or rejection of a data point. The benefit of using the PI instead of an arbitrary threshold is that the prediction level guides the selection of the interval width.

2.3. Steps 4a and 4b: AD and ADAM

Once an anomalous data point is identified, two strategies for processing future data are compared. The first strategy is to simply flag the data point as anomalous and proceed to calculate the PI for the next chronologically sequential data point using the newly classified anomalous data point as input to the data-driven model of the data stream. The second strategy is to flag the data point as anomalous and proceed to calculate the PI for the next chronologically sequential data point, using the data-driven model prediction of the anomalous data point (instead of the newly classified anomalous data point itself) as an input to the data-driven model of the data stream. The former strategy will hereafter be referred to as anomaly detection (AD), while the latter strategy will be referred to as anomaly detection and mitigation (ADAM).

2.4. Case study

To demonstrate the efficacy of the anomaly detection methods developed in this study for data QA/QC, it was applied to a windspeed sensor data stream from Corpus Christi Bay. The sensor is an R.M. Young model 05106 marine wind monitor, which collects windspeed and direction at a frequency of 1 Hz. The AD and ADAM strategies were tested using all four modeling methods and 95% and 99% PIs. These 16 combinations will hereafter be referred to as anomaly detectors.

The windspeed models used in these detectors were developed using data from the period of January–May 2004. A plot of the autocorrelation function (ACF), along with a comparison of the mean and variance of the windspeed calculated from three non-overlapping 40-day windows, was performed. The results indicate that, over these 5 months, the windspeed behaved like a stationary process. A model independent technique (Kaplan, 1993) was used to select the size of the moving window needed to model the windspeed, which indicated that a moving window of 30 previous measurements (i.e., $q = 30$ in Eq. (1)) was adequate. The windspeed models were trained using 30,000 examples selected at random from the period of January–May 2004 and the resulting anomaly detectors were tested on data collected in June 2004. The user-defined parameters for these models, which were selected as described in Section 2.3, are shown in Table 1.

Fig. 2 shows the predictive abilities of the generated data-driven models on a segment of the windspeed data collected on June 15, 2004, which are part of the testing dataset. The NC predictor appears to be the least accurate, while the naïve predictor appears to be the most accurate. The 16 anomaly detectors were then compared based on their ability to identify erroneous data in the testing data.

Since the data used in this study were subjected to manual quality control measures before they were archived, it was expected that the detectors would not identify many data anomalies in the archive. However, this was not the case. The detectors identified approximately 6% of the data during the month of June as anomalous. This result encouraged focused inspection of these data. For example, Fig. 3 shows a 2-min segment of the data stream from June 15, 2004, in which six suspicious events affecting nine data points can be easily identified. All six of these events had been classified as anomalous by one or more of the 16 detectors. Subsequent investigation of these types of data anomalies by the SERF data managers revealed that events such as these were most likely caused by wireless transmission errors. While it is still unknown why visual QA/QC was unable to detect these anomalies, it is suspected that it was the result of the QA/QC method's employment of ensemble statistics to ease operator fatigue. These statistics may have smoothed out these anomalies, causing them to go undetected. Further analysis of anomalous data revealed other suspicious events that were of significantly longer duration than those shown in Fig. 3. For example, Fig. 4 shows a 35-min segment of the data stream from June 22, 2004, during which a suspicious long duration event occurs. The windspeed between minutes 8 and 26 in the plot appears to have been offset by a constant 7 m/s. It is the sharpness of the transition from the slower ($\sim$5 m/s) to the faster ($\sim$12 m/s) windspeed and back, as well as the existence of data in both the slow and fast regimes that appear to correlate with data in the opposite regime, which suggests that a significant portion of the data presented in this figure does not represent the actual windspeed. This data segment is of particular interest because its behavior is similar to the behavior of a type of sensor fault (offset bias fault) described by Koushanfar et al. (2003), which causes sensor data to be offset by a constant value. Errors resulting from short-duration faults, such as those shown in Fig. 3, may not have a significant effect on the utility of the data if time averages are used. However, due to the high frequency with which these events are observed in this data stream, even moderate time averages are adversely affected. For example, the 2-min average (a standard granularity for wind data collected by the National Oceanic and

Table 1
Values for data-driven time-series models.

Model	Parameter	Values
NC	Number of clusters (k_c)	6
LN	Learning rate	0.01
	Momentum factor	0.1
MLP	Learning rate	0.01
	Momentum factor	0.1
	Number of hidden layers	1
	Number of nodes in first hidden layer	50

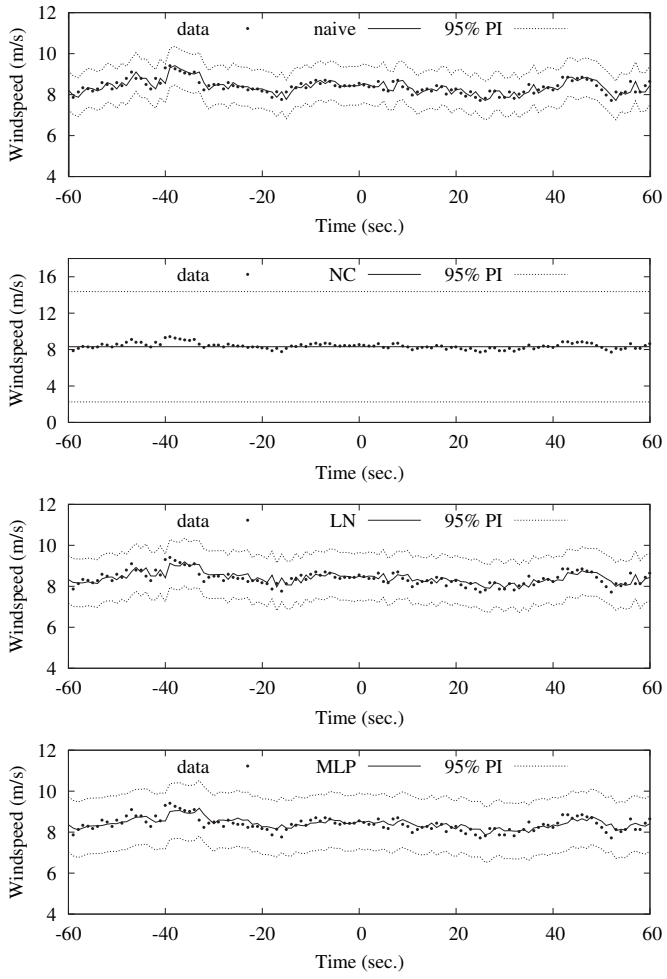

Fig. 2. Performance of different data-driven methods for predicting the Corpus Christi Bay windspeed data stream. Note the different *y*-axis scale for the NC predictor.

Atmospheric Administration's National Data Buoy Center) of the raw data shown in Fig. 3 has a value of 7.2 m/s, whereas the 2-min average of the data with the nine anomalous data points removed has a value of 8.3 m/s, a difference of approximately 15%. Long-duration errors, such as those shown in Fig. 4, are even more worrisome because their effect can only be mitigated if very long time averages are used.

The existence of errors in the June 2004 data indicated that the data from January to May (used for training) also contained errors. Thus, before proceeding with an assessment of the 16 detectors, it was necessary to clean the training data and retrain the regression models. Cleaning was performed using the naïve-AD detector with a 95% PI. The naïve detector was chosen because it performs well in identifying anomalous data and because it does not rely on a model of the data stream that could have been affected by errors in the

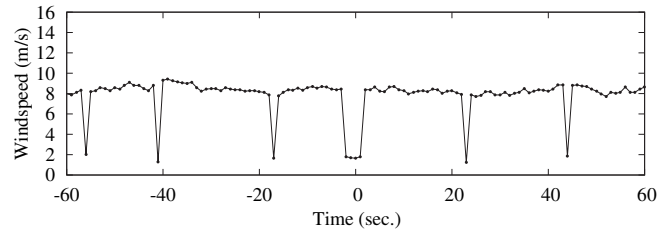

Fig. 3. Data exhibiting errors resulting from short duration faults.

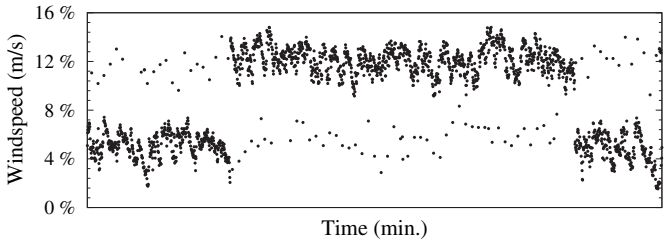

Fig. 4. Data exhibiting errors resulting from long duration faults.

uncleaned training data. The AD strategy was used because its long-term performance is not affected by previous misclassifications (as opposed to the ADAM strategy, which replaces measurements classified as anomalous with its own estimate, based on prior data, thus perpetuating the effect of classification errors over the long term). Records containing data classified as anomalous were removed from the training set.

Once the training data were cleaned, the models were retrained and the residuals were tested for normality using a Lillifors test at the 5% level. The test indicated that although the distributions were symmetric and unimodal, they had thicker tails than a normal distribution. This feature of the residual distributions can be illustrated using a quantile–quantile (Q–Q) plot, such as the one shown in Fig. 5, which compares the quantiles of the MLP model residuals (*y*-axis) with the quantiles of a standard normal distribution (*x*-axis). The Q–Q plot compares the quantiles of one distribution against the quantiles of another distribution and is used to identify differences between the two distributions. A reference line is also plotted. If the two distributions are similar, then the points will fall approximately on the reference line shown in Fig. 5, whereas departures from this line indicate differences in the distributions. It can be seen in Fig. 5 that the points fall on the reference line in the central region of the distribution (e.g., around the mean) but depart from the reference line towards the tails. This departure becomes obvious around $x = \pm 1.5$ and is caused by the MLP model residual quantiles having a larger magnitude than the standard normal quantiles in the tail region, thus indicating that the MLP model residuals have thicker tails than those of a normal distribution.

This result affects the validity of the implicit assumption in the PI calculation (Eq. (5)) that the model residuals are normally distributed. However, since the central region of the model residual distribution behaves similarly to that of a normal distribution, the PI calculated using Eq. (5) in this region will be unaffected. In the

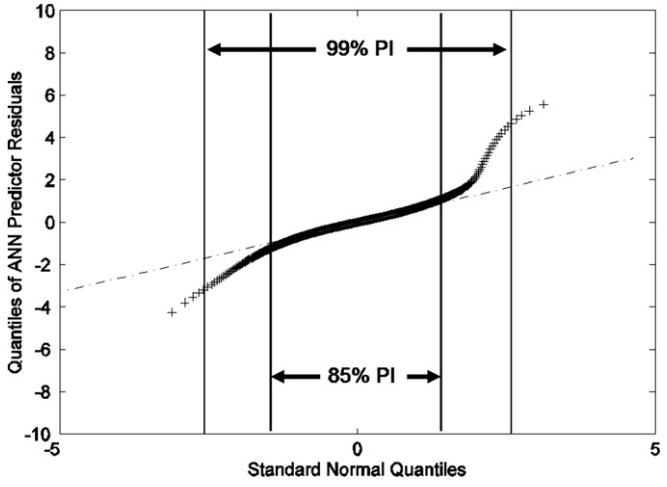

Fig. 5. Q–Q plot of MLP model residual distribution and standard normal distribution. Vertical lines indicate bounds of 85% and 99% PIs.

case of the MLP model residuals, this region covers approximately $x = \pm 1.5$. Thus, for prediction levels of 85% or less, the PI calculated with Eq. (5) will accurately bound the region of plausible measurement values that the measurements will take with a frequency of $1 - p$. For prediction levels greater than approximately 85%, the PI will underestimate the frequency that the measurement will fall outside of the interval resulting in a higher false positive rate than $(1 - p)$% This results from the existence of a higher frequency of extreme (tail) values in the model residual distribution than would be indicated by a normal distribution. Since this effect increases as the PI bounds move further into the tails of the model residual distribution, the false positive rate will be increasingly larger than $(1 - p)$%. However, the extent of the deviation for this data set appears to be moderate for prediction levels less than 99%. For example, the vertical lines in Fig. 5 are at positions $x = \pm 2.58$ on the standard normal quantiles and indicate the bounds of the 99% PI. Within this region, the quantiles of model residual distribution deviate moderately from the quantiles of the normal distribution; thus, it is expected that the false positive rate enhancement for prediction levels less than 99% will be moderate. The other model residual distributions exhibit behavior similar to that of the MLP model residuals, indicating that for prediction levels between 85% and 99%, the PI will moderately overestimate the frequency that a measurement will fall within its bounds, thus causing slightly higher than expected false positive rates.

To test this observation, the performance of the 16 anomaly detectors for identifying additional erroneous data was then quantified using a sample of over 2700 other data points from the data archive that had not been cleaned. True/false positives were identified visually, using domain knowledge provided by the SERF data managers. The data managers indicated that measurements that deviate from their close neighbors by approximately 2 m/s or more and that break the relatively smooth trend of the windspeed are most likely erroneous. For example, the erratic oscillations noted in Figs. 2 and 3 are erroneous because an air mass cannot plausibly accelerate and then decelerate at the rate required to cause such oscillations.

Table 2 shows the detectors' false positive rate for identifying erroneous data in the windspeed data stream. It can be seen that the ADAM strategy reduces the false positive rates of the LN and MLP-based detectors, whereas it increases the false positive rates of the naïve and NC-based detectors. For the LN and MLP-based detectors, without the use of mitigation, previously processed erroneous data adversely affect future classifications. This occurs because using erroneous data as inputs to the LN or MLP models requires the model to make a prediction using a type of input data (i.e., erroneous measurements) with which it was not trained (i.e., to extrapolate beyond the training data), thus reducing the accuracy of the one-step-ahead predictions and their corresponding PIs. The naïve and NC-based detectors, however, appear to be less sensitive to erroneous input data. Unlike the LN and MLP detectors, the naïve and NC-based methods do not predict the future windspeed using a function of the input values. Rather, these detectors predict the future windspeed using a similar previously observed example. Thus, input measurements to these detectors that vary significantly from the current locally averaged windspeed (e.g., data errors) will not cause a future windspeed prediction that is vastly different from previously observed windspeeds.

However, the use of the ADAM strategy can negatively affect the performance of a detector when it misclassifies a non-anomalous point as anomalous, because this causes the detector to replace the valid measurement with an incorrect value, which sometimes results in the detector continuing to make mistakes and replacing valid measurements with incorrect values until the cycle is broken. In the case of the LN and MLP-based detectors, this behavior is uncommon and outweighed by the positive benefits of mitigation described previously, whereas in the case of the naïve and NC-based detectors, this behavior significantly affects the detectors' performance. Since the naïve-based detectors use only one data point to predict the future windspeed, it is understandable that, when using the ADAM strategy, this method perpetuates misclassifications because future predictions reflect the incorrect values that replaced valid measurements. However, it is less clear why the NC-based detectors behave in this manner. Perhaps it is because the cluster membership is more heavily influenced by recent measurements than by distant measurements.

Table 2
False-positive and false-negative rates for detecting June 2004 windspeed data errors using 95% and 99% PIs.

Model	95% PI		99% PI	
	False positive (%)	False negative (%)	False positive (%)	False negative (%)
Naïve-AD	5.18	5.30	4.33	6.82
Naïve-ADAM	6.38	0.00	2.78	2.27
NC-AD	0.387	89.4	0.0387	100
NC-ADAM	3.36	92.4	0.0773	100
LN-AD	10.7	5.30	6.84	9.85
LN-ADAM	1.86	0.00	0.271	3.79
MLP-AD	8.16	0.00	4.68	11.4
MLP-ADAM	0.851	1.52	0.155	3.03

In addition, the use of model predictions as model input in the ADAM approach may result in smoothing out the time series towards its central tendency (since the variability of the model predictions is lower than that of the raw measurements), thus biasing the detector towards classifying future data as anomalous. This effect did not appear to significantly affect the detectors, though it could be more pronounced when the anomalies persist for a longer period of time such that the moving window is predominantly composed of predictions rather than actual data.

Table 2 also indicates that an increase in the prediction level from 95% to 99% decreases the number of false positives, though this decrease is not as dramatic as the change associated with the use of mitigation, indicating that the appropriate use of mitigation has the most significant effect on the detectors' false positive rate.

The false negative rate for the detectors is also shown in Table 2. It can be seen that the naïve, LN, and MLP-based detectors misclassify significantly fewer erroneous data than the NC-based detectors, which misclassify almost all of the erroneous data. Thus, the NC-based detectors are not useful for detecting erroneous data in the windspeed data stream. It can also be seen that the use of the ADAM strategy significantly improves the ability of the LN-based detectors to correctly classify erroneous data, whereas it does not significantly affect the false negative rates of the other detectors. The decrease in the LN-based detectors' false negative rates, due to the use of the ADAM strategy, can again be attributed to erroneous data adversely affecting future classifications by requiring extrapolation of the LN model. Furthermore, an increase in the prediction level from 95% to 99% results in an increase in the false negative rate, which, for the best-performing detectors, is larger than the corresponding decrease in the false positive rate, indicating that a 95% PI provides a reasonable trade-off between misclassifying erroneous and non-erroneous data in this data stream.

3. Discussion

The anomaly detectors used in the case study represent 16 instantiations of the proposed anomaly detection method, which relies on a threshold deviation from a model prediction to delineate the boundary between anomalous and non-anomalous data. This type of method will misclassify both anomalous data that have

smaller deviations than the threshold and non-anomalous data that have larger deviations than the threshold. Thus, selecting an appropriate threshold is important. The benefit of using the p% PI, rather than an arbitrary, user-defined threshold, is that the PI provides guidance (in the form of the prediction level) for the selection of the threshold without requiring any knowledge of the process variables being measured. The p% PI indicates the region likely to contain at least p% of the possible sensor measurements, given both the modeling and measurement errors; thus, we expect that approximately $(1-p)$% of the non-anomalous data will be misclassified as anomalous. Despite the anticipation of a slightly higher false positive rate resulting from the heavy-tail behavior of the model residual distributions (Fig. 5), when mitigation is applied appropriately (as described in Section 2.4) the false positive rates (Table 2) for all the detectors in the case study except the naïve-AD detector with 99% PI do not exhibit this behavior. This indicates that the normal assumption implicit in the PI calculation is reasonable for the Corpus Christi Bay windspeed data.

Note that the prediction level, however, cannot give an *a priori* estimate of the false negative rate, because the behavior of the anomalous data is unknown, and thus it is impossible to estimate how many of these data will fall inside a given interval. For this reason, the prediction level should be set such that the false positive rate is reasonably low (e.g., prediction levels of 95% or 99%). Fortunately, in many practical applications of anomaly detection, such as data QA/QC, setting the prediction level in this way yields good results because anomalous data that fall outside the PI are more important to classify correctly than those that fall inside the PI, since, given a reasonably accurate model of the time-series data, falling inside the PI indicates that anomalous data are roughly within the measurement error of the sensor. For example, the nine obvious erroneous data points in Fig. 3 were identified by the MLP-ADAM detector with 95% PI, and cleaning these points changed the average value of all 120 points by approximately 15%.

In addition to providing a principled framework for threshold selection, the proposed anomaly detection method allows a detector for one data stream to operate independently of the behavior of other data streams or their detectors by utilizing univariate autoregressive models of the data stream. The use of univariate autoregressive models, however, precludes the use of information gathered by other sensors, without which the anomaly detector may misclassify certain types of data anomalies, but the case study demonstrates that this limitation does not significantly affect the success of the anomaly detection method for performing QA/QC on one Corpus Christi Bay windspeed data stream.

Finally, the data-driven autoregressive anomaly detection method developed in this study must accommodate two features of environmental sensor data streams: non-stationary data (i.e., data whose pattern changes with time) and gaps in the sensor data stream. Because many environmental processes are not stationary (Amenu et al., 2007) over long time periods, data-driven models must be updated periodically. For batch trained models, such as those employed in this case study, model updating can be accomplished by retraining the models periodically. The frequency with which a model will need to be retrained will depend on the rate at which the model parameters change, a feature that will vary between applications. In the case study presented here, the windspeed appears to be stationary over a 5-month period, indicating that retraining the models monthly would be more than sufficient to keep up with the time evolution of the windspeed process. However, future sensor data should be analyzed to adapt the training frequency to any future changes in the windspeed process dynamics.

Batch retraining can be time consuming, but, because of energy constraints, sensor data are usually sent from the sensor to the data archive in small batches, rather than one at a time, so retraining only has to be faster than the frequency of data broadcasts, not faster than the sensor sampling rate. In the case of the naïve predictor, training can be performed in approximately 30 s. Since the SERF windspeed data are sent in 5-min intervals, model updating can occur between data transmissions. Training of the NC, LN, or MLP predictor requires time-consuming identification of user-defined parameters (number of clusters for the NC predictor, learning rate for the LN predictor, and architecture and learning rate for the MLP predictor). Since the data arrive faster than retraining can occur, a dual-model approach can be used, where a new model is trained while the previous model is being used for anomaly detection.

Alternatively, on-line training algorithms (i.e., algorithms in which the model is incrementally updated for new data as they arrive from the sensor) could be considered for updating the models (e.g., Bifet and Gavaldà, 2009, 2006). However, with an on-line training algorithm, it is unclear how best to calculate the standard deviation of model errors needed for the PI calculation (Eq. (2)). Therefore, batch retraining is preferable for updating data-driven models to account for changing temporal patterns in the data.

Gaps in the sensor data stream are the result of the sensor going off-line because of the harsh environment in which the sensors operate. The windspeed data set considered in the case study has over 500,000 gaps of durations ranging from 1 s to 90 days. The vast majority (99.99%) of these gaps, however, are less than 5 s in duration. Because the detectors require a defined set of input features, these gaps render the error detectors unable to classify certain future measurements as anomalous/non-anomalous. The naïve-based detectors require the measurement at a time 1 s previous to the prediction time; thus, they cannot classify the first measurement after a sensor goes back on-line. The remaining methods require measurements at time minus one through 30 s in order to classify a new measurement; thus, they cannot classify any data within the first 30 s after the sensor goes back on-line. Some of these measurement gaps can be addressed by using the detector to fill in missing values with the predicted value of the measurements. This method will only work for short duration gaps, though, as the accuracy of the model prediction will degrade with successive iterations. However, since only 0.01% of the gaps are longer than 4 s, this method will be useful in the majority of cases. For the few remaining cases, manual inspection of a maximum of 30 data points per sensor is likely to be feasible, especially since multiple sensors are unlikely to fail at the same time.

4. Conclusion

Real-time detection of anomalies in environmental streaming data has many practical applications, such as data QA/QC, adaptive sampling, and anomalous event detection. This research developed an anomaly detection method based on univariate autoregressive data-driven models of the sensor data stream and the PI. This method performs fast, incremental evaluation of data as they become available, can scale up to large quantities of data, and requires no pre-classification of anomalies. The value and efficacy of this anomaly detection method for data QA/QC is illustrated using a case study involving a windspeed data stream from Corpus Christi Bay. Anomaly detectors using different data-driven modeling techniques and both the AD and ADAM anomaly handling strategies identified a significant number of erroneous measurements in the windspeed data that manual QA/QC had failed to detect. The errors had durations ranging from 1 s to several minutes and affected approximately 6% of the data. After cleaning the errors in the training data, an assessment of eight instantiations of both the AD and ADAM strategies indicated that the performance of the LN and MLP-based detectors in detecting errors in the testing data was

significantly improved by the use of the ADAM strategy, and that the MLP-ADAM detector using a 95% PI performed best, with a false positive and a false negative rate of 1% and 2%, respectively.

The case study results suggest that the AD and ADAM anomaly detection methods developed in this study are useful tools for identifying anomalies in environmental streaming data. Since these methods only require a time-series model of the data stream (rather than model of the environmental process), they can be easily applied to many real-time environmental sensor data. However, it should be noted that, while the MLP produced the best model for the windspeed data considered in the case study, this model may not be the most appropriate choice for other types of environmental data. The four data-driven techniques considered in this paper have been shown to produce good predictions on different types of time-series data and are a useful starting point for selecting the most appropriate model, but the anomaly detection method developed here is not restricted to these models and can easily accommodate other data-driven modeling techniques.

There are at least two areas for extensions of this research. The first area is to investigate methods that relax the crisp membership of measurements to the class of anomalous/non-anomalous data, for example by using fuzzy logic or Bayesian methods to assign a degree of membership of a measurement to the class of anomalous data. The second extension would be to investigate methods that can use data from multiple sensors for anomaly detection in a single-sensor data stream. The use of multi-sensor data will allow the anomaly detector to exploit spatiotemporal correlations inherent in environmental processes (e.g., if windspeed at a particular location increases, it will also usually increase at nearby locations), and thus, improve the capability of the anomaly detector. Such methods, however, will have to overcome the difficulty of modeling the sensor data stream with an incomplete set of model input data caused by missing or delayed sensor measurements.

Acknowledgments

The authors would like to thank Desiree Trujillo, John Perez, Terry Riggs, and Jim Bonner of Shoreline Environmental Research Facility, Corpus Christi, TX, for providing data and knowledge regarding their sensors. This study was supported by the Office of Naval Research under grant N00014-04-1-0437. The authors gratefully acknowledge the help and valuable suggestions of Dr Praveen Kumar, University of Illinois at Urbana-Champaign.

References

Amenu, G., Markus, M., Kumar, P., Demissie, M., 2007. Hydrologic applications of Minimal Resource Allocations Network (MRAN). Journal of Hydrologic Engineering 12 (1), 124–129.

Bannayan, M., Hoogenboom, G., 2008. Weather analogue: a tool for real-time prediction of daily weather data realizations based on a modified *k*-nearest neighbor approach. Environmental Modelling and Software 23, 703–713.

Belle, R., Upadhyaya, B., Skorska, M., 1983. Sensor fault analysis using decision theory and data-driven modeling of pressure water reactor subsystems. Nuclear Technology 64, 70–77.

Bifet, A., Gavaldà, R. 2006. Kalman filters and adaptive windows for learning in data streams, in: Proceedingsof the 9th International Conference on Discovery Science (DS 2006). Springer-Verlag Lecture Notes in Artificial Intelligence 4265, 29–40.

Bifet, A., Gavaldà, R. 2009. Adaptive parameter-free learning from evolving data streams. Technical Report LSI R09–9.

Bishop, C., 1995. Neural Networks for Pattern Recognition. Oxford University Press, New York.

Bolton R.J., Hand, D.J., 2001. Unsupervised profiling methods for fraud detection, in Proceedingsof Credit Scoring and Credit Control VII. Edinburgh, UK, pp. 5–7.

Bonner, J.S., Kelly, F.J., Michaud, P.R., Page, C.A., Perez, J., Fuller, C., Ojo, T., Sterling, M., 2002. Sensing the coastal environment, in: Proceedings of the Third International Conference on EuroGOOS; Building the European Capacity in Operational Oceanography, pp. 167–173.

Box, G., Jenkins, C., 1970. Time Series Analysis: Forecasting and Control. Holden-Day Inc., San Francisco, CA.

Brockwell, P.J., Davis, R.A., 2002. Introduction to Time Series Forecasting. Springer-Verlag, New York.

Bulut, A., Singh, A.K., Shin, P., Fountain, T., Jasso, H., Yan, L., Elgamal, A., 2005. Real-time nondestructive structural health monitoring using support vector machines and wavelets, in: Meyendorf, N., Baakline, G.Y., Michel, B. (Eds.), Proceedings of the SPIE Advanced Sensor Technologies for Nondestructive Evaluation and Structural Health Monitoring, Vol. 5770, pp. 180–189.

Dellana, S.A., West, D., 2009. Predictive modeling for wastewater applications: linear and nonlinear approaches. Environmental Modelling and Software 24, 96–106.

Duda, R., Hart, P., Stork, D., 2001. Pattern Classification. Wiley-Interscience, New York.

Evans, M., 2002. Practical Business Forecasting. Blackwell Publishing, Malden, MA.

Fantoni, P.F., Mazzola, A., 1996. Multiple failure signal validation in nuclear power plants using artificial neural networks. Nuclear Technology 113, 368–374.

Gjølberg, O., Bengtsson, B.-A., 1998. Forecasting quarterly hog prices: simple autoregressive models vs. naive predictions. Agribusiness 13 (6), 673–679.

Han, J., Kamber, M., 2006. Data Mining: Concepts and Techniques, second ed. Morgan Kaufmann, New York.

Hastie, T., Tibshirani, R., Friedman, J., 2001. The Elements of Statistical Learning. Springer-Verlag, New York.

Hecht-Nielsen, R., 1990. Neurocomputing. Addison-Wesley, New York.

Hodge, V.J., Austin, J., 2004. A survey of outlier detection methodologies. Artificial Intelligence Review 22, 85–126.

Illa, J.M.G., Alonso, J.B., Marré, M.S., 2004. Nearest-neighbours for time series. Applied Intelligence 20, 21–35.

John, G.H., 1995. Robust decision trees: removing outliers from databases, in: Proceedings of the 1st International Conference on Knowledge Discovery and Data Mining, pp. 174–179.

Kaplan, D.T., 1993. In: Pecora, L.M. (Ed.), A model-independent technique for determining the embedding dimension, Vol. 2038. Chaos in Communications SPIE, pp. 236–240.

Koushanfar, F., Potkonjak, M., Sangiovanni-Vincentelli, A., 2003. On-line fault detection of sensor measurements. Proceedings of IEEE Sensors, Vol. 2, 974–979.

Kozuma, R., Kitamura, M., Sakuma, M., Yokoyama, Y., 1994. Anomaly detection by neural network models and statistical time series analysis, in: Neural Networks 1994. IEEE World Congresson Computer Intelligence, Orlando, FL.

Krajewski, W.F., Krajewski, K.L., 1989. Real-time quality control of streamflow data–A simulation study. Water Resources Bulletin 25 (2), 391–399.

Mourad, M., Bertrand-Krajewski, J.-L., 2002. A method for automatic validation of long time series of data in urban hydrology. Water Science and Technology 45 (4-5), 263–270.

Nairac, A., Townsend, N., Carr, R., King, S., Cowley, P., Tarassenko, L., 1999. A system for the analysis of jet engine vibration data. Integrated Computer-Aided Engineering 6 (1), 53–65.

NRC (National Research Council), 2006. CLEANER and NSF's Environmental Observatories. National Academy Press, Washington, DC.

NSF (National Science Foundation), 2005. Sensors for Environmental Observatories: Report of the NSF Sponsored Workshop December 2004. NSF, Arlington, VA.

Ramaswamy, S., Rastogi, R., Shim, K., 2000. Efficient algorithms of mining outliers from large data sets. Proceedings of the ACM SIGMOD Conference on Management of Data. Dallas TX, 427–438.

Rousseeuw, P., Leroy, A., 1996. Robust Regression and Outlier Detection, third ed. John Wiley & Sons, New York.

Rummelhart, D.E., Hinton, G., Williams, R., 1986. In: Rummelhart, D.E., McClelland, J.L., theResearch Group, P.P.D. (Eds.), Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Learning internal representations by error propagation, Vol. 1. MIT Press, Cambridge, MA, pp. 318–362.

Schütze, H., Silverstein, C. 1997. Projections for efficient document clustering, in: Proceedings of SIGIR '97, Philadelphia, PA, pp. 74-81.

Silvestri, G., Verona, F., Innocenti, M., Napolitano, M., 1994. Fault detection using neural networks, in: Proceedings of the IEEE International Conference on Neural Networks, Vol. 6, pp. 3796–3799.

Tang, J., Chen, Z., Fu, A., Cheung, D., 2002. A robust outlier detection scheme in large data sets, in: Proceedingsof the 6th Pacific-Asia Conference on Knowledge Discovery and Data Mining. Taipei, Taiwan, May 2002.

Thomakos, D.D., Guerard, J.B., 2004. Naive, ARIMA, nonparametric, transfer function and VAR models: a comparison of forecasting performance. International Journal of Forecasting 20 (1), 53–67.

Trapletti, A., Leisch, F., Hornik, K., 2000. Stationary and integrated autoregressive neural network processes. Neural Computation 12, 2427–2450.

Upadhyaya, B., Glockler, O., Eklund, J., 1990. Multivariate statistical signal processing for fault detection and diagnostics. ISA Transactions 29 (4), 79–95.