Cluster-Based Anomaly Detection in Condition Monitoring of a Marine Engine System

Erik Vanem and Andreas Brandsæter

Group Technology and Research
DNV GL
Høvik, Norway
Erik.Vanem@dnvgl.com,
Andreas.Brandsaeter@dnvgl.com

Department of Mathematics
University of Oslo
Oslo, Norway
erikvan@math.uio.no, andrbran@math.uio.no

***Abstract*— Sensor data can be used to detect changes in the performance of a system in near real-time which may be indicative of a system fault. However, there is a need for efficient and robust algorithms to detect such changes in the data streams. In this paper, sensor data from a marine diesel engine on an ocean-going ship are used for anomaly detection. The focus is on cluster-based methods for anomaly detection. The idea is to identify clusters in sensor data in normal operating conditions and to assess whether new observations belong in any of these clusters. New observations that obviously do not belong to any of the identified clusters, may be regarded as anomalies and call for further scrutiny. The cluster-based approaches to anomaly detection presented in this paper are truly unsupervised, and they are applied to sensor data with no known faults. Being fully unsupervised, however, the cluster based approaches do not need to explicitly assume that all observations in the training data are fault-free as long as the amount of faulty data is not large enough to form a separate cluster. Moreover, anomalies in the training data may be detected.**

Various clustering techniques are applied in this paper to provide, a simple and unsupervised approach to anomaly detection. This could then be used as an efficient initial screening of the data streams before more detailed analysis is applied to suspicious parts of the data. The methods explored in this study include K-means clustering, Mixture of Gaussian models, density based clustering, self-organizing maps and spectral clustering. The performance of the various methods is reported, and also compared to other methods recently proposed for anomaly detection such as auto associative kernel regression (AAKR) and dynamical linear models (DLM). Overall, cluster-based methods are found to be promising candidates for online anomaly detection based on sensor data.

Keywords*— *Anomaly detection; Condition monitoring; sensors; data-driven methods; unsupervised learning

I. INTRODUCTION

Sensor data collected from machinery systems on board ships provide real-time information about the condition of the ship. Such sensor-based condition monitoring can be used to detect changes in the performance of the system in near real-time which may be indicative of a system fault or even an impending fault. However, there is a need for efficient and robust algorithms to detect such changes in the data streams. Typically, a data-driven condition monitoring system include anomaly detection, fault identification and prognostics. The first task is to monitor the data streams to detect deviations from normal system behavior indicative of a change of the system. This is referred to as anomaly detection, and for this task only nominal data is needed to train the algorithms. The next step is fault isolation or fault identification where a diagnostic tool is applied to estimate what type of deviation the anomaly is, i.e. to distinguish real faults from unexpected but normal behavior, and to identify the type of fault. In order to train such algorithms, information (data) about both normal and faulty states of the system is needed. Essentially, in a data-driven approach, this is a classification task, where one need labelled data in order to perform the classification (see e.g. [1] for a review of classification methods that may be used for this purpose). Finally, the prognostics task try to estimate the future behavior of the system, conditioned on the current state, and to estimate the remaining useful life (RUL). Typically for this task to be feasible with a data-driven approach, there is a need for run-to-failure data under varying conditions, something which is rarely available.

Sensor data from an ocean-going ship are analyzed in this paper, where data from various sensors installed at different places within a marine diesel engine are collecting essential parameters such as power output from the engine, engine speed, bearing temperatures and various other temperatures, speeds and pressures for selected engine components. The idea is to utilize the information contained in these sensor signals to monitor the condition of the engine. In particular, if very different patterns in new data compared to the data that has been used to train the anomaly detection models are observed, this can be regarded as a potential fault. The initial data streams collected from the ship are high-dimensional, with more than 100 data streams covering e.g. four different generator engines. However, in this study, a subset of the data streams is carefully chosen for analysis. Only one out of four main generator engines are investigated and the signals that are believed to be informative about the condition of the engine is selected, based on engineering knowledge. Hence, what remains is a 24-dimensional dataset that will be used for condition monitoring. Further dimension reduction is applied in order to alleviate condition monitoring and anomaly detection.

The focus of this paper is on unsupervised methods for anomaly detection based on clustering. The idea is to identify

clusters in the sensor data in normal operating conditions and to assess whether new data seem to belong in any of these clusters. New data that obviously do not belong to any of the identified clusters, may be regarded as anomalies and call for further scrutiny. Probably, such cluster-based anomaly detection techniques could be an efficient first screening of the data to give an indication of an anomaly. This could again trigger more detailed analysis of the data in order to diagnose the deviation and possibly flag an alarm. However, the cluster-based approach to anomaly detection may not be sufficient in itself to issue an alarm, and there are many ways for the data to fall outside a cluster without there being an actual fault in the system. Hence, the unsupervised techniques that are explored in this paper could be recommended for initial screening of the data and should be used in combination with other methods.

The cluster-based approaches to anomaly detection presented in this report is truly unsupervised, and they are applied to sensor data with no known faults. This does not mean, however, that the data are guaranteed to be without faults, and there may be unknown faults in the data. Being fully unsupervised, the cluster based approaches investigated in this report does not need to explicitly assume that all observations in the training data are fault-free aa long as the amount of faulty data is not large enough to form a separate cluster. This may be an advantage compared to for example the method based on AAKR [2, 3], where a faulty training data point may have a bigger influence on the signal reconstruction and thereby on the anomaly detection performance. With the cluster-based anomaly detection presented in this paper, also anomalies in the training data may be detected in a fully unsupervised manner, and there is no need to be completely confident that the training data contains absolutely no faults.

In this paper, various clustering techniques are applied to sensor data from a marine diesel engine. Previously, different approaches for anomaly detection have been applied to the same dataset, i.e. the use of dynamical linear models (DLM) and sequential testing [4] and the use of auto associative kernel regression (AAKR) [5]. Both these approaches are based on fitting a model to normal data and predict or reconstruct new sensor data and then comparing to the predicted or reconstructed signals. Sequential testing is then performed on the residuals, i.e. the differences between the predicted or reconstructed signal and the actual observed signal, in order to detect anomalies. In both cases, sequential probability ratio tests (SPRT) were applied. It can be noted that both these approaches applied some form of clustering, but not for anomaly detection directly. Even though these methods generally work well, they did encounter some problems with the marine engine data streams, due to the different operational conditions which give rise to spurious jumps in the data. This time- and operational state dependence in the data makes prediction and re-construction challenging and anomaly detection based on signal reconstruction or predictions are nor straightforward. Hence, in this paper, a simple and unsupervised approach to anomaly detection based on clustering is explored. This could then be used as an efficient initial screening of the data streams before more detailed analysis is applied to suspicious parts of the data.

II. Data Description and Exploratory Analysis

The dataset that is explored in this study contains several sensor signals that can be related to the main bearing condition of one of four separate diesel engines on a ship. It is noted that the collected data do not contain any known faults or failures of the system, and the data are not compared to maintenance logs of the system. The list of signals is included in Table I, and engine 1 contains 24 sensor signals. By inspecting the data, it is observed that the signals for MG1TE702 contain only zero-values, presumably due to malfunctioning sensor or loss of connectivity and these signals are excluded from the subsequent analysis.

TABLE I. Sensor signals in the dataset

Main generator engine 1	
MG019	MGE1 ENGINE SPEED [rpm]
MG1PT201	MGE1 LO PRESS ENGINE INLET [bar]
MG1PT401	MGE1 HT WATER JACKET INLET PRESS [bar]
MG1PT601	MGE1 CHARGE AIR PRESS AT ENGINE INLET [bar]
MG1SE518	MG1 TC A SPEED [rpm]
MG1SE528	MG1 TC B SPEED [rpm]
MG1TE201	MGE1 LO TEMP ENGINE INLET [C]
MG1TE272	MGE1 LO TEMP TC OUTLET A [C]
MG1TE282	MGE1 LO TEMP TC OUTLET B [C]
MG1TE511	MGE1 EXHAUST GAS TEMP TC A INLET [C]
MG1TE517	MGE1 EXHAUST GAS TEMP TC A OUTLET [C]
MG1TE521	MGE1 EXHAUST GAS TEMP TC B INLET [C]
MG1TE527	MGE1 EXHAUST GAS TEMP TC B OUTLET [C]
MG1TE600	MGE1 AIR TEMP TC INLET [C]
MG1TE601	MGE1 CHARGE AIR TEMP AT ENGINE INLET [C]
MG1TE700	MAIN BEARING NO 0 TEMP MGE1 [C]
MG1TE701	MAIN BEARING NO 1 TEMP MGE1 [C]
MG1TE702	MAIN BEARING NO 2 TEMP MGE1 [C]
MG1TE703	MAIN BEARING NO 3 TEMP MGE1 [C]
MG1TE704	MAIN BEARING NO 4 TEMP MGE1 [C]
MG1TE705	MAIN BEARING NO 5 TEMP MGE1 [C]
MG1TE706	MAIN BEARING NO 6 TEMP MGE1 [C]
MG1TE707	MAIN BEARING NO 7 TEMP MGE1 [C]
PM100.07	MG1 POWER [kW]

The sensor signals cover a period of about 10 months starting from December 2014 with a sampling frequency of one minute. However, before the data is analyzed in this study, the hourly means are extracted from the data and used in the subsequent analysis. It is observed that many of the signals are highly correlated. For example, the various temperature measurements for the main bearings are all very strongly correlated, see the traceplots in Fig. 1. A couple of interesting features that can be observed from this plot is a) that all the temperature sensors seems to fall out at a certain point (all drops to zero) and b) that the sensor for main bearing No 2 seems to stop working and reports the same value over a period of time. Traceplots of the engine speed is also shown in the figure. The figures show both the initial data (ever minute) and the hourly mean data. It is observed that the effect of the

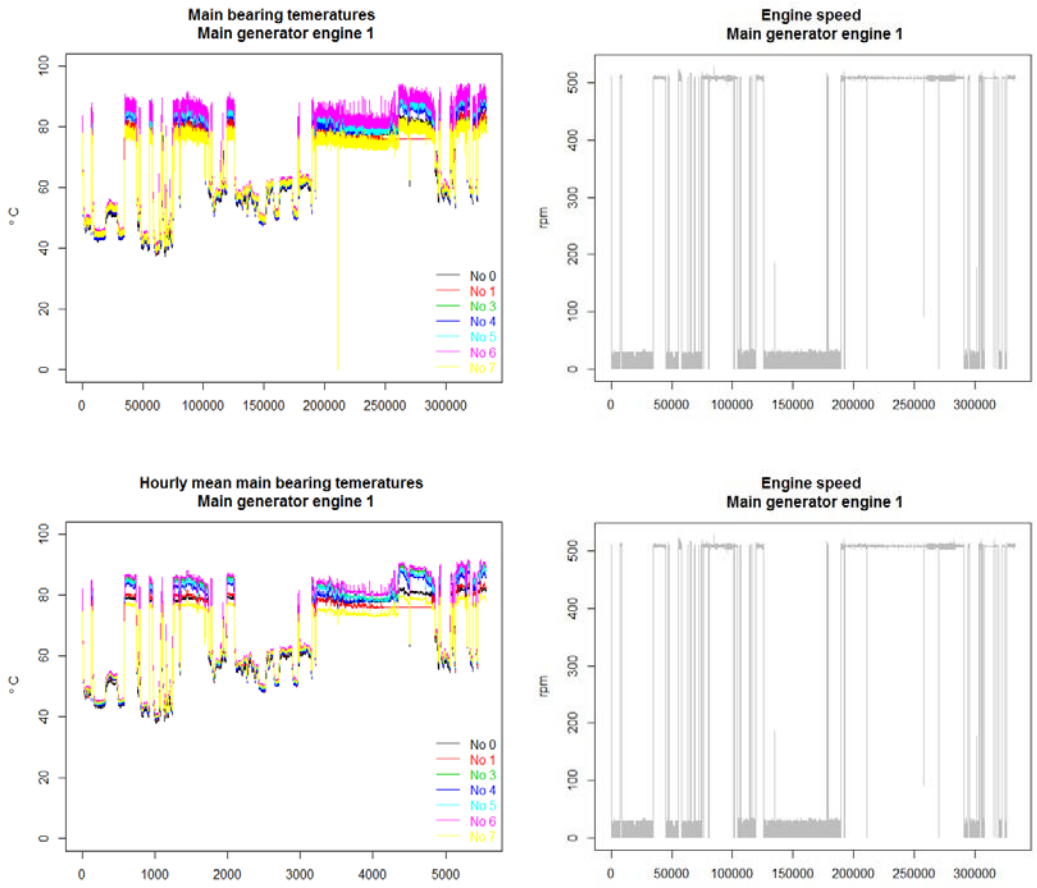

Fig. 1. Example of traceplots illustrating strong correlation between some of the signals; various temperature readings for main bearing temperatures (left) and engine speed (right); minute data (top) and hourly averaged data (bottom)

sensors falling out is vanishing in the hourly averaged data. Furthermore, small fluctuations in the signals for the various temperatures and in the engine speed are removed by taking the hourly average. This presumably makes it easier to detect anomalies by reducing the noise in the data without affecting the actual signal. Note that the reduced dataset contains hourly averaged values and that this is different from a moving average. Thus, there are no overlap between data points within the different hours.

Correlation plots (not shown herein) indicate that the hourly averaging does not notably change the cross-correlation between the signals. It should be recognized, however, that temporal dependence and autocorrelations in time series data may introduce cross-correlations that are otherwise not present. The partial autocorrelation function of the individual data streams gives an indication of the temporal dependence and memory in the data (not shown herein). Such plots show that there is strong temporal dependence in the temperature signals, but less so or the engine speed. Temperatures display a memory of at least 10 minutes, but with residual serial correlation beyond this. Nevertheless, even though there are strong temporal dependencies in the data, in the cluster-based anomaly detection this time-dependence will be disregarded, and data-point will be

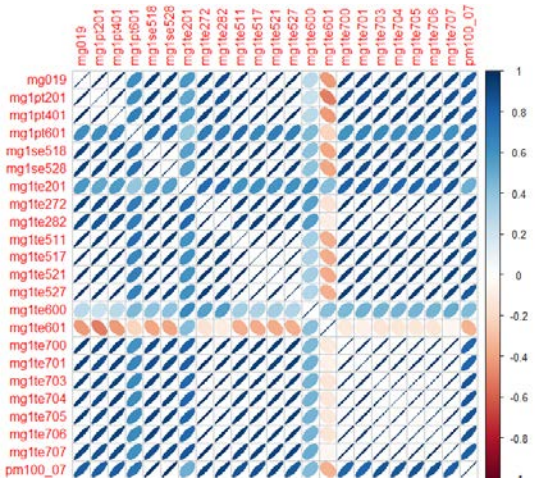

Fig. 2. Correlation plots for the training data

clustered individually without any regard of the sequence they arrive in.

The data for engine speed display two main modes of operation, with some transient states between these. This corresponds to engine being turned on or off, for example in a

load sharing scheme with the other generator engines. [4] shows various density plots illustrating that these main modes are reflected in the other sensor signals are show, but these are not reproduced herein.

In order to check how the clustering methods perform on the collected sensor signals, the data are divided into a training dataset and a test dataset. The training data are then used to identify clusters in the data, and the test data can then be assigned to one of these clusters. This should mimic new observations being collected. The underlying assumption is that the data naturally tend to cluster in a few clusters and that if new data arrives that are far from these clusters, this deviating behavior causes suspicion of faults in the system. In the exercise presented herein, the time-dependence in the signals are neglected and the training data consist of 75% of the original data randomly selected. The remaining 25% of the data constitute the test data. It is noted that randomly splitting the data into training- and test data is normally not recommended for time series data, but for the purpose of clustering the data when ignoring the serial correlation this can be defended. A correlation plot of the training data is shown in Fig. 2. It is noted that the linear correlation is almost identical in the test datasets.

A. Data Pre-Processing and Dimension Reduction

The original data for generator engine 1 is 24-dimensional, but one of the bearing temperature signals is missing so the remaining dataset contains 23 data streams. Although it is possible to perform clustering in this 23-dimensional space, one may hope to get better results if some form of dimension reduction is performed prior to the clustering. Hence, principle component analysis is performed on the training data to reduce the dimension. The same transformation can then be applied to the test data, transforming these data to the principal components identified from the training data. It is found that by keeping the first 7 principal components, 99.5% of the information in the data is conserved, and this is the number of principal components brought forward for further analysis.

It should be realized that even if the first principal components contain most of the information in the data, it might be that anomalies or changes in the data appear stronger in the least varying principal components. Hence, there is a possibility for missing important signals in the data when disregarding the last principal components, and this should be acknowledged. Nevertheless, clustering in higher dimensions are more challenging, so selecting the first 7 principal components and thereby effectively reducing the dimension from 23 to 7 is believed to be an important pre-processing step. Possibly, equally good results could have been obtained by selecting even fewer principal components.

When carrying out the principal component analysis, the input variables are normalized to have zero mean and unit standard deviation. Hence, no standardization is deemed necessary for the principal components before clustering.

III. Clustering Methods for Unsupervised Anomaly Detction

This section outlines the cluster analyses on the sensor signals and the subsequent application for anomaly detection. Various methods for clustering have been investigated, and different ways of using the various cluster methods for anomaly detection is explored.

A. K-means Clustering

The first clustering technique applied to the data are the standard K-means clustering algorithm (see e.g. [6]). This method requires that the number of clusters are specified and clusters the data into the specified number of clusters based on the squared Euclidean distance. Essentially, the method identifies K center-points and clusters the data around these in such a way that the distance between the data and the center-points within each cluster is minimized. This is done in an iterative way switching between finding the cluster centers and clustering the data around these in such a way that the overall within-cluster variance is minimized. In this study, different number of clusters are investigated and $K = 5$ is found to be a reasonable value. However, it is not straightforward to use these clustering in anomaly detection, and other clustering methods will be explored later that can also be used for anomaly detection.

B. Mixture of Gaussian Models

A mixture of Gaussian models can be used to estimate the density of the data, and this can also be used for clustering purposes. Compared to the *K*-means clustering algorithm, clustering with mixture of Gaussian models has two main advantages. First, a parametric model is fitted to the data, so it is possible to obtain density estimates and p-values for how likely the data are given the model. Moreover, since *K*-means clustering is based on the Euclidean distance, the clusters will be defined by hyperspheres around the cluster centers, whereas the Mixture of Gaussian models take the correlation into account and can give ellipsoid-shaped clusters of varying shapes and orientations.

The density of a Gaussian mixture model is on the form of a mix of *K* individual Gaussian densities, and the density function can be written as (see e.g. [6])

$$f(\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k \phi(\boldsymbol{x};\, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{1}$$

The π_k are the mixing proportions of the model determining the contribution from each of the *K* mixtures, and they should sum to one. The density of each mixture is described by the Gaussian density function, $\phi(\cdot)$ and each of the mixtures has a mean vector $\boldsymbol{\mu}_k$ and a covariance matrix $\boldsymbol{\Sigma}_k$. Fitting such a model to data means estimating the model parameters, $\hat{\pi}_k$, $\hat{\boldsymbol{\mu}}_k$ and $\hat{\boldsymbol{\Sigma}}_k$, and this is normally done using the maximum likelihood and the Expectation-Maximization (EM) algorithm. Having estimated a mixture model, it may provide an estimate for the probability than an observation, *i* belongs to a component, *l* as shown in eq. (2) and clustering may be performed by assigning the observation to the component with the highest probability.

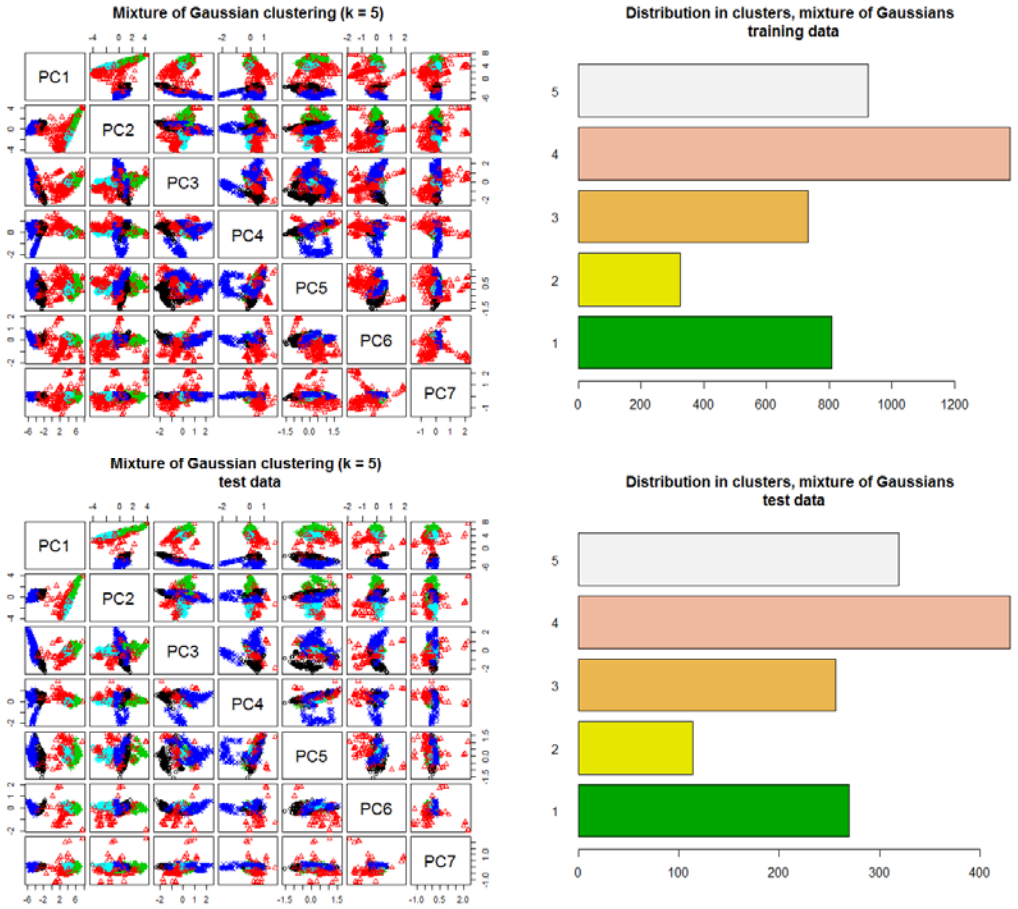

Fig 3. Data clustered in K=5 clusters with mixture of Gaussian model (left) and distribution of observations within the clusters (right); Clustering on training data (top) and applied to the test data (bottom)

$$\hat{p}_{il} = \frac{\hat{\pi}_l \phi(\boldsymbol{x};\, \hat{\boldsymbol{\mu}}_l, \hat{\boldsymbol{\Sigma}}_l)}{\sum_{k=1}^{K} \hat{\pi}_k \phi(\boldsymbol{x};\, \hat{\boldsymbol{\mu}}_k, \hat{\boldsymbol{\Sigma}}_k)} \tag{2}$$

Just as the K-means algorithm contains K centers, a Gaussian mixture model will contain K components with center points, and one of the tasks of fitting a mixture model to data is to determine the value of K. One way to do this is to calculate the Bayesian Information Criterion (BIC) and chose the model that maximizes this. Alternatively, the integrated complete likelihood (ICL) can be used for determining the optimal number of clusters in the data [7]. In this study, however, $K = 5$ will be assumed, corresponding to the number of clusters selected in the K-means clustering.

The results of clustering the training data by the mixture of Gaussian models and subsequently assigning the test data to these clusters are shown in the scatter- and bar-plots in Fig. 3. It reveals that the same clustering structure is observed in the training as in the test data, which is as expected.

1) Anomaly detection based on mixture of Gaussian clustering: Having established a Gaussian mixture model based on the training data, this could be used directly in condition monitoring and anomaly detection of new observations. Essentially, the implicit assumption is that the training data represent all nominal behaviour and operation of the system, and any new patterns in the sensor signals that are extreme according to the established model will be flagged as anomalous and possibly faulty. There are many ways of defining extremes within the context of mixture models, and in this study an anomaly will be defined based on some probability of being extreme according to the fitted model. With a mixture model, an observation can be regarded as an anomaly if it is extreme according to all the model components, and a p-value can be calculated for each Gaussian component separately. Then, an overall p-value could be the maximum p-value among the K components.

In the multivariate setting, there are different ways defining the probability of being extreme, see e.g. [8, 9], and in this study this is defined in terms of the Mahalanobis distance. In the d-variate normal case, the squared Mahalanobis distance will be distributed according to the χ^2-distribution with d degrees of freedom, and one may define the probability of being extreme, P_D, as the probability of having a squared Mahalanobis distance greater than what is observed. For an observation $\boldsymbol{x}_i$ with observed distance D_i one gets, $P_D(\boldsymbol{x}_i) = P_{\chi^2_d}(d \geq D_i^2)$. Hence, in the following, the P_D value is calculated for each observation based on every Gaussian component, $k = 1, \ldots, K$, of the mixture

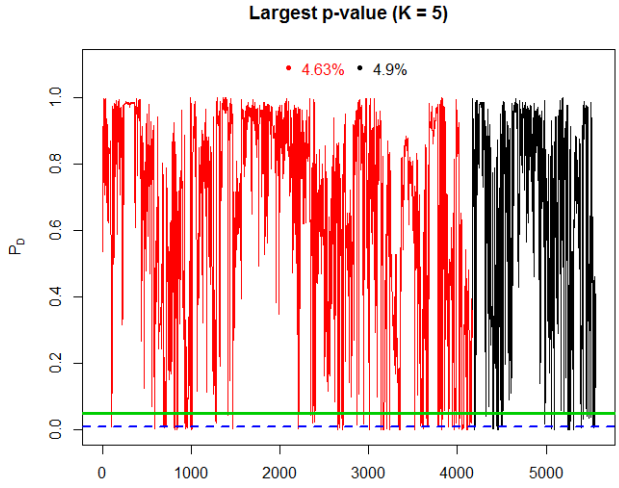

Fig. 4. Time series of maximum p-values used for anomaly detection

model and a *p*-value can be set as the largest of the K P_D -values, i.e.

$$p = \max_{1 \le k \le K} P_{D,k} \quad (3)$$

This value reflects the probability of being extreme in the component where the observation is least extreme. One may then use this for anomaly detection and flag any observations with small p-values as possibly anomalous. Typically, this means that the probability of being as or more extreme is low and the observation is an outlier. This corresponds to testing whether the observation belongs to the mixture component for which it is most likely to belong to and if one can reject the hypothesis that the observation belongs to this component, one may reject the overall hypothesis that the observation belongs to the mixture model and hence regard it as an anomaly. What remains is to choose a suitable α-level for the test.

The anomaly detection approach described above has been applied to the sensor data analyzed in this study, and each observation with $p < 0.05$ is initially regarded as an anomaly. Fig. 4 shows the time series of the largest *p*-values for both the training and the test data. It also reports the number of anomalies in the training and test data, respectively, for the mixture model with K=5. The solid horizontal line in the plot corresponds to $p = 0.05$. It is observed that approximately 4-5% of the observations are flagged as possibly anomalous. This agrees well with the 5% level of the test and could be expected even if

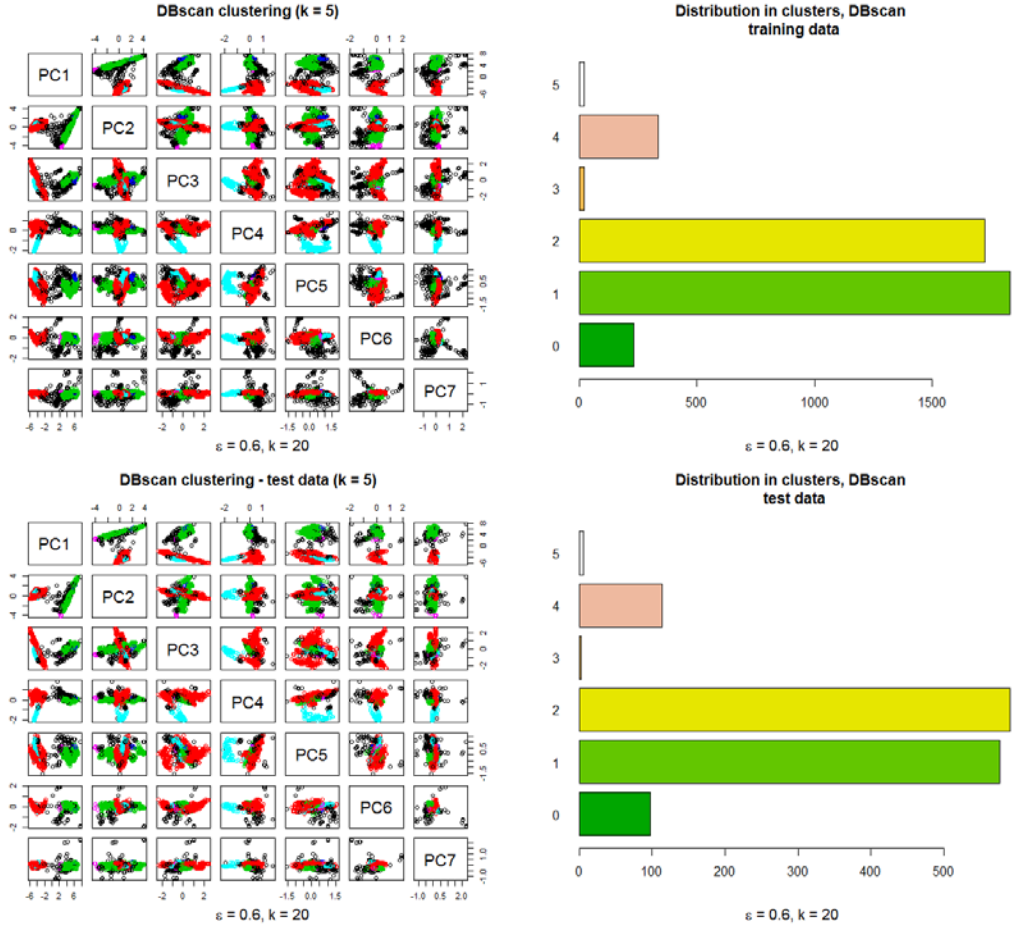

Fig. 5. Data clustered with DBscan (left) and distribution of observations within the clusters (right); Clustering on training data (top) and applied to the test data (bottom)

the mixture models were entirely correct and in the absence of any anomalies.

C. Density Based Clustering – DBscan

Another approach to clustering is based on the density of observations in the feature space and clusters observations with many neighboring points into clusters. DBSCAN is an algorithm for such clustering which will be explored in the following [10].

DBscan does not require the number of clusters to be specified. The number of clusters in the data will be determined by the algorithm. However, there is a need to specify two parameters, one determining the size of a neighborhood (ε) and the other specify the minimum number of core points that needs to be contained within a neighborhood for it to form a cluster, k. In principle, any distance function could be used, but in this application, the Euclidean distance will be assumed.

DBscan estimates the density around each data-point by counting the number of observations within the specified neighborhood size. It then distinguishes between core points, bordering points and a noise points. An observation will be regarded as a core point if at least the minimum number of points specified are within the specified distance, ε. Points that are within the neighborhood distance from a core point is said to be directly reachable from those core points. A point that is directly reachable from a core point, but with less than the minimum number of points within the neighborhood distance is referred to as a border point. A cluster is then formed based on all points that are reachable from a core point. Hence, each cluster must contain at least one core point and one or more border points. Points that are not reachable from any other points are regarded as outliers or noise-points. In this way, clusters may take any shape. This makes it possible to discover different types of clusters in the data than what can be found from the previous methods above.

One feature of the density based clustering algorithm is that it directly identifies outliers or noise points, and this may be used for anomaly detection. However, even though the number of clusters do not need to be specified, the resulting cluster structure will be highly dependent on the neighborhood distance, ε and the minimum number of points, k. Typically, these are not independent and it may not be straightforward to determine the optimal values of these parameters. As ε increases, the number of clusters decreases and the number of noise points decreases towards 0. One way to determine a reasonable value for the neighborhood distance ε is to look for a "knee"-point in a plot of the k-nearest neighbor distances. These plots display the sorted distances to the k-nearest neighbors for all data points. Such plots, for most values of k, indicate that reasonable values for ε might be in the range of 0.25 to 1, with perhaps longer distances for higher values of k.

It is often recommended to use domain knowledge to determine the value of k in DBscan, and it should be larger for large datasets than for small. In this paper, results for k=20 and ε = 0.6 are shown. These parameter choices yield clustering with 5 clusters. The resulting clusters are indicated in the scatter- and bar-plots in Fig. 5. Black circles represent the noise points that are not assigned to any cluster. It is interesting to observe that, compared to the K-means and mixture of Gaussian clustering, there is a more uneven distribution of points to the clusters. The number of noise points corresponds to anomaly rate of 5.55% and this is slightly higher than the ratio of anomalies detected by the mixture of Gaussian clustering above.

1) Anomaly detection with DBscan: Having applied DBscan to the training data and defined a set of clusters, new observations can be assigned to any of the clusters as they are collected. Observations that do not belong to any of the clusters will be regarded as noise points and can be regarded as possible anomalies. The results of predicting cluster membership on the test data are included in Fig. 5. Again the noise points are black circles. The distribution of observations assigned to the various clusters are also shown, and the distributions appear to be very similar to the distribution for the training data. The ratio of noise points in the test data, as detected by the DBscan clustering is 7.1%. This is slightly higher than for the training data but still seems reasonable.

D. Self-Organising Maps (SOM)

Self-organizing maps, also referred to as Kohonen maps, is a type of artificial neural networks that are used in unsupervised learning, and was proposed in [11]. They contain nodes, sometimes also referred to as neurons, and each node has a weight vector with the same dimension as the input data and is represented as a location on the map, which is usually 2-dimensional. Initially, the weight vectors of a map are set at random and then iteratively updated by feeding input vectors from the training data. For each training data point, the distance (typically Euclidean distance) to all weight vectors is computed and the node with the weight vector that is closes to the input data will be called the best matching unit (BMU). The weight vectors of the nodes in the neighborhood of the BMU will be updated by pulling them closer to the input vector (training data point). This is repeated iteratively for all training data and for a

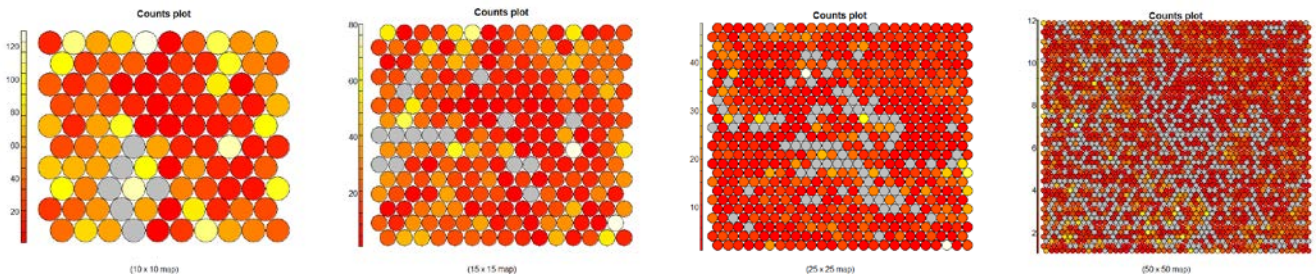

Fig. 6. Node count plots on the training data for different map sizes

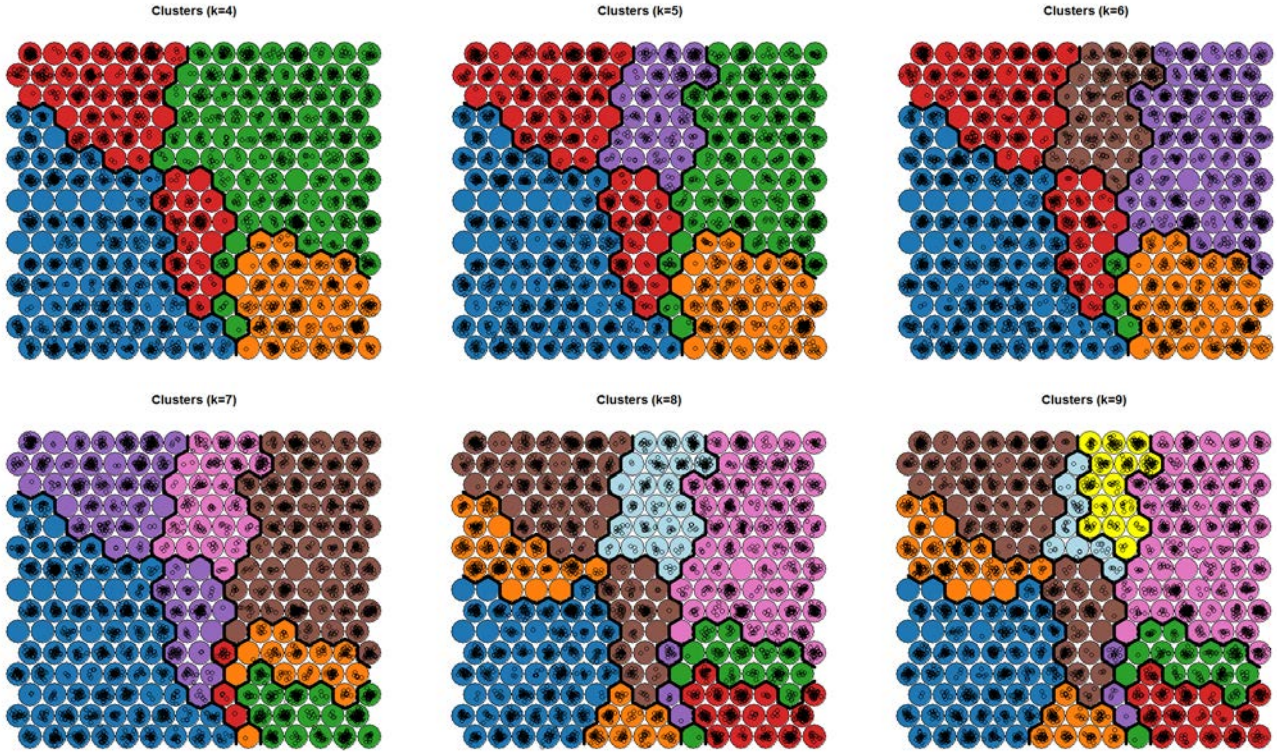

Fig. 7. Clustering of the self-organizing maps for different number of clusters; hierarchical clustering

specified number of iterations (cycles). The result of this is a map which associates output nodes with groups or patterns in the training data set. With a trained map, new observations can be mapped to the map by assigning input vectors to the node with weight vector closest to the input vector, the so-called winning node. The new input data will be assigned to this node, and hence to the corresponding location on the map.

When building a map on a set of training data, one must specify the dimensions of the map and the distance function used to calculate the distances. In this study, all maps are based on the squared Euclidean distances. Moreover, one must specify the number of cycles or number of times the training data should be sent to the network. To check whether a reasonable map has been specified, there are some diagnostics plots that may be made. One may plot the number of data points belonging to each node of the map to guide the selection of map size. Ideally, the distribution of counts should be relatively uniform over the map. Plots of node counts for four different map sizes are shown in Fig. 6, i.e. for a map with 10 × 10 nodes, 15 × 15 nodes, 25 × 25 nodes, 50 × 50 nodes. It is not obvious which of these should be preferred. Hence, selection of optimal map size is not straightforward, even though it seems obvious that the map with 50 × 50 nodes is too large, with quite many empty nodes and no nodes with more than 12 counts. Possibly, also the map with 25 × 25 nodes is too large.

Having selected a reasonable map size, one may check if enough cycles have been used in the training by looking at a plot of the changes between each iteration (not shown herein). This should reach a minimum plateau to indicate convergence. More cycles are typically required to train more complex maps. The largest map with 50 × 50 nodes does not seem to have converged even after 5000 cycles on the training data. However, both the maps with 10 × 10 and 15 × 15 nodes seem to stabilize and be fully trained before the end of 5000 cycles. In the remainder of this study, a map of 15 × 15 nodes will be assumed.

Clustering with self-organizing maps can be based on the distances to neighboring nodes. Typically, nodes with smaller distances to its neighbors will also be more similar to its neighbors compared with nodes with large distances to its neighbors. Hence, nodes with large neighbor distances may be natural borders between clusters of nodes. In order to determine the number of clusters in the map, one may again look at the between-clusters or the within-clusters sum of squares based on an initial clustering of the map nodes. This indicates that around 5 clusters are reasonable. The actual clustering of the map nodes will be performed by hierarchical clustering. and the resulting clustering of the map is illustrated in Fig. 8 for number of clusters k = 4, …, 9. These plots agrees well with a value of k=5. Going from k=5 to k=6 or k=7 does not introduce any main new clusters, but simply assigns separate clusters for a few bordering nodes, whereas k=8 and k=9 results in one new main cluster.

Having performed clustering on the self-organizing map, one may look at the cluster assignment for the training data and also predict the cluster membership on the test data. The resulting clustering, based on k=5 and a map with 15 × 15 nodes is shown in Fig. 9 for both the training and the test data, respectively.

1) Anomaly detection using self-organazing maps: There are different ways one could use clustering based on self-organizing maps for anomaly detection. For example, one could

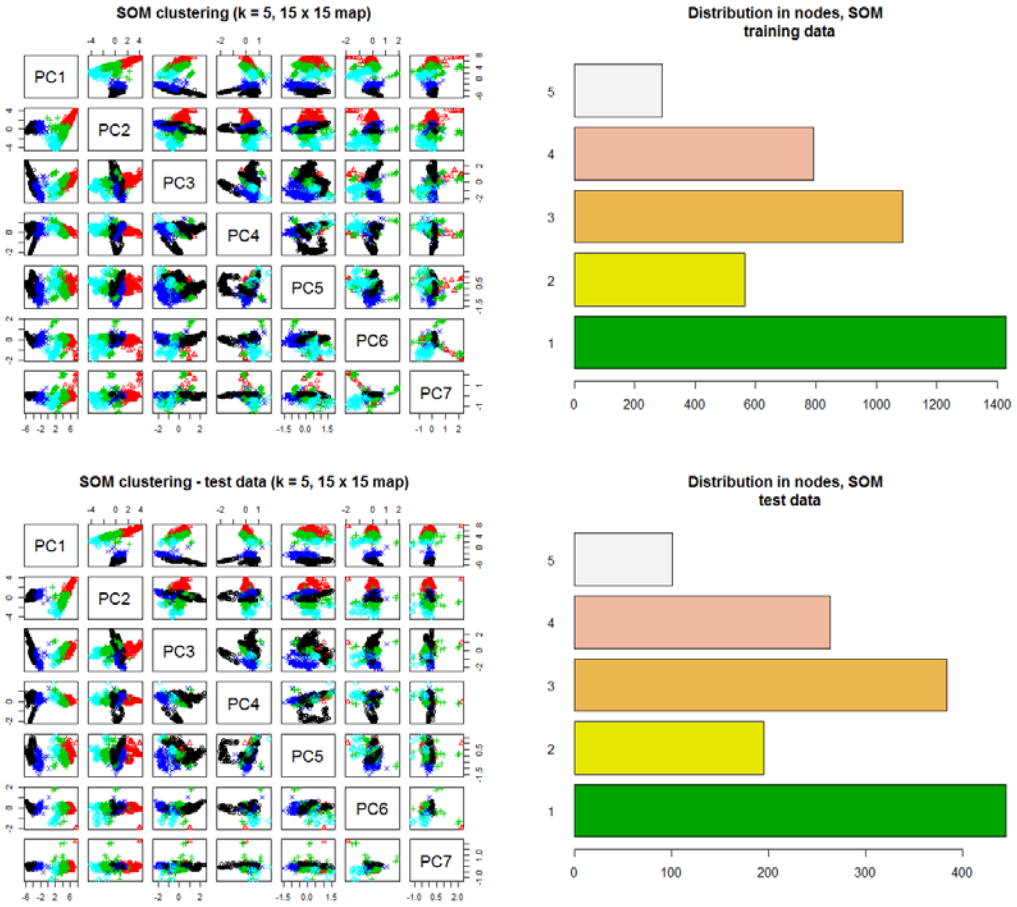

Fig. 8. Data clustered with Self-organizing maps (left) and distribution of observations within the clusters (right); Clustering on training data (top) and applied to the test data (bottom)

identify some nodes in the map as outliers and do anomaly detection similar to the scheme based on DBscan above. However, in this study, a somewhat different approach is investigated, based on signal reconstruction and residual analysis. This resemble more the anomaly detection approaches based on AAKR or DLM as reported in [5, 4].

Reconstruction of new observations based on a trained map consists of first mapping the new observation to a node in the trained map and then to predict the parameter values corresponding to that node. This will typically be the average of all the training data that belongs to the same node. Assuming that the map has been trained on anomaly-free data, large deviations of the reconstructed signal from the observed signal can be regarded as a possible anomaly. One must then either define a threshold for when a residual is construed as large, or one could apply a sequential test such as the sequential probability ratio test (SPRT) as outlined in e.g. [5, 4]. In this study, however, a simple threshold approach is taken, and a possible anomaly is flagged whenever the absolute value of the residual is larger than a predetermined threshold. For the purpose of this exercise, this threshold is set to ± 0.6 since it is observed that the prediction error is typically below this for the training data. This could be done on each sensor signal, or in this case, on each principal component. Trace plots of the test data and the predictions based on self-organizing maps as well as the residuals three first principal components are shown in Fig. 9. The threshold is indicated by a horizontal dashed line.

Applying such an anomaly detection approach on the ship sensor signal, one gets an anomaly rate of 0.58% on the training data and 1.66% on the test data. This is in general agreement with, but somewhat lower, than the anomaly ratios obtained by the mixture of Gaussian clustering and DBscan clustering presented above.

E. Spectral Clustering

Spectral clustering is based on the spectrum or eigenvalues of the similarity or affinity matrix and performs the clustering in a lower dimensional space [12]. Spectral clustering is known to perform well on identifying complicated cluster structures in the data such as spirals, doughnut-shaped or other non-convex clusters.

To calculate the affinity matrix, a measure of similarity needs to be specified in terms of a kernel function and an associate bandwidth parameter. In this study, a Gaussian radial basis kernel function is applied and the inverse kernel width is estimated to 3.60 based on 75% of the training data. In addition, one must specify the number of clusters to group the data in, and

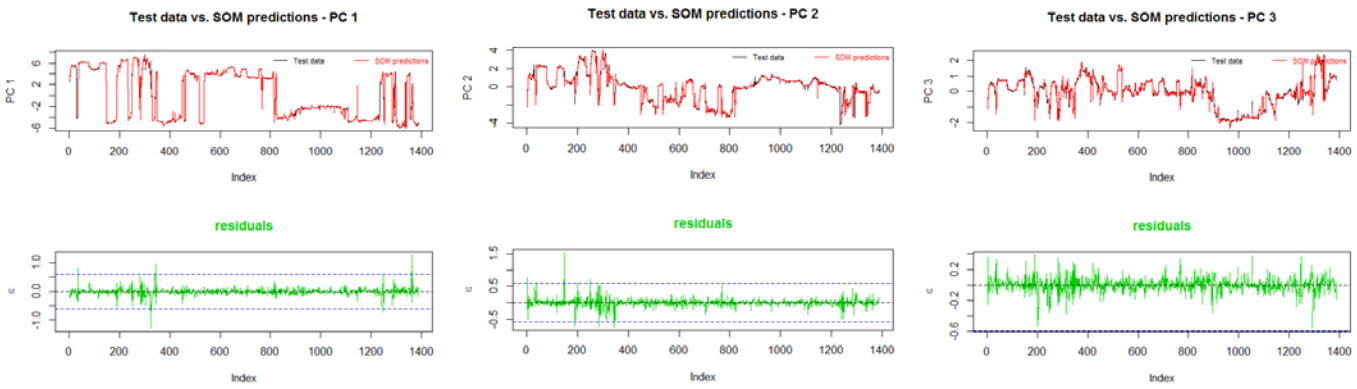

Fig. 9. Anomaly detection can be performed by studying the residuals between the observed signals and the ones predicted by the trained map; three first principal components.

in this study, K=5 is selected, as suggested by the analysis with K-means. It is noted, however, that this algorithm is initialized at random, so the actual results will also exhibit some randomness. Hence the exact results may not be reproduced if the algorithm is re-run several times.

The spectral clustering method is rather complex and the performance of this method deteriorates for large data sets. Indeed, the algorithm is very slow on the ship sensor data used in this study. In addition, spectral clustering is transductive and predictions of new observations based on the fitted spectral clustering model, i.e. the calculated affinity matrix is not straightforward. One would need to transform the new observations to the lower-dimensional space defined by the eigenvectors found in the training data and assign a new observation to the closest cluster based on the trained kernel function. There are methods for incremental updating of the spectral clustering in an online setting [13], but this is somewhat different from assigning new observations to clusters defined from the training data. Alternatively, one could simply assign new observations to the closest cluster center based on some distance measure. However, the spectral clustering approach to clustering is not brought forward to be applied for anomaly detection in this study.

IV. Discussion

The cluster-based anomaly detection methods presented in this paper are all found to perform reasonably well on the ship sensor data, and it has been demonstrated that they may be used at an early screening stage to flag anomalous sensor readings. By comparing the methods one finds that the results are slightly different for the different methods, but with a large degree of overlap. That is, most of the data points that are regarded as an anomaly by one of the methods is also regarded as anomalous by some or all of the others. Comparing all the methods, it is seen that the overall anomaly rate, as detected by any of the methods are 11.3% for the training data and 13.8% for the test data.

On way to get more robust anomaly detection could be to apply an ensemble of methods and disregard anomalies that are only detected by one method. This would presumably reduce the number of false alarms. For example, if applying 8 methods or variants (different parameters) and flagging an alarm only if two or more methods regards an observation as a possible anomaly, one would obtain anomaly ratios of 6.8% for the training data and 8.1% for the test data, respectively. If detection by three or more methods are required, the anomaly rates would reduce even further, to 3.8% for the training data and 4.8% for the test data. This seems to be reasonable anomaly rate for these data and could be a reasonable approach.

A detection scheme where each of the methods flag a possible anomaly only for a certain number of subsequent anomalous sensor readings is another scheme. For example, by requiring two subsequent anomalous observations to trigger an alarm, the anomaly rate for each individual method decreased notably. The overall anomaly rate from all the methods with this setup is 7.7% for the training data and 8.4% for the test data, respectively. This is considerably lower than the overall anomaly ratio as detected without requiring 2 subsequent anomalous readings, but perhaps still too high for these data. However, the methods could be combined to raise a flag only if a minimum number of the methods agree on a possible anomaly.

A. *Time-Dependencies in the Data*

The sensor data analyzed in this paper are essentially time series data, with temporal dependencies on various scales, both across different sensor streams and within the same signal. This means that the different signals are not only correlated at any given time, but also correlated to observations at different time lags. These temporal cross- and autocorrelations have not been taken into account in this study, and the observations are simply regarded as independent observations of the engine system. Presumably, there are much information in the temporal dependencies in the data that has not been utilized, and it may be that better and more robust detection strategies could have been developed if the time-dependence had been taken into account.

Moreover, time-series data should in general be treated differently from independent observations of a system. Not only do the temporal dependence in the data contain information that could be exploited to improve results of the analysis, but failure to correctly account for the temporal dependence could yield incorrect results. For example, it is a well-known fact that

autocorrelation in time series might influence the cross-correlation between time series.

The ordering of the data is meaningful in time-series. This influences how the data should be split in different parts, e.g. in a training and test set. In this study, this fact is ignored, and the splitting of the data into training and test data is done completely at random. It is acknowledged that this may influence the results. Splitting the data into two parts without accounting for the autocorrelation in the data will presumably give more similar training- and test-data than what would be the case if the data had been truly independent. This is also reflected in the results, where the clustering on the training- and the test data yielded very similar distribution of observations across the clusters. On the other hand, this ensures the representativeness of the training data compared to the test data, which is important for obtaining reasonable results.

The application of Dynamical Linear Models for anomaly detection, as presented in [14, 4] is an attempt to more explicitly exploit the time dependencies in the data in a condition monitoring system. However, such models are better suited for slowly varying time series, and the sudden jumps in the data, also in the training data for normal conditions, made it hard for such models to work well. However, DLM is a powerful framework that could be used on sensor data where the signals are typically varying slowly in time, and where sudden jumps or shifts represents anomalies. For the sensor data analyzed in this study, however, DLM turns out to not be an appropriate choice.

B. Importance of Representative Training Data

All data-driven approaches require representative training data to perform well. If the methods are trained on a dataset that is not representative of new observations, one cannot expect the methods to perform well on new data. In unsupervised anomaly detection, the implicit assumption is that the training data contain measurements of the system in all normal conditions, and that if new observations exhibit very different characteristics they will be regarded as anomalies, for example due to faults in the system or due to deviation from nominal operation of the system. Now, if measurements of some normal conditions are not included in the training data, future measurements under such conditions cannot be predicted and may be categorized as an anomaly even though it is perfectly normal. On the other hand, if the training data contain extensive measurements from a faulty or wrongly operated system, this would be regarded as normal and the method would fail to identify similar future measurements as anomalies.

In the study presented herein, anomaly detection is performed on sensor signals collected from a main generator engine onboard a ship in operation. Even though the data are time-series data, the splitting between training data and test data was done by randomly taking 75% of the observations to be the training data. As discussed above, this ignores the temporal ordering of the data and is not entirely correct for time series data, but it ensures that the training data is a good representation of the test data. To illustrate how important this is to achieve reasonable results, the mixture of Gaussian clustering method is repeated with the training data chosen to be the first 75% of the sensor measurements, whereas the last 25% are kept as the test data. In this case, the training data will be less similar to and thus less representative of the test data. The same pre-processing and dimension reduction is performed as outlined above. In this case, the anomaly rate in the training data is 3.75% and the rate in the test data is almost 75%. This is obviously too high, and is an effect of the training data not being representative for the test data. Similar results are obtained by repeating the DBscan and self-organizing maps methods.

This demonstrates the importance of having representative training data for doing unsupervised anomaly detection. However, it is not straightforward to obtain such a representative training data. Splitting the data into these two parts by random has been demonstrated to yield two subsets that are representative of one another. However, there is no guarantee that any of these subsets are representative of future observations of the system that has not yet been observed. This would be the case when one should employ anomaly detection in an actual condition monitoring system in real time. In that case, one would need to have training data that one could reasonably assume to be representative for *all* future measurements of the system, in that

- The training data contain observations corresponding to all possible nominal conditions in order to avoid false alarms
- The training data contain no observations from a faulty or wrongly operated system in order to avoid missed alarms

Obviously, one cannot ensure that these conditions are fully met, but one way to fulfil the first condition is to extend the coverage of the data used for training. In the case of ship monitoring systems, the training data should cover all operational modes and all environmental conditions the ship is believed to be operating in. This means that, ideally, training data covering several years of operation should be collected, to cover normal variations due to different seasons, different trades, different fuel quality, different operations, etc. To comply with the second condition, one may need to perform some cleaning of the data to reduce the amount of anomalous observations in the training data.

Notwithstanding these difficulties of obtaining a representative training data for use in online anomaly detection, the study presented herein demonstrates that different cluster methods can be used in different ways for anomaly detection on sensor data, and that the various methods perform reasonably well as long as the training data is representative of future measurements.

V. Summary and Conclusions

This paper has investigated the use of various cluster-based methods for truly unsupervised anomaly detection in condition monitoring based on sensor data. The study demonstrates that the cluster-based methods are useful and can be used for anomaly detection in different ways. The mixture of Gaussian approach can assign *p*-values to new observations according to how likely they are to appear according to the fitted model and flag anomalies accordingly. The density-based clustering method, DBscan, allows for certain data-points to be regarded as noise points, and these may be regarded as possibly anomalous in an anomaly detection setting. Self-organizing

maps can be used to reconstruct measured signals and anomaly detection can be performed on the residuals, in a manner similar to other anomaly detection techniques such as AAKR.

In isolation, all of the methods explored in this study perform reasonably well, with anomaly rates in the order of a few percent. However, it will presumably be possible to construct more robust anomaly detection schemes by combining an ensemble of methods, and by allowing individual spurious anomalous readings. Such systems would need to be fine-tuned to the application in question, but the analysis presented herein demonstrates that this may notably reduce the anomaly rate.

Compared to methods such as AAKR, the cluster-based approaches to anomaly detection is truly unsupervised in that it may handle situations where there are a few faulty data points in the training data. If the number of such faulty data points are smaller than what is needed to form a separate cluster, these anomalies in the training data may be discovered and will not influence the analysis of new observations. Nevertheless, as with all data-driven methods, it will be crucial to have representative training data to obtain reliable results.

ACKNOWLEDGMENT

The study presented in this paper is partly carried out within the centre for research-based innovation, BigInsight.

REFERENCES

[1] E. Vanem, "Statistical methods for condition monitoring systems", International Journal of Condition Monitoring, vol. 8, pp. 9-23, 2018.

[2] J.W. Hines and D.R. Garvey, "Development and application of fault detectability performance metrics for instrument calibration verification and anomaly detection", Journal of Pattern Recognition Research, vol. 1, pp. 2-15, 2006.

[3] J. Garvey, D. Garvey, r. Seibert and J.W. Hines, "Validation of on-line monitoring techniques to nuclear plant data", Nuclear Engineering and Technology, vol. 39, pp. 149-158, 2007.

[4] E. Vanem and G.O. Storvik, "Anomaly detection using dynamical linear models and sequential testing on a marine engine system", In Proc. Annual Conference of the Prognostocs and Health Management Society 2017 (PHM 2017), October 2017.

[5] A. Brandsæter, G. Manno, E. Vanem and I.K. Glad, "An application of sensor based anomaly detection in the maritime industry", in Proc. IEEE PHM2016, IEEE Reliability Society, June 2016.

[6] T. Hastie, R. Tibshirani and J. Friedman, "The Elements of Statistical Learning", Springer, 2nd edition, 2009.

[7] J.-P. Baudry, A.E. Raftery, G. Celeux, K. Lo and R. Gottardo, "Combining mixture components for clustering", Journal of Computational and Graphical Statistics, vol. 19, pp. 332-353, 2010.

[8] F. Serinaldi, "Dismissing return periods!", Stochastic Environmental Research and Risk Assessment, vol. 29, pp. 1179-1189, 2015.

[9] E. Vanem, "A simple approach to account for seasonality in the description of extreme ocean environments", Marine Systems & Ocean Technology, online first, 2018.

[10] E. Martin, H.-P. Kriegel, J. Sander and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise", in Proc. KDD-96, AAAI, August 1996.

[11] T. Kohonen, "Self-organized formation of topologically correct feature maps", Biological Cybernetics, vol. 43, pp. 59-69, 1982.

[12] A.Y. Ng, M.I. Jordan and Y. Weiss, "On spectral clustering: Analysis and an algorithm", in Proc. NIPS 2001, Neural Information Processing Systems, December 2001.

[13] H. Ning, W. Xu, Y. Chin, Y. Gong and T.S. Huang, "Incremental spectral clustering by efficiently updating the eigen-system", Pattern Recognition, vol. 43, pp. 113-127, 2010.

[14] E. Vanem, I.K. Glad and G.O. Storvik, "Dynamical linear models for condition monitoring with multivariate sensor data", in Proc. COMADEM 2016, COMADEM International, August 2016.