A Literature Review of Time-Series Clustering Techniques and Machine Learning Techniques Used for Monitoring of Wind Turbines

Written by
Yohann Jacob Sandvik

Abstract

Here the abstract will be.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Clustering is a method used to categorize big amounts of data into groups known as *clusters*, when there is little, or no information available about the underlying groups. It is a popular choice for extracting patterns from large datasets, because clustering falls within the category of *unsupervised machine learning*, meaning that it does not require the dataset to be labelled. Clustering is also a common step in data mining algorithms, where the goal is to learn rules relating the different variables in a dataset. In the last decade clustering has started to become more common to use on time-series datasets as they are abundant, and labeling is often costly and time-consuming. Time-series clustering has been applied on financial time series, medical time series and time series from a variety of other industries.

1.1 Motivation

As of 2018 wind power, together with solar power made up 7% of the world's electricity production,[1] and has been referred to as "the fastest growing source of energy" by the Norwegian company Statkraft[2]. As the effects of climate change steadily are becoming a reality, shifting to renewable energy sources is imperative, and wind power will play a bigger part in meeting the worlds energy demand in the future.

To make wind power as a whole more lucrative, a good start would be to reduce the downtime, and improve the performance of turbines. The argument that time-series clustering may be a good approach for this is two-fold.

1. A single wind turbine can have several sensors sampling very often, meaning that a wind farm can produce colossal amounts of time-series data. An unsupervised approach is usefull because labelling of all this data is requires a lot of time and resources.

2. When wind farms become big enough it will become too costly to manually inspect every turbine to construct an effective model for condition monitoring, further automation is required [1]. Time-series clustering is therefore a good alternative for condition monitoring.

[1] https://www.iea.org/geco/electricity/

[2] https://www.statkraft.com/globalassets/old-contains-the-old-folder-structure/documents/wind-power-aug-2010-eng_tcm9-11473.pdf

1.2 Objective

The literature review is meant to be a preliminary work for a master thesis where select techniques will be evaluated on actual time series data produced by a wind farm in Norway. To get a better understanding of wind turbines, and the challenges with using machine learning techniques for conddition monitoring of wind turbines, literature on this topic should be reviewed. The project assigment is also a continuation of the master thesis written in the spring of 2019 by Espen Waaga. In his thesis he explored the effectiveness of clustering raw time series in regards to similarity in time, and shape. In his "Future work" section Espen Waaga suggests that a natural next step would be to look at clustering with regards to similarity in change, by implementing a model-based approach. In addition, due to the quadratic time complexity of the dynamic time warping (DTW) algorithm it is inappropriate to use as a similarity measure for large datasets. Therefore, it could be interesting to see if a feature-based approach could be used to cluster time series with regard to similarity in shape with a lower time complexity than the DTW algorithm. This literature review has three objectives in the form of questions.

Objectives

1. What machine learning methods are currently being used to monitor the condition, and performance of wind turbines?
2. What are the different methods of feature-, and model-based time-series clustering currently used?
3. What time-series clustering methods (if any) are appropriate to test on time-series data produced by wind turbines?

1.3 Structure of Review

Chapter 2

Method

2.1 Search Terms

To find the relevant literature on the subjects of interest the search engine Oria was used to search the university library of the NTNU. Oria was preferred over other search engines such as Google Scholar because Oria allows one to combine multiple search terms in unison using "AND" or "OR", and because it allows the user to choose whether the search term should be in the title, subject, or other parts of the articles. Table 2.1 summarizes the search results. The *Title* and *General content* columns show which terms were used in the different searches; which terms where required to be in the title, and which terms could be in the "general content", meaning any part of the article. Let "$\times$" represent the AND operator between two search terms, and "$\wedge$" represent the OR operator. The "*" operator means that the search will include any word beginning with the word before the star, e.g *detect** includes *detection*, *detecting*, *detected*, etc. The N_f and N_i columns show how many results each search yielded, and how many articles from each search were included in the review, respectively. The method of choosing which articles to include is outlined in section 2.2.

Nr.	Title terms	General terms	N_r	N_i
1	wind $\times$ turbine* $\times$ (monitor* $\wedge$ detect*) $\times$ review	None	32	3
2	wind $\times$ turbine* $\times$ (monitor* $\wedge$ detect*)	machine $\times$ learning	100	47
3	time $\times$ series $\times$ clustering	None	219	46
		Total number of articles included		96

Table 2.1: Search results

Search 1 and 2 are used to find articles covering the first objective mentioned in section 1.2, and search 3 is ment to cover the second objective.

Criteria	Reason
Primary goal is time-series prediction or time-series forecasting.	It is outside the scope of this assignment.
Uses subsequence time-series clustering methods.	It is outside the scope of this assignment.
Time-series clustering is used only as a minor preprocessing step.	Not considered relevant enough to the objectives of the review.
Data used is not time series data.	Not considered relevant enough to the objectives of the review.
Paper does not actually use clustering algorithms.	Not considered relevant enough to the objectives of the review.
The data used consists of image time series.	Data to different from data to be used in master thesis.
The time-series clustering methods explored in the work are **not** model-based or feature-based.	The raw-data-based approach has been somewhat covered in Espen Waaga's master thesis, hence it is emmitted in this review.
The specific model-based approach using the tail dependence of time series.	Method not found relevant enough for the data that will be used in the master thesis, more relevant for financial time series.

Table 2.2: Exclusion criteria for articles in search 3

2.2 Screening Method

To make sure that the articles used were relevant, the review is limited to articles published in peer-reviewd journals, after January 2014. There were three levels of screening, screening of the title, abstract, and full article. Title-screening was primarily for seeding out duplicate articles returned from the search-engine. The screening of the abstract and full-article were to identify the articles that were not relevant for the review and exclude them. It has been a challenge to include enough literature to meet the objectives of the review, but also not more literature the author alone could handle. So although the objectives is to get an overview of the different time series clustering methods, and an overview of the current machine learning methods used for monitoring wind turbines, the author does not claim to have made a complete exhaustive summary of all the possible methods.

Search number 1 was used to find existing literature reviews on condition monitoring of wind turbines. Three good literature reviews on the subject where found, and one good review on machine learning methods used for condition monitoring of wind turbines was found in search 2. So, when screening the remaining articles from search 3 the focus was to find articles not included in the aforementioned reviews, to complement them as well as possible. When screening articles from search 3 articles meeting one (or more) of the criteria outlined in table 2.2 were excluded from the review.

Chapter 3

Wind Turbine Monitoring

In this chapter a brief overview of which components make up a wind turbine will be given, and how a wind turbine functions. Further, the different machine learning techniques enountered will be shown, and the theory behind the most popular models will be explained. The chapter will end with a discussen about the advantages of the different models in relation to what kind of data one has available.

3.1 Wind turbine Components

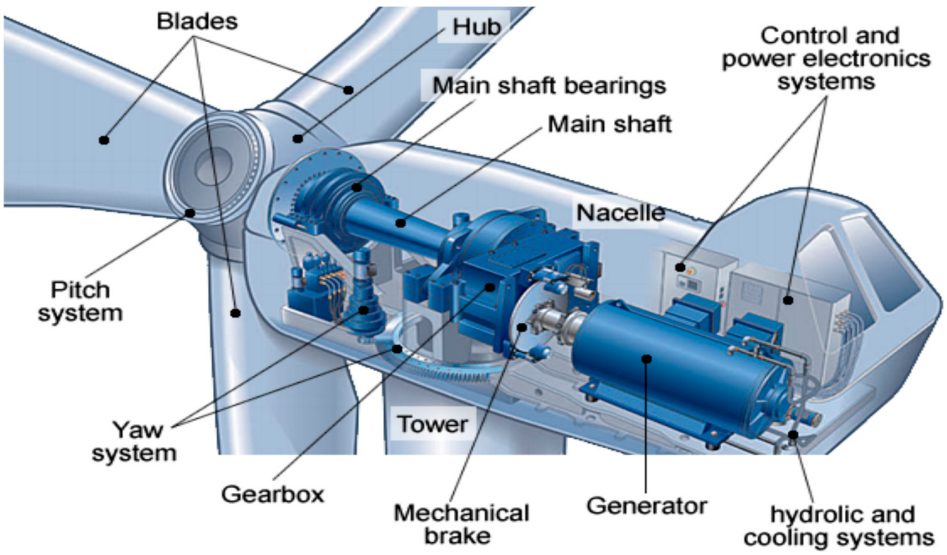

Figure 3.1: Illustration of the different parts of a wind turbine, taken from Hossain, Abu-Siada, and Muyeen [2]

Figure 3.1 shows the main parts of a wind turbine which includes the rotor (blades and hub), shafts, gearbox and generator. Simplified a wind turbine works by wind pushing the blades, generating torque that makes the hub rotate. The hub is connected to the gearboxjkk through the main shaft. The gearbox then gears down the torque and gears up the rotational speed to a level that the generator can use to induce current, that goes to a station that transforms the voltage to a level that can be used in the electrical grid.

3.2 Sensors and Data Acquisition

Information about a wind turbine can come from many sources, it can come from external sources such as images from a camera, or from internal sensors measuring operational data. The collective term for systems measuring operational data is supervisory control and data acquisition (SCADA) systems. To choose what algorithm to use, or what model to use, one must first consider what data one has available. From the literature considered, these where the most used forms of data used as input for the model:

- Vibration measurement
- Acoustic emission monitoring
- Temperature measurement
- Power signal measurement
- Oil debris monitoring
- Strain monitoring
- Optical fiber monitoring
- Ultrasonic testing
- Image analysis

Analysis of vibration signals is the most common form of condition monitoring used in industry for any form of rotating equipment [3]. By measuring the acoustic emission generated by a component of a wind turbine, one can estimate how much damage it has sustained. The temperature of components in a wind turbine is closely correlated with the health of the component, and is therefore used often in condition monitoring applications [4]. The power signal can also say a lot about how wind turbine is performing, specifically the wind speed - power curve. When monitoring the debris in the oil of a wind turbine gearbox one is analysing the size, type and number of wear particles present in the lubricant, as they can indicate the degree of damage in the gearbox [5]. Strain monitoring, optical fiber monitoring, ultrasonic testing, and image analysis are all used to detect structural damage in different components of the wind turbine, usually the blades [6–11], or tower [12]. However, the most common approach was to use a combination of multiple sensor-values to make predictions about the condition about the wind turbine.

3.3 Machine Learning Techniques

A machine learning is a subset of artificial intelligence. Machine learning models extract rules from data, which can then be applied to classify, or estimate components of another dataset. Machine learning algorithms are formally divided into *supervised learning*, *unsupervised learning* and *semi-supervised learning*. Supervised learning models require labelled datasets to extract information from the dataset, and are usually used to perform classification tasks, or to estimate a variable that is considered dependent on the input variables (regression). Unsupervised learning algorithms do not require labelled datasets. Semi-supervised learning uses a combination of labelled and unlabelled datasets. The different machine learning models encountered are split into regression-based models, supervised classification-based models and unsupervised classification-based models.

Extraction method	Articles
ARMA models	[6, 9, 13–15]
Discrete Wavelet Transform (DWT)	[14, 16–18]
Principal Component analysis (PCA)	[6, 9, 11, 18–21]
Basic signal statistics	[18, 22, 23]

Table 3.1: Feature extraction methods

3.3.1 Feature Extraction

There are two central problems in condition monitoring that can be solved by feature extraction and selection. The first is the sheer volume of information being produced. A wind turbine with only 20 sensors, sampled at 100 Hz will produce 170 MB of information per day. Feature selection is used here to reduce the number of features to only those relevant for condition monitoring. The second problem is that for systems using only one signal such as vibration, there are many components that are superposed to create the measured signal, and noise is present. Feature extraction is used to separate the interesting components from each other, and cancel the noise. For some machine learning models feature extraction is not a neccesary preprocessing step. For others, careful thought must be given as to how to extract features. Table 3.1 shows the most frequent methods found in the articles.

The DWT is a method used for decomposing time-dependent signals. Compared to the discrete Fourier transform (DFT) the DWT can capture time-dependent and frequency-dependent information of a time series, which makes it more suitable for non-stationary signals. However, the DFT is also used [14, 22]. Basic signal statistics refers to values easy to calculate over a fixed window size, such as the root-mean-square (RMS) value and the min/max value ARMA models are frequently used for time signal analysis, and will be expanded upon in section 4.2. PCA is an unsupervised machine learning form used in multivariate systems. It produces the linear combinations of variables that has the highest variance, also called the *principal components*. PCA, and generalizations of PCA is also a valuable tool for dimensionality reduction, and will also be expanded upon, in section 4.2. Before going into which machine learning models have been used for condition monitoring, an extended explanation will be given of some of the more complex, and most used machine learning models: Artificial neural networks (ANNs), networks of restricted Boltzman machines (RBMs) and support vector machines (SVMs). A brief explanation will be given about the other models encountered as well.

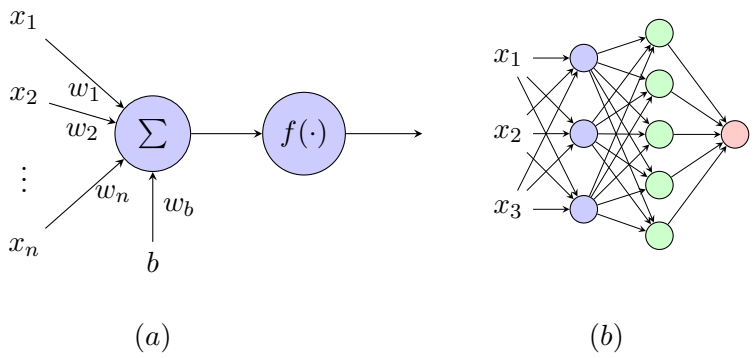

Figure 3.2: (a) A Perceptron. (b) Example of a simple ANN.

Artificial Neural Networks

Figure 3.2 (a) depicts the building blocks of an ANN, the perceptron. The perceptron is a model of an artificial neuron, it takes in n inputs, performs a weighted sum of the inputs and a bias b, and sends the sum through an objective function. Originally the objective functions were just threshold functions returning 1 if the sum was above the threshold, and 0 if the sum is below the threshold. When one started using multiple layers of perceptrons as ANNs, other objective functions where introduced including *tanh*, *sigmoid* and the *recitfied linear unit*. The perceptron is only able to perform binary classification on points that are linearly separable. However, by combining multiple layers of perceptrons into networks, training the network with the backpropagation algorithm and by intruducing non-linear objective functions ANNs are able to capture complex non-linear relationships. A simple ANN is depicted in figure 3.2 (b). The first layer in an ANN is called the input layer, the last layer is called the output layer and all the layers inbetween are called hidden layers. ANNs using special perceptrons with "memory" also exist, and are called recurrent neural networks (RNNs). Some ANNs use special layers that only perform wieghted sums with the closest set of neurons from the previous layer, such networks are called convolutional neural networks (CNNs). If one designs an ANN to have as many perceptrons in the output layer as in the input they can be used as a generative model to recreate the input, this is what is known as an autoencoder.

Support Vector Machines

SVMs were originally implemented as a type of binary classifier that could classify linearly separable variables. The SVM transforms input variables to a set of hyperplanes, which can be used both for classification and regression tasks [24]. However, by introducing kernel functions one became able to map the input data to a vector space where they are linearly separable. Some example of kernel functions include: Linear kernal, radial basis function and sigmoid function.

Restricted Boltzman Machines

RBMs can be considered as a stochastic neural network with two layers, an input layer and a hidden layer [4], note that they do not have an output layer. The defining difference between RBMs and ANNs is that ANNs are deterministic models, while the units that make up an RBM are inherently stochastic. RBMs alone are said to be generative models [25], like the autoencoder. If one combines multiple RBMs, such that the hidden layer of one RBM feeds in to the input layer of another, and combines this with a final output layer one has a deep belief network (BDN). DBNs can perform discriminative tasks such as classification and regression,

and are not solely generative as the RBMs alone.

Other models

Gaussian processes (GPs) are a probabilistic machine learning technique that can be used for regression, and classification tasks. Similar to SVMs they perform best when the relationship between the input variables and the target variable are linear, but by the use of kernels one can map the input variables to a hyperplane where the targets are linearly seperable. [26] The defining diference are that GPs are probabilistic while SVMs are deterministic. K-nearest neighbors is a machine learning model that can be used for classification and for regression. When used for regression the target is predicted based on the points from the training set which are "nearest" in terms of input variable values [27]. Random forests (RF) when used for regression tasks randomly create regressors which each give a prediction of the target variable [27]. Through the training process the model learns which regressors should have more weight in the final prediction.

3.3.2 Regression-based Models

A frequent approach is to use a supervised learning model to capture the normal behaviour of a wind turbine, or wind turbine component, by predicting the value of one time dependent variable, or by recreating the input variables. Then the deviation between the prediction and the actual value is used to detect anomalous behaviour. If the system attempts to predict one variable this deviation is called the estimation error (E_e), if the system is intended to recreate the input the deviation is called the reconstruction error (R_e). The input data used in this approach is most often SCADA data. What varies in these approaches is what machine learning model they use to model the wind turbine. Three curves in particular hold a lot of information about the performance of a wind turbine, namely the wind speed, active power curve; wind speed, rotor speed curve and wind speed, blade angle pitch curve. The majority of the regression based models used a subset of these curves as target variables [27–33]. Another popular approach is to attempt to forecast the temperature in the gearbox, or generator windings [4, 33–37]. As mentioned before temperature is closely correlated with the health of a component, but since temperature changes so slowly it is hard to use temperature monitoring alone for fault prediction [12]. However, if one can make a model capture the complex sources of temperature change, the deviation between predicted temperature, and actual temperature could be used for fault prediction. Table 3.2 shows the typical machine learning models used for regression.

Machine learning model	Articles
ANNs	[31, 33, 35–40]
GP regression	[28, 29]
SVM	[27, 30, 32]
PCA	[20]
KNN	[27]
RF regression	[27]
Network of RBMs	[4, 34]

Table 3.2: Machine learning algorithms used by normal behaviour models

Straczkiewicz and Barszcz [38] train an ANN to estimate the vibration in a gearbox using gearbox oil temperature, wind speed, rotor speed and active power as an input. By performing linear regression on E_e of the ANN they are able to detect early states of damage in a gearbox,

months before it is replaced. Rodriguez-Lopez et al. [35] compare different ANNs using SCADA data from 14 wind turbines monitored over several years, trained to estimate the temperature of the bearing on the non-drive end of a generator. They are able to detect failueres within 2 months of occurance. Mazidi et al. [33] propose a model of three different ANNs that estimate the rotor speed, gearbox temperature, and generator winding temperature using SCADA data. E_e for the ANNs are then forwarded to a proportional hazard model which sets a dynamic threshold for what is considered anomalous behaviour. Qian et al. [40] propose an ANN to predict the value of gearbox oil, and bearing temperature, and use E_e to indicate faults. To ease the computational load of the local machines their model is implemented on a cloud platform. From two case studies their model was able to detect anomolous behavior in the reconstruction error that was due to an error in the cooling system. NN are by far one of the best models to capture complex non-linear relationships between input variables, and in contrast to the kernel methods such as GP and SVM they are less relient on data being preprocessed, and features being extracted from the data beforehand. Instead what features a ANN is able to extract is decided by the size, and complexity of the archtitecture. One of the disadvantages of using ANNs is that they require a lot of training data to be accurate compared to other machine learning models. The amount of training data required by a ANN is also decided by the complexity of it's architecture. Gonzalez et al. [27] chooses not to use ANNs in their approach because they are prown to overfitting, instead they compare three other approaches of SVM, KNN and RF to estimate the active power. They found that the RF regressor trained with high frequency SCADA data produced the best results when trained on short periods. Tao et al. [30] use Grey correlation algorithm for eigenvector extraction and genetic algorithms for feature selection and an SVM to estimate all three performance curves. Wang et al. [4] use a DBN that consists of two layers of RBMs, and an output layer. They uses the DBN to predict the gearbox main bearing temperature, and sets a threshold for E_e to detect anomalies. Zhao et al. [34] also uses a network of RBMs, but use them an autoencoder, and then uses R_e to detect an anomaly. DBN have a great potential for both regression and classification tasks, but as with ANN their performance is highly dependent on their architecture, and the data used to train them [4]. Yang, Huang, and Yang [39] use an auto-associative ANN as an autoencoder, and then use the Hotel T_2 statistic as a dynamic threshold for R_e.

3.3.3 Supervised Classification-based Models

Machine learning model	Articles
ANN based	[7, 8, 10, 41–43]
Tests several individual classifiers	[6, 13, 15, 16, 44]
Uses fusion / ensamble of classifiers	[9, 14, 45]
SVM	[17, 18, 22, 23, 46, 47]
DBN	[48]
Hidden Markov model (HMM)	[49]

Table 3.3: Machine learning algorithms used by supervised classification models

Shihavuddin et al. [7], Qiu et al. [8], Zhang, Lu, and Wang [10], and Reddy et al. [41] use images as input, and different types of ANN for classifying structural damage in the blades. Chen et al. [43] uses SCADA data as input, and deep ANN for detecting ice on the blades. The testing, and comparing of several individual classifiers was also frequently used in the papers considered. Since there are so many different classifiers that are used, explaining all of them is outside the scope of this assignment. The reader is referred to [6, 13, 15, 16, 44] to read more about them.

However, the most popular classifier model by far is the SVM. The SVM is used for detecting damage in the blades [22], general fault detection in the turbine [47], but most for gearbox faults [17, 18, 23, 46]. Kim and Yun-Ho [49] use a statistical model based on HMMs. They combine multiple vibration signals measured by a condition monitoring system with thresholds, and set different alarm levels. These alarm levels act as the observations for the HMM. The states of the model are *Normal* and *Fault*. The model is trained with sequences of observations, which it uses to determine which state is most probable of producing said sequence of observations. The theory behind an HMM will be expanded upon in section 4.2.

3.3.4 Unsupervised Classification-based Models

The classification models have been split into supervised and unsupervised classifiers because the unsupervised methods are of greater interest for this review. The use of unsupervised learning methods are not as widespread as the use of supervised learning methods. Stetco et al. [50] only included one article in their review that compared a regression model based on feed forward neural networks to two unsupervised models using gaussian mixture models and self organizing maps. It should be noted that the articles using an unsupervised learning approach that are included in this review generally were published after Stetco et al. [50].

Machine learning model	Articles
RBMs	[51]
K-means clustering	[21]
One-class SVM (OCSVM)	[11]
Multiway PCA	[19]

Table 3.4: Machine learning algorithms used by unsupervised classification models

Zhang and Ma [21] is the only implementation found using time-series clustering used for condition monitoring of wind turbines. They first use parallell factor analysis (PARAFAC) for dimensionality reduction, which is a generalisation of PCA, and then use K-means clustering on the reduced feature space. In their approach they are able to identify distinct operation modes of the wind turbines, and reduce data redundancy by PARAFAC. However, they mention that there is still further work that can be done, especially in testing their model with data from turbines operating at different wind speeds. Yang, Liu, and Jiang [51] use a spatiotemporal pattern network for feature extraction, and then use unsupervised stacked RBMs for anomaly detection. Wang et al. [11] uses images as input, and the hidden layers of a CNN trained on an unrelated dataset to extract features, and then compresses the features using PCA. The compressed features are then fed into an OCSVM which classifies faults. Pozo, Vidal, and Salgado [19] uses multiway PCA to set up a baseline model, and then uses hypothesis testing to determine whether the power curve of a wind turbine is an anomaly or not. A summary of the articles related to machine learning methods for condition monitoring of wind turbines can be found in 8.2.

3.4 Discussion

Table 3.5 summarizes the advantages and disadvantages of the different machine learning models in terms of accuracy, complexity, interpretability and amount of training data required. As can be expected to acheive high accuracy the complexity of the model increases, and there is a trade-off between complexity and interpretability. The supervised classification models seem to

Model	Potential accuracy	Complexity	Interpretability	Training data required
ANN	High	High	Low	High
SVM	Medium - high	Medium	Medium	Medium
RBMs	High	High	Low	High
DBN	High	High	Low	High
PCA	Medium	Low	High	Low
HMM	Medium	Medium	Medium	Medium
KNN	Low	Low	High	Low
RF	Medium - high	Medium	Low	Low

Table 3.5: Summary of advantages of different machine learning methods

be the most popular ones, however regression based models are a close second. The supervised classification models show great promise in terms of accuracy, but are not of that much interest for this review, since the wind turbine data that will be used in the spring is not labelled with faults occurance, or anomalous behaviour. The regression based models do not require explicit labeling to model normal behaviour, so they could be used to detect some anomalous behaviour. What is problematic with using a complex regression based model such as an ANN or DBN, is that without labelled data it might be hard to validate the performance of said model, because it often hard to determine why a particular observation is regarded as an anomaly. However, what can be transferred from these papers are the different methods for feature extraction, and selection. What is preferable with ARMA models, HMMs, and PCA compared to ANNs and DBNs is that they are easier to interpret, which is valuable when using unlabelled data.

It is promising to see that K-means clustering paired with PARAFAC showed such good results. Since the model developed by Zhang and Ma [21] still needs to be tested on wind turbines in more variable conditions (in terms of wind speed), there is still room for exploration on this topic. The use of RBMs and OCSVM for unsupervised anomaly detection is very interesting, but suffers the same problem as regression based models: without labelled data it is hard to validate the models, and interprate the anomalies they detect. The work done by Pozo, Vidal, and Salgado [19] strengthens the argument for exploring PCA, and generalizations of PCA for feature extraction, and selection.

Chapter 4

Time Series Clustering

This chapter first gives a definition of a time series that will be used throughout the paper, it goes on to give an overview of the different time series clustering approaches and finally discusses the different representation methods, and clustering algorithms encountered in the literature.

What is a time series

In this report we will deal with *discrete time series*. A time series is defined as a set of observations $\{x_t\}$ recorded at a specific time t. A discrete time series is a time series where the set of times when observations are made (T_0) is discrete [52]. A multivariate time series can be viewed as a set of vectors $\{\mathbf{x}_t\}$ where each set of vector elements $\{x_t^i\}$ is an individual time series. This means that the elements of the same vector $[x_t^1, x_t^2, ..., x_t^N]$ are separate observations made at the same time instance t. In a wind turbine, measurements of the temperature in the gear box made every 10 seconds can be considered a univariate time series. While the set of measurements made every 10 seconds of the temperature in the gear box, the power produced by the turbine, and the wind speed ahead of the blades can be considered a multivariate time series.

4.1 Overview of Time Series Clustering

There are three types of TSC, *whole-series TSC*, *subsequence TSC* and *time-point TSC*. Whole-series TSC is when multiple "whole" time series are clustered with respect to each other. Subsequence TSC comprises the clustering of subsequences of the same time series with respect to each other. The defining difference between whole-series and subsequence TSC is that whole-series TSC clusters multiple time series while subsequence TSC clusters clusters different subsequences of the same time series. When performing time-point TSC the goal is to cluster individual observations of a time series wrt. to each other. In this review we will only consider work using whole-series TSC, so when the phrase *time-series clustering* is used, one can assume that whole-series TSC is what is being refered to.

Whole series TSC can broadly be divided into three main approaches. The raw-data based approach, the feature-based approach and the model based approach. In the raw-data based approach one measures the similarity between the raw time series themselves and clusters them based on this. When clustering raw time series the majority of the work goes into selection of similarity metric and clustering algorithm, and one clusters the time series with regard to similarity in time or similarity in shape [53]. In the feature-based approach one also clusters time series with regard to similarity in time, and shape, but the work is somewhat shifted away

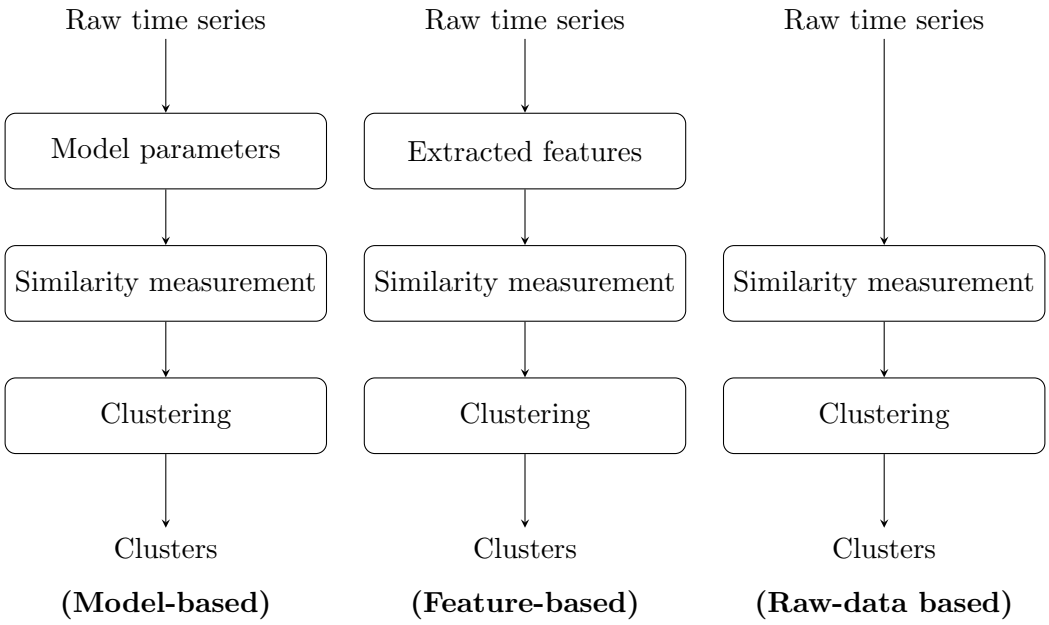

Figure 4.1: Illustration of the three approaches to TSC, and their components. The illustration is inspired by figure 2 in Aghabozorgi, Shirkhorshidi, and Wah [53]
.

from choice of similarity metric and over to choice of representation. Either to extract more relevant information from the time series, or to reduce the computational complexity of the similarity measurement. In the model-based approach the goal most often to cluster time series with regard to the underlying data generating process [54]. The underlying assumption being that two time series that appear different might still have been generated by the same process. The common denominator of the three approaches to TSC mentioned is that they are all made up of three distinct parts: representation method, similarity metric and clustering algorithm. This is illustrated in figure 4.1. The choice of representation is key, and is the task of retaining the valuable information while disregarding the irrelevant information. When calculating the similarity between all combinations of time series one is clustering, the resulting similarity measures are stored in what is called a *dissimilarity matrix*. The choice of similarity metric is important in a raw-data approach as it decides which aspects of the time series will be used to measure (dis)similarity, and has a has a significant impact on the time-complexity of the clustering system. However the requirements of similarity measure are somewhat relaxed in the feature-based and model-based approaches as the representation method already limits the number of aspects which one can compare time series with, so many of the same similarity measures are used.

4.2 Time Series Models and Representations

To select suitable mathematical models for a dataset, we have to allow for the random nature of future observations. This is done by assuming that each observation in a time series x_t is a realization of a particular random variable X_t. The time series can then be modelled as a collection/set of random variables $\{X_t\}$, also known as a *stochastic process* [52]. In the following subsections the theory of the three most prevalant will be presented: ARMA models, HMM and PCA.

4.2.1 Autoregressive Moving Average Model

To define an ARMA model, one needs to have a clear understanding of the terms white noise process, and stationary process. We say that the stochastic process $\{Z_t\}$ is "white noise" with zero mean, and variance σ^2 ($\{Z_t\} \sim WN(0, \sigma^2)$) if and only if $\{Z_t\}$ is zero mean, and every random variable contained in $\{Z_t\}$ is uncorrelated with every other random variable contained in $\{Z_t\}$. A stochastic process $\{X_t\}$ is said to be weakly wide-sense stationary if the mean, and variance are constant for all terms in the process. $\{X_t\}$ is said to be weakly short-term stationary if the mean and variance of terms are constant for distinct time periods within the duration of the process, but are not constant for all terms in the process. For brevity the term "stationary process" will be used when refering to a *weakly wide-sense stationary process*. An ARMA model descirbes a time series in terms of difference equations. It can be considered a combination of two smaller models, an autoregressive (AR) model and a moving average (MA) model. Let $\{X_t\}$ be a stationary process. An MA(q) model will describe every term X_t as a linear combination of q distinct white noise terms as in equation (4.1).

$$X_t = Z_t + \theta_1 Z_{t-1} + ... + \theta_q Z_{t-q} \tag{4.1}$$

Whereas an AR(p) model will describe every term X_t as a linear combination of p previous terms of $\{X_t\}$ as in equation (4.2)

$$X_t = \phi_1 X_{t-1} + ... + \phi_p X_{t-p} \tag{4.2}$$

Putting equations (4.2) and (4.1) together, an ARMA(p, q) model will describe every term X_t as a linear combination of p previous terms, and q white noise terms as in equation (4.3).

$$X_t - \phi_1 X_{t-1} - ... - \phi_p X_{t-p} = Z_t + \theta_1 Z_{t-1} + ... + \theta_q Z_{t-q} \tag{4.3}$$

Given that the polynomials $1 + \theta_1 z + \theta_2 z^2 + ... + \theta_q z^q$ and $1 - \phi_1 z - \phi_2 z^2 - ... - \phi_p z^p$ have no common factors [55]. When modelling a time series with an ARMA model, one usually attempts to decompose the time series into a set of components, as shown in equation (4.4). Here X_t is the observed time series, r_t is called the trend component, s_t is called the seasonal component, c_t is called the cyclical component and ϵ_t is called the innovations or residual component.

$$X_t = r_t + s_t + c_t + \epsilon_t \tag{4.4}$$

r_t, s_t and c_t represent the deterministic components of trend, shirt term periodic behaviour and long term periodic behaviour components of the time series respectively. ϵ_t represent the random component of the time series, and is often the component of the time series one attempts to model with an ARMA model.

4.2.2 Hidden Markov Models

Let $\{X_n\}$ be a stochastic process where the random variables contained in $\{X_n\}$ only can take on a finite number of values which we will call states. Let X_n denote the state at time period n. The probability of X_n transitioning from state i to state j at the next time period $n + 1$ is called the transition probability, and is denoted p_{ij}. It seems natural that p_{ij} is conditional on what the state has been in previous time periods. $\{X_t\}$ is said to be a *Markov chain* if p_{ij} only is conditional on the past state, as shown in equation (4.5).

$$\begin{aligned} p_{ij} &= P(X_n = i | X_{n-1} = i_{n-1}, X_{n-2} = i_{n-2}, ..., X_1 = i_1, X_0 = i_0) \\ &= P(X_n = i | X_{n-1} = i_{n-1}) \end{aligned} \tag{4.5}$$

Suppose now that the states that the process is in are hidden from the observer. Instead there exists a finite set of signals $\{S\}$ that are emmitted when the process enters a state. For each state

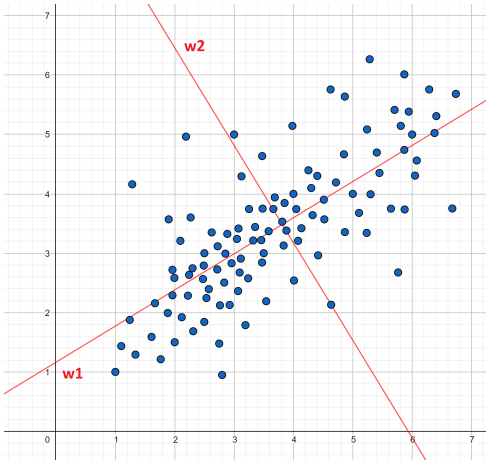

Figure 4.2: Illustration of how the principal components are found

there is a set of *emission probabilities* $P(S = s|X = j)$ associeted with a subset of the signals that can be emmitted. In addition, let the probability of emmitting signal s, at time period n, in state j ($P(S_n = s|X_n = j)$) be independent of previous states, and signals emmitted. A model of this type where the signals $S_1, S_2, ...$ are observed, and the underlying Markov states remain hidden is called a *hidden Markov model* [56].

4.2.3 Principal Component Analysis and Independent Component Analysis

There are two ways of reducing dimensionality in classification and regression tasks. One can choose to only use a subset of the dimensions given, or one can use a combination of the features given. PCA and ICA are both tools that reduce dimensionality by using a linear combination of the original dimensions given. Consider a multivariate time series represented by the matrix X with n columns, where each column is a univariate time series of length m. X can then be reduced to a matrix of fewer columns of the same length by multiplying it with a projection matrix W, as in equation 4.6. Let Z represent the reduced matrix. In PCA, the columns of Z are called the *principal components*, and in ICA they are called the *independent components*.

$$Z = WX \tag{4.6}$$

PCA and ICA is are methods for choosing W. In PCA W is chosen in such a manner that the resulting dimensions are the dimensions with the highest variance. The principal components are in fact the eigenvectors of the covarience matrix of X, so all the principal components are orthogonal. This is best illustrated graphically as in figure 4.2 where w_1 and w_2 illustrate the first and second principal components respectively.
In ICA one assumes that the matrix of observed variables is a linear combination of mutually independent variables with an addition of noise, as illustrated in equation 4.7. Where X is the matrix of observed variables, S is the matrix of independent hidden variables, A is called a "mixing matrix" and E is a noise matrix.

$$X = AS + E \tag{4.7}$$

In ICA one aims to reconstruct S as well as possible. ICA is based on the central limit theorem which states that a sum of random variables with arbitrary probability distributions will tend towards a Gaussian distribution. Hence, one chooses W such that the resulting independent components are mutually independent and show as little Gaussian properties as possible.

4.3 Representation Methods

There are numerous ways which a time series can be represented. Aghabozorgi, Shirkhorshidi, and Wah [53] define a time series representation given time series data $\{x_t\} = \{x_1, x_2, ..., x_T\}$ as transforming the time series into another vector $\{x_t\} = \{\hat{x}_1, \hat{x}_2, ..., \hat{x}_L\}$ where $L < T$. In theory $L = T$, but most often the point in transforming a time series is to reduce the amount of information present in the raw time series to more easily reveal patterns of interest.

Feature extraction method	Articles
Basic signal statistics.	[57–60]
PCA or ICA.	[58, 59, 61–65]
Time-frequency decomposition.	[66–73]
Matrix, or tensor decomposition.	[61, 74–77]
Symbolic aggregate approximation (SAX).	[66, 78, 79]
Permutation based coding.	[80]
Spline functions.	[81]
Topological feature extraction.	[82]
Autoencoder.	[60]

Table 4.1: Feature extraction based representation methods.

Table 4.1 summarizes the different feature extraction based representation methods encountered in literature. Here the basic signal statistics category refers to the same values as in table 3.1: window average, max/min values and RMS values. basic signal statistics are usually coupled with a form of dimensionality reduction such as PCA. Hulsegge and Greef [58] and Wang et al. [59] both extract signal statistics from the time series they analyse (road grade time series [59], and slaughterhouse health time series [58]), and then use PCA for dimensionality reduction. They then use euclidean distance to calculate the dissimilarity matrix between the time series. He and Tan [61] uses multivariate time series generated by a piezoelectric sensor-actuator system, then represent the time series with a 3-order hysteresis tensors. They then use multilinear PCA to reduce the dimensionality of the tensor, and use a selv-developed tensor distance metric to measure similarity between the tensor representations. Li [63] uses common PCA to represent the centers of clusters. The principal components of the principal components are used to reconstruct the time series, and the reconstruction error is used as a similarity metric. The category of time-frequency decomposition encompasses papers using the DWT, the continous wavelet transform, the DFT, spectral densities and fragmented periodograms. Zaifoglu, Akintug, and Yanmaz [66], Lee, Na, and Lee [68], and Muruga Radha Devi and Thambidurai [69] all use the DWT to decompose the time series signals. Zaifoglu, Akintug, and Yanmaz [66] explores DWT as an alternative to other representation methods, while Lee, Na, and Lee [68] and Muruga Radha Devi and Thambidurai [69] use DWT alone. Caiado, Crato, and Poncela [72] and Harvill, Kohli, and Ravishanker [73] both use periodograms to represent the time series to conserve some of the time-domain information about the signal. Sun [74] suggest representing multivariate time series with their covariance matrices, then projecting them to a tangent space and using the euclidean distance to measure the similarity between them. Izakian and Mesgari [75] and Bowala and Fernando [76] both use singular value decomposition (SVD) to decompose multivariate time series. Izakian and Mesgari [75] uses SVD to represent the time series, and uses a particle swarm optimization to estimate the coefficients of an SVD representation of the cluster centre. On the other hand, Bowala and Fernando [76] extract windows of the different

time series, represent them using SVD, and then reconstruct them with a lower resolution. Thus compressing the length time series, this approach can be considered a form of piecewise aggregate approximation. SAX is also a form of piecewise aggregate approximation, but in SAX one represents segments of time series of specified length, with symbols. This method is also used to compress the length of the time series. Aghabozorgi and Wah [78, 79] have an approach that uses SAX as a representation method for the time series, and both use the approximate distance as the similarity metric. Zaifoglu, Akintug, and Yanmaz [66] experiments with SAX as an alternative to the other approaches (AR model, and DWT). For the SAX representation they use the minimum distance between symbolic representations. Ruiz-Abellon, Gabaldon, and Guillamon [80] suggest a system that extracts windows of the time series, and maps them to unique codewords based on how they permute. They do this for all the univariate components of a multivariate time series object, and then use the frequency of occurence of different codeword combinations to construct a probability distribution of different codewords. Hence, they transform a multivariate time series into a probability distribution of codeword combinations. Pereira and Mello [82] analyze the trajectory of time series in phase-space. They then extract the topological features of the trajectory curves, such as the number of holes, birth time and death time of holes. Finally the measure similarity between time series as the euclidean distance between their topological features. Bode et al. [60] compare a feature extraction based representation method, to a raw-data based approach. In the feature extraction based approach they train an autoencoder to reconstruct the time series used, then they separate the autoencoder into the encoder and the decoder parts. They then used the compressed features at the output of the encoder as the representation of the time series in the clustering tasks. They then go on to test a multitude of different clustering algorithms to cluster the compressed features.

Time series model	Articles
ARMA model.	[54, 66, 83–89]
HMM.	[90–92]
State space model.	[62, 81, 93]
Varience ratio statistics.	[94]
Copula based model.	[95]
Network model.	[96, 97]

Table 4.2: Model based representation methods

Table 4.2 shows the different model based representation methods encountered, as one can see the ARMA model is the representation method most frequently encountered. However, within this approach there is also great variety, especially with regard to model complexity. Zaifoglu, Akintug, and Yanmaz [66], D'Urso, De Giovanni, and Massari [85], D'Urso et al. [86], and Zeng et al. [87] fit the time series with an AR model, and use the similarity between the AR-coefficients to cluster the time series. D'Urso, De Giovanni, and Massari [83] models the time series with a generalized AR conditional heteroscedasticity model, which is a variation of the ARMA model that is better suited for non-stationary time series. More specifically, in GARCH models one models the variance of the innovations component as an ARMA model itself. They then test different similarity metrics based on squared Euclidean distance between the model coefficients to cluster the time series. Aslan, Yozgatligil, and Iyigun [84] and Otranto and Mucciardi [88] both use forms of threshold AR models. Threshold AR models aim to describe non-linear time series by splitting the time series into operational regimes using a threshold variable, and then modeling the different regimes with different linear AR models [84]. Aslan, Yozgatligil, and Iyigun [84] then tests 22 different distance metrics to measure

similarity between the model coefficients, which is then used to cluster the time series. Otranto and Mucciardi [88] on the other hand, use the Wald statistic to test whether to time series are from the same generative process, and use the p-value as a distance measure. Nguyen et al. [54] use AR models to represent different clusters, and cluster the different time series as to which AR model is the most likely generative process for a given time series, using the expectation-maximization algorithm. Dias, Vermunt, and Ramos [90] have a very similar approach to Nguyen et al. [54], just that they use HMMs to represent the clusters, and cluster the time series with regard to which HMM is the most likely generative process. Gomez-Losada, Pires, and Pino-Mejias [91] represent air pollution time series with a HMM. The states of the HMM correspond to different long-term states of pollution emission levels. They cluster the different time series based on which "polution emission state" each time series is in at a particular time. Ghassempour, Girosi, and Maeder [92] model each time series with a HMM, and then calculate the Kullback-Liebler distance between the likelihood of different observation sequences given the state and specific HMM. Motlagh, Berry, and O'Neil [62] represents user electricity-load time series with a state space model. Individual time series are treated as dynamical systems with states in phase space. The states are then represented with mapping functions, and uses the euclidean distance between parameters of the mapping functions to cluster time series. The state space representation of the time series is compared to a feature-based approach using PCA to compress signal statistics that are extracted from the electricity-load time series. Nair et al. [93] also explore the use of state space models, but to extract features of a time series. Finazzi et al. [81] compare a feature-based approach and a model-based approach for clustering time series. In the functional approach they represented as a linear combination of spline functions. The Euclidean distance between the vector of function coefficients associated with each time series is then used to measure similarity. The idea behind their state space model approach is to model every observed time series as a linear combination of a set of hidden "state" time series. Bastos and Caiado [94] use variance ratio statistics to represent the equity index time series, then they use cluster the time series into groups maximizing interdependence between the time series in clusters, and minimizing the interdependence between time series in different clusters. Disegna, D'urso, and Durante [95] uses spatial multivariate time series. It extracts the spatial information from the multivariate time series, then goes on to decompose the univariate components into their deterministic and random components described in section 4.2. The extracted innovation components are then represented by multivariate cumulative distribution functions, which are called copulas. The similarity metric for the multivariate time series is calculated as the p-norm of the difference between their copula representations. Ferreira and Zhao [96] and Zhou et al. [97] have a unique approach to clustering time series. Ferreira and Zhao [96] test a variety of similarity metrics to calculate the dissimilarity matrix between a set of univariate time series. They then use points to represent the time series, and use the measures of similarity to create vertices between the points to create a network model. To cluster the time series they then use community detection algorithms on the network model constructed. Zhou et al. [97] extends the model developed by Ferreira and Zhao [96] to be able to apply it to multivariate time series. Zhou et al. [97] also develop an an algorithm based on non-negative matrix factorization to detect communities in the multivariate network model they use to represent the time series.

4.4 Theory Behind Clustering Algorithms

As mentioned before clustering is a form of unsupervised machine learning. The goal is to divide the dataset into clusters, by maximizing some similarity metric for members of the same cluster, and minimizing the same metric for members of different clusters. In this section a detailed description will be given of the most common clustering algorithms enountered in literature.

4.4.1 K-means family

"K-means family" is a collective term for a family of hard clusterimg algorithms where K-means is probably the most famous algorithm. They all require the number of clusters (K) that the data should be divided into to be specified beforehand. What differs between the algorithms in the K-means family is their representation of cluster centers. The K-means algorithm represents the cluster centers with the mean of the members in the clusters, which is fairly intuitive. The K-medoids algorithm represents the cluster centers with the medoid of the members in the clusters. K-medoids is also called "partitioning around medoids". The medoid in this context is the member of a cluster which has the smallest average dissimilarity to the other members of the cluster. The advantage of K-medoids, over K-means is that it represents the cluster centers with actual members of the clusters, instead of a virtual average. This can be computationally efficient when dealing with data objects of large size, such as time series.

The general family of K-means algorithms work iteratively. In the first iteration the cluster centers are randomly initialized, then the dissimilarity between all data objects and the cluster centers are calculated, the data objects are then assigned to the cluster with the minimal dissimilarity to the cluster center, the cluster centers are updated and algorithm repeats by recalculating the dissimilarity between all points and the different cluster centers. These steps are repeated untill the cluster centers, and membership assignments converge. The biggest disadvantages with the family of K-means algorithms are that they can be stuck in local optimums, and that the number of clusters have to be specified beforehand.

4.4.2 Fuzzy C-means family

The fuzzy C-means family of algorithms are equal to the family of K-means algorithms in all ways except one. The family of C-means algorithms are said to be *soft* clustering algorithms. This means that all data objects can have a degree of memberships to all clusters, where as in the family K-means algorithms data objects can only be members of one cluster. This fuzziness property leads to two other parameters that must be defined. The power of membership degree which decides how many clusters each data object can be members of, and the degree of membership matrix which keeps track of the membership degrees of all data objects. The power of membership degree must be fixed together with the number of clusters, and the degree of membership matrix is updatated each iteration.

4.4.3 Hierarchical clustering

Hierarchical clustering algorithms come in many flavours, but they all build hierarchies of clusters [1]. A great advantage with hierarchical clustering algorithms is that the number of clusters is not specified beforehand. Hierarchical clustering algorithms can be split into agglomorative, and divisive algorithms. Agglomorative hierarchical clustering algorithms start with all data objects as their own clusters.

4.4.4 Self-organizing maps

4.4.5 Expectation-maximization

4.4.6 Spectral clustering

4.5 Clustering Algorithms

The expectation maximization algorithm is used for clustering when the approach uses a set of mixture models to represent the clusters. Finazzi et al. [81]

Clustering Algorithm	Articles
K-means family	[59–61, 68, 69, 79, 81, 82, 84, 92, 94]
Fuzzy C-means family	[67, 75, 83–87, 95]
Hierarchical clustering	[58, 60, 64–66, 69, 71, 73, 74, 80, 81, 88, 94]
Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH)	[60, 76]
Self-organizing map (SOM)	[57, 69, 93]
Expectation-maximization	[54, 60, 81, 90]
Spectral clustering	[60, 72, 84]
Affinity propagation	[60]
DBSCAN	[60]
Mean shift	[60]
Custom algorithms	[63, 70, 77, 78, 89]

Table 4.3: Clustering algorithms encountered in literature

As mentiond in section Bode et al. [60] use the encoder part of an autoencoder to compress the time series produced by an energy research center. The test hierarchical clustering, K-means, spectral clustering, expectation-maximization, BIRCH, affinity propagation, mean shift and DBSCAN to cluster the compressed features. The feature-based approach is compared to a raw-data based approach, and they found that the feature-based had better accuracy on a labelled dataset.

4.6 Discussion

Chapter 5

Data exploration

In this section we will explore the use of some basic techniques from PCA to explore a small dataset of 13 multivariate time series from a wind farm in Norway. By reccomendation of Professor Adil Rasheed, the exploration will see if one can detect anomolous behaviour in single turbines in a wind farm by inspecting the cumulative explained varience for different number of principal components, and the reconstruction error.

5.1 Dataset

This dataset is taken from a wind farm in Norway with 13 wind turbines. Each turbine can be viewed as a multivariate time series, with four dimensions: active power produced by a turbine in kilowatts, wind speed measured in front of the blades in meters per second, the rotational speed of the rotor in rpm and the absolute direction of the wind speed in degrees. The values are sampled every five minutes from the 31. of August 2019 00:00 until the 13. of September 2019 21:55, which totals 12239 samples per univariate time series and 48956 samples per wind turbine. The dataset chosen does not have any missing values, so there was no requirement for estimating missing values. A plot of all the individual time series from one turbine is shown in figure 5.1. Before estimating the principal components of the dataset, the dataset was first normalized.

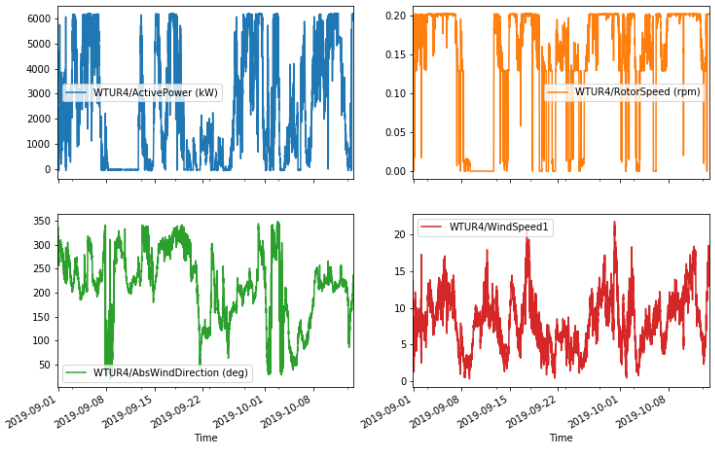

Figure 5.1: The data from one wind turbine

5.2 Explained Varience of Principal Components

One interesting point of comparison for the different wind turbines is the cumulative explained variance of a wind turbine for a given set of principal components. The total variance is the sum of the variences of the individual principal components. Since these multivariate time series have four dimensions four principle components should then be able to explain the variance of the entire dataset, as they are the eigenvectors of the covarience matrix of the time series. The explained variance of a principal component is the ration of the variance of said component to the total variance, and the cumulative explained variance of n principal components is the sum of the explained variance of the n first principal components. Since these turbines are in the same wind farm they are exposed to roughly the same environmental conditions, and one would expect them to have similar cumulative explained variance curves.

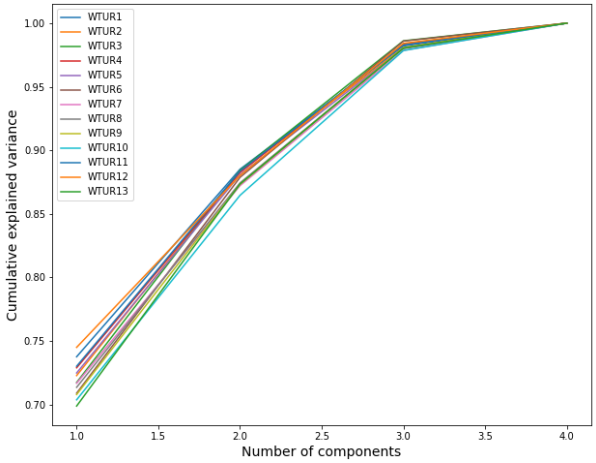

Figure 5.2: Cumulative explained variance for a given number of principal components

In figure 5.2 there is a plot of the cumulative explained varience of the 13 turbines for a given set of principal components. As expected the total variance can be explained by four principle components for all wind turbines. However, as the number of principal components decreases there is a greater variety in the cumulutive explained variance for the different wind turbines. For only one principal component the biggest difference in cumulative explained varience is between turbine two at 75% explained variance, and turbine 13 at 70% explained variance. This difference is not that substantial, from the plot in figure 5.2 one can see that all the curves follow more or less the same shape, so one cannot conclude that any of these curves are due to anomolous behaviour.

5.3 Reconstruction Error

Another indicator of anomolous behaviour is if the multivariate time series associated with a turbine has a significantly different R_e compared to the multivariate time series of the other turbines in the wind farm. Figure 5.3 shows the total MSE of the reconstructed time series associated with the different wind turbines. Here too, one can observe that the different wind turbines all follow approximately the same curve with regard to reconstruction error. With the exception of turbine seven has a slight spike at round about 5000 samples. To get a better reference of what constitutes anomolous behaviour, the data from one turbine will be artificially perturbed in the coming section.

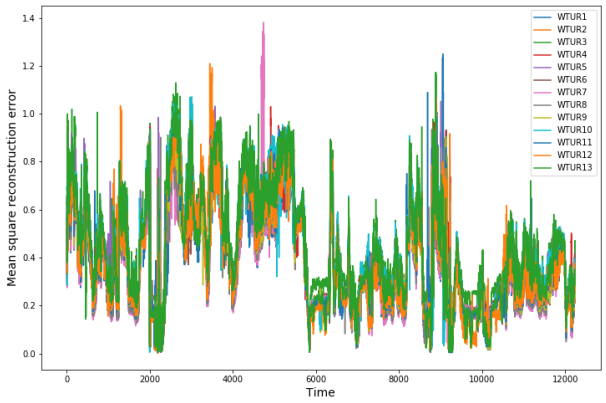

Figure 5.3: Reconstruction error, when representing data with two principal components

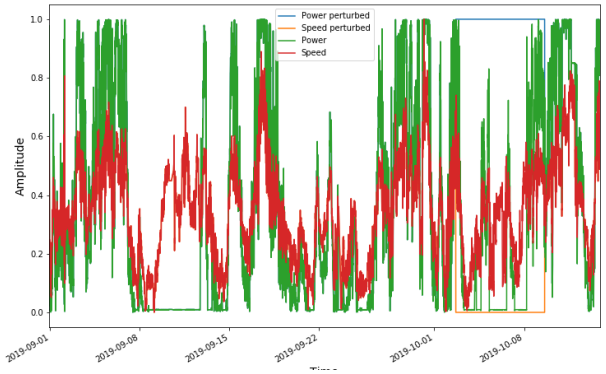

Figure 5.4: Illustration of the perturbed data

5.4 Artificially Perturbation of Data

To artificially perturb the data 2000 samples from turbine 12 are changed. Sample 9000 to sample 1100 of the wind-speed and power time series are set to one and zero respectively. Figure 5.4 illustrates this. This drastically changes the curve of cumulutive explained variance which is depicted in figure 5.5, and the reconstruction error of the time series associated with turbine 12 which is depicted in figure 5.6 and 5.7. From figure 5.5 one can see that the cumulative explained variance curve of turbine 12 is significantly lower than the curves of the other turbines in the farm, giving a reference of how some anomolous behaviour would show up in a cumulative explained variance curve. In figure 5.6 one can see the reconstruction error of turbine 12 increases overall after the perturbation, and increases even more in the particular samples where the time series was altered. Finally, if one compares the reconstruction error of the perturbed data of wind turbine with the reconstruction error of the original data from the other turbines in figure 5.7, one can see that the spike in turbine seven mentioned earlier is of the same magnitude as the spike in the reconstruction error of wind turbine 12 when artificially perturbed.

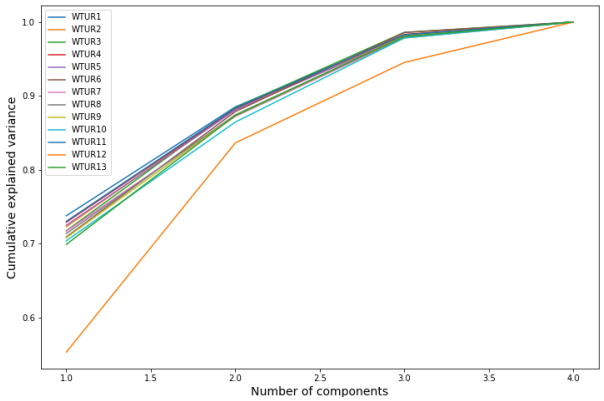

Figure 5.5: Cumulative explained variance for a given number of principal components with perturbed data

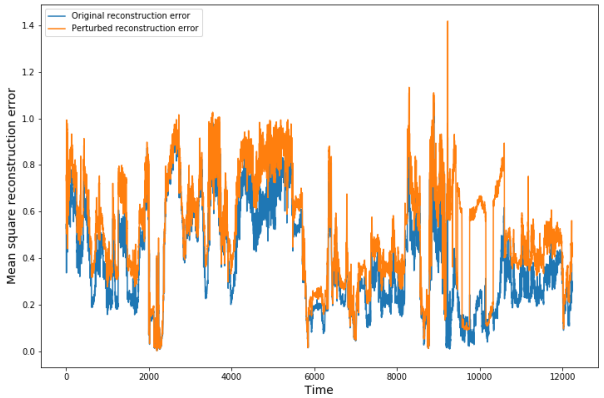

Figure 5.6: Reconstruction of wind turbine 12 using two principal components, with original and perturbed data

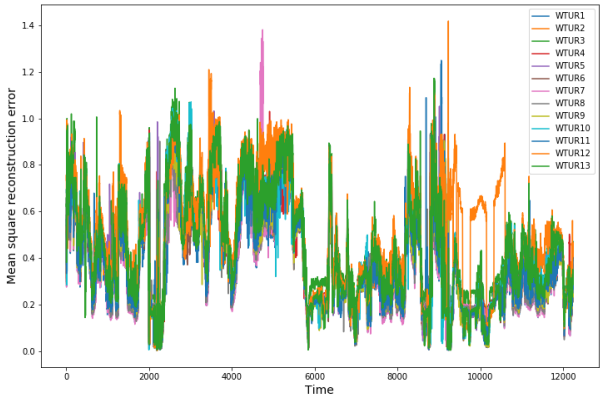

Figure 5.7: Reconstruction error, when representing data with two principal components using perturbed data

Chapter 6

Discussion

Chapter 7

Conclusion

Chapter 8

Appendices

8.1 Appendix A: Summary of articles included from search 1 and 2

Ref.	Input	Feature extraction method	Machine learning model
[13]	Vibration signal.	ARMA model and J48 decision tree	Tests a set of (38) meta-, misc-, rule- and tree-based classifiers for fault detection in blades.
[34]	SCADA data.		Deep autoencoder made of Restricted Boltzman Machines (RBMs) to model normal behaviour of SCADA variables (gearbox and generator temperature). Uses E_e for anomaly detection, with adaptive threshold set using extreme function theory (EFT).
[51]	SCADA data.	Spatiotemporal pattern network	Unsupervised use of RBMs for anomaly detection.
[14]	SCADA data.	Time-frequency domain analysis, DWT and ARMA model	Uses fusion of several classifiers for fault detection in a wind turbine.
[6]	Ultrasonic testing.	Tests linear and non-linear PCA and ARMA models	Neighbourhood component analysis for feature selection. Tests 20 different supervised classifiers for detecting ice on blades.
[28]	Wind-Power curve.		Uses Gaussian Process regression with EFT to determine whether a particular power curve is an outlier.
[22]	Acustic emission.	FFT.	Uses Distinguishability Measure for feature selection, and logistic regression and SVC for binary blade fault classification.
[40]	Power signal, wind speed and ambient temperature.		Hierarchical Extreme Learning Machine (H-ELM) for detection of anomolous behaviour.
[29]	SCADA data.		Gaussian processes regression to estimate wind-power curve

[27]	SCADA data.		Tests KNN, random forest, and SVR to estimate power curve. Detects anomalies by E_e.
[98]	Vibration signals.		Uses unsupervised dictionary learning extracting features which are then used to determine fault in drivetrain bearings.
[38]	Oil temperature, wind speed, rotor speed and active power.		Trains ANN to estimate vibration signal, uses E_e for anomaly detection.
[19]	SCADA data.	PCA.	Sets up baseline model using multiway PCA, then finds outliers by hypothesis testing whether multivariate distribution is equal to baseline.
[16]	FAST wind turbine simulator.	Image texture analysis tools.	KNN, Linear Discriminant Model, decision trees, bag-tree, linear SVC.
[39]	SCADA data.	K-means for outlier elimination.	Uses Auto-Associative Neural Networks as an autoencoder, and the Hotel T2 statistic as a dynamic threshold for the R_e.
[30]	SCADA data.	Grey correlation algorithm for eigenvector extraction.	Use genetic algorithm for feature selection, and SVR for estimating performance curves (active power, rotor speed and blade pitch angle).
[33]	SCADA data.		Uses three NN for normal behaviour modelling of rotor speed, gearbox temperature and generator temperature. E_e sendt to proportional hazard model which sets dynamic threshold.
[44]	SCADA data.		Uses Inductive Transfer Learning and five differen ML classifiers for ice detection on blades.
[46]	Vibration signals.	Variational mode decomposition (VMD).	Uses multi-scale permutation entropy (MPE) used for feature selection, CO-VAL for domain normalization and an SVC for binary fault classification.
[32]	SCADA data.		Uses bins and SVR to estimate blade angle pitch curve.
[35]	SCADA data.		Tests different architectures of ANNs to for estimating temperature of non-drive end bearing. Uses E_e for anomaly detection.
[7]	Images taken by drones		Recurrent Convolutional Neural Network to classify structural damage in blades.
[8]	Uses images taken from ground level		Convolutional Neural Network, and YOLO-based small object detection approach (YSODA) for damage detection in blades and hub.

[17]	Vibration signals, acustic emission and oil particle analysis	DWT.	Uses decision tree for feature selection, and SVC for assessing fault severity in gearbox
[41]	Uses images taken from ground level		Convolutional Neural Network to detect cracks & damage in blades.
[9]	Ultrasonic testing	PCA and ARMA models.	Neighbourhood Component analysis for feature selection and an ensamble of KNN, linear SVC, decision trees, LDA and subspace discriminant to estimate amount of dirt and mud on blades.
[47]	Uses pitch position, rotor speed and generator speed.		Detects faults with an SVC with parameters optimized by Cuckoo-swarm optimization.
[15]	Vibration signals	ARMA model.	Dominating features selected with J48 decision tree, fault classification done with Bayesian- and lazy classifiers.
[18]	Vibration signals, acustic emission and oil particle analysis	DWT and PCA.	Dominating features selected with decision tree, fault detection done with SVC.
[10]	Images taken from imaging array		Uses a deep neural network for binary classification of blade defects.
[11]	Uses images taken by drone	Uses a CNN trained on an unrelated image dataset to extract general features.	Compress features with PCA, and pass them to a unsupervised one-class SVM.
[45]	Uses the FAST wind turbine simulator to get SCADA data.	Random forest.	Uses XGBoost to train an ensamble of classifiers for specific faults.
[36]	SCADA data.		Uses several ANN to build a normal behaviour model of temperature in gearbox and high speed shaft, then uses E_e together with the age of the age of the turbine to predict anomolous behaviour.
[31]	SCADA data.		Uses Pearson product-moment rank correlation to select features, and applies different ANN structures to predict the active power.
[48]	SCADA data from a simulink model of a wind turbine.		Uses DBN for detecting anomolous behaviour. First traines individual RBMs to recreate input, and then uses labeled data to fine tune DBN to detect faults.

[37]	SCADA data.		Uses kolmogorov-smirnov test to compare different turbines at same moment in time combined with the E_e of the gearbox bearing temperature of an ANN to detect anomalies.
[49]	Vibration signals.	Approximates vibration distributions at different rotor speeds with Weibull distribution.	Uses a HMM for statistical fault detection.
[42]	SCADA data.		Compares linear models, ANNs and state-dependent parameter models for fault detection.
[21]	SCADA data.	parallel factor analysis (PARAFAC) as a decomposition method	uses K-means clustering after decomposition for fault detection.
[20]	Uses a wind turbine simulator for SCADA data.		Multiple PCA models are as a statistical reference reflecting the data variability in local zones and used in parallel for online fault detection.
[23]	Vibration signals.	Variational mode decomposition	Uses Fisher score and ReliefF algorithm for feature selection. Feeds selected signals into a multi-class SVC for bearing fault detection.
[43]	SCADA data.		Uses deep neural networks for detection of icing on the blades.
[99]	SCADA data.		Combines NN with alarms generated by SCADA system to reduce false alarm rate.
[4]	SCADA data.		Uses K-means clustering to partition turbines into different operating states, and a specific DBN of RBMs for each cluster to forecast the gearbox main bearing temperature. Uses E_e to detect anomalies, threshold set by Mahalanobis distance.
[100]			This is a literary review of vibration based condition monitoring and fault diagnosis of planetary gearboxes in wind turbines.
[50]			This is a literary review of machine learning methods used for condition monitoring of wind turbines.

Table 8.1: Summary of machine learning methods for wind turbine condition monitoring

8.2 Appendix B: Summary of articles included from search 3

Ref.	Representation	Similarity measure	Clustering Algorithms	Evaluation
[90]	Mixture Gaussian hidden Markov model (MGHMM).		Expectation-maximization (EM).	Bayesian information criterion (BIC).
[94]	Variance ratio statistics.	Euclidean distance.	Hierarchical clustering mainly, and K-means.	Duda-Hart $Je(2)/Je(1)$ indices.
[91]	HMM. States correspond to concentration regimes.	Which state each HMM is in.	Cluster together time series with corresponding HMMs in the same state.	
[53]	This is a review of time series clustering.			
[57]	Raw time series and some extracted statistics: variance, covariance, spread and differences.		Growing hierarchical self-organizing map (GHSOM).	
[81]	Compares to methods: a model-based approach using a state space model and functional approach where time series are represented as linear combinations of spline functions.	Euclidean distance.	State space modelling, K-means and complete-linkage hierarchical clustering.	L-curve and gap statistic.
[82]	EMD for filtering out stochastic components, then extract topological features.	Euclidean distance.	K-means.	Precision, recall, F1-score and Matthews correlation coefficient.

[96]	Construct network between time series using dissimilarity matrix. Use KNN, and ϵ-NN to create networks from matrix.	Test a multitude of different distance functions. DTW performs best.	Test many community detection algorithms to sort network into clusters.	Rand index.
[78]	SAX.	Approximate distance, Euclidean distance, and DTW.	Custom three step algorithm (MTC), with preclustering, subclustering, and merging to form final clusters.	Accuracy, precision, recall and F-measure.
[83]	Generalized autoregressive conditional heteroskedasticity (GARCH) model.	Tests different metrics based on squared Euclidean distance between unconditional volitility and time varying volitility.	Tests different variations of fuzzy C-medoids.	Xie-Beni index, and Fuzzy Rand index.
[73]	Bispectral Smoothed Localized Complex EXponential (BSLEX) algorithm.	Aggragated quasi-distance between smoothed bispectra across blocks.	Agglomorative hierarchical clustering with Ward's linkage.	Silhuette index as stopping criterion, and Rand Index, entropy and purity to evaluate cluster effectiveness.
[92]	HMMs.	Kulback-Leibler distance between the likelihood of a certain observation sequence given HMM.	K-medoids.	Silhouette index, Bavies-Bouldin index and Dunn index.
[95]	Copula-based model for time series.	P-norm of difference between copula of two points, and upper bound copula.	Fuzzy C-medoids.	Fuzzy Silhouette (FS) index, adjusted Rand index (ARI), fuzzy Rand index (FRI).
[66]	DWT, SAX and AR model.	Minimum distance, Euclidean, Minkowski, Pearson correlation coefficient and DTW distance.	Agglomerative Hierarchical clustering with Ward linkage.	Uses the clusters produced to perform regional frequency analysis, and then evaluates model using bias, RMSE, relative RMSE (RRMSE) and Nash criterion.
[64]	ICA.	Not specified.	Hierarchical clustering with complete linkage.	

[79]	SAX.	Approximate distance between symbolic representations of time series.	Extended version of K-modes.	SSE for stopping criteria, Rand index, Normalized Mutual Information (NMI), Purity, Jaccard, F-measure, Folks and Mallows (FM) and entropy.
[74]	Tranforms the covariance matrices of the time series into a tangent space.	Euclidean distance.	Hierarchical clustering with average linkage.	
[71]	Normalized spectral densities.	Total variation distence.	Agglomorative hierarchical clustering with complete and average linkage.	Dunn's index.
[84]	Self-exciting threshold autoregressive model.	Primarily tests Euclidean distance, Haussdorf distance and DTW, but, tests 22 different ones.	Primary method is spectral clustering, but also tests K-medoids, and fuzzy C-means.	Measures accuracy of method on clustering simulated data, and uses Gap statistic as stopping criterion.
[85]	AR model.	A type of exponential Euclidean distance.	Fuzzy C-medoids.	Fuzzy Silhouette index.
[67]	Continous wavelet transform.	Multi-scale PCA similarity matric.	Fuzzy C-means.	Precision and recall of classification according to labels, and silhouette index.
[93]	Preprocessing using Hodrick-Prescott filter, primarily represents time series with state space models.		Self-organizing map (SOM).	Silhouette index as stopping criterion.
[86]	ARIMA model.	Euclidean distance between AR weights.	Trimmed fuzzy C-medoids.	Decides number of clusters by looking at the rate of decrease, and second derivative of an objective function with regard to a trimming factor α.

[80]	Permutation based coding of time series.	Use four distance metrics based on mutual information, entropy and Cramer's V association measure.	Hierarchical clustering with single, complete and average linkage.	
[54]	Mixture of autoregressions (MoAR) models.		Maximum pseudolikelihood esitmation using EM algorithm.	
[65]	Use PCA and custom ICA algorithm for feature extraction.	Euclidean distance between extracted features.	Hierarchical clustering with average, single, complete and Ward linkage, and K-means.	CH, Friedman, C-index, Dunn's, SDbw and Silhouette index.
[58]	Extracts various signal statistics, and performs feature extraction using PCA.	Euclidean distance.	Hierarchical clustering with complete linkage.	
[68]	DWT with the Haar wavelet, and a globel sensitivity analysis.	Euclidean distance, to minimize variance.	K-means.	
[97]	Multi-relational network in topological domain, static (time-invariant), and dynamic (time-varying).		Multi-nonnegative matrix factorization. Compares their approach to three other community detection algorithms.	Rand index, adjusted Rand index and purity.
[75]	Uses Singular value decomposition (SVD) to represent the cluster centroids.	Pearson correlation coefficient.	Fuzzy C-means with particle swarm optimization.	Precision, and F-measure.
[61]	Multivariate time series are transformed into 3-order hysteresis tensors, then multilinear PCA is used to reduce dimensionality.	Tensor distance metric. Cluster centers initialized based on cycle feature variation.	Tensor K-means (CTK-means).	RI, ARI, Jaccard coefficient and FM index.

[101]	Compares ten model-based clustering methods. GMM and Markov-switching model perform best.		Expectation-maximization (EM).	Misclassification rates.
[69]	Vari-segmented DWT.	Euclidean distance.	K-means, hierarchical agglomerative clustering and SOM.	
[70]	Discrete Fourier transform (DFT).	Euclidean distance.	Delaunay Triangulation method.	Purity and F-measure.
[76]	Piecewise SVD, and piecewise aggregate approximation.	Euclidean Distance.	BIRCH.	
[59]	Extracts signal statistics, and uses PCA for feature selection.	Euclidean distance.	K-means.	Analyses the correlations of specific features with different clusters.
[72]	Fragmented periodogram.	Euclidean distance.	Spectral clustering.	
[60]	Extract statistical features of time series, then use a convolutional auto-encoder for further feature extraction.	Mainly Euclidean distance for the model-based approach.	K-means, hierarchical clustering with Ward linkage, spectral clustering, Gaussian mixture models, BIRCH, affinity propagation, mean shift, DBSCAN.	Adjusted Rand index.
[62]	Compares a feature-based approach using PCA, with a model-based approach using state-space models for the individual time series.	Inverse exponential Euclidean distance for feature based approach, and Euclidean distance for model-based approach.		Silhuette index as stopping criterion.
[87]	AR model.	Euclidean distance.	Fuzzy C-means.	
[88]	Flexible space-time autoregressive (FSTAR) models.	Use Wald statistic to compare model parameters of univariate STAR models, and p-value as a similarity metric.	Hierarchical agglomorative clustering.	ARI.

[63]	Common PCA.	Cluster centroids represented by common projection axis of all time series in a specific cluster, then reconstruction error of time series using cluster centroid used as similarity metric.	Custom algorithm, similar to K-means.	Precision.
[77]	Map the time series to multiple high-dimensional tensors using multiple kernals.	Matrix L^p-norms.	Self-developed multi kernal clustering algorithm (MKC).	NMI and Rand Index (RI).
[89]	Vector autoregressive (VAR) models.	Euclidean distance.	Test two self-developed algorithms based on performing statistical test of whether time series come from same DGN.	Purity index.

Table 8.2: Summary of model based time-series clustering methods

8.3 Appendix C: Code

To produce all the relevant plots all the listings can be posted and run in the same file, to produce all the relevant plots used in this report. They are split up into separate listings such that it is easier to link each listing to each plot.

```
import pandas as pd
from sklearn.decomposition import PCA
from datetime import datetime
import time
import timeit
import numpy as np
import matplotlib.pyplot as plt
import copy

work_directory = 'Not relevant'
data_sets_path = \
'data_sets/big_turbine_data/'
filename = 'big_turbine_data'
path_pictures = work_directory + 'pictures/'

completeDF = pd.read_csv(work_directory+data_sets_path+filename+'.csv',
                         delimiter=',',
                         skiprows=1)

completeDF['Time'] = pd.to_datetime(completeDF['Time'], format='%Y-%m-%dT%H:%M:%S.%fZ')
completeDF.set_index('Time', inplace=True)
```

Listing 1: Module fetching and loading data

```
# Dropping max/min cols
column_names = completeDF.columns
for name in column_names:
    if ('Max' in name) or ('Min' in name):
        completeDF.drop(name, axis=1, inplace=True)

# Renaming columns
column_names = completeDF.columns
new_col_names = []
for name in column_names:
    split_name = name.split('/')
    turbine_num = split_name[2]
    if split_name[-1] == 's)':
        value_meas = split_name[-3]
    else:
        value_meas = split_name[-1]
    new_name = turbine_num + '/' + value_meas
    split_new = new_name.split('(Avg)')
    new_col_names += [split_new[0] + split_new[1]]
completeDF.columns = new_col_names

# Create a dictionary of windturbine dataframes
column_names = completeDF.columns
dict_of_turbine_dfs = {}
for i in range(1,14):
    turbine_num = 'WTUR' + str(i) + '/'
    tempDF = pd.DataFrame()
    for name in column_names:
        if turbine_num in name:
            tempDF[name] = completeDF[name]
        dict_of_turbine_dfs[turbine_num[:-1]] = tempDF

# Normalize values from each dataframe
dict_norm_dfs = {}
for turb_name in list(dict_of_turbine_dfs.keys()):
        df = dict_of_turbine_dfs[turb_name]
        dict_norm_dfs[turb_name] = (df - np.min(df.values, axis=0)) / (np.max(df.values, axis=0) - np.min(df.va
```

Listing 2: Data formatting and normalization

```
dict_of_turbine_dfs['WTUR4'].plot(figsize=(12,8),subplots=True, layout=(2,2))
plt.legend(fontsize=10)
name = path_pictures + 'one_turbine_all_vals.png'
plt.savefig(fname=name)
```

Listing 3: Code used to produce plot in figure 5.1

```python
# PCA Finding portions of explained variance
cumulative_explained_variance = {}
for turb in list(dict_norm_dfs.keys()):
    X = dict_norm_dfs[turb].values
    pca = PCA().fit(X)  # project from 64 to 2 dimensions
    cumulative_explained_variance[turb] = np.cumsum(pca.explained_variance_ratio_)

# Figure XX Plot explained variance for a given number of principal components
plt.figure(figsize=(10,8))
for turb in list(cumulative_explained_variance.keys()):
    plt.plot(cumulative_explained_variance[turb], label=turb)

# plt.title('Explained variance vs. Number of PCA components', fontsize=18)
plt.xlabel('Number of components', fontsize=14)
plt.ylabel('Cumulative explained variance',fontsize=14);
plt.legend(fontsize=10)

name = path_pictures + 'cumulative_explained_variance.png'
plt.savefig(fname=name)
```

Listing 4: Code used to calculate cumulative explained variance, and produce plot in figure 5.2

```python
mse = {}
for turb in list(dict_norm_dfs.keys()):
    Xstd = dict_norm_dfs[turb].values
    pca = PCA(n_components=2)
    pca.fit(Xstd)
    P=pca.components_
    T = Xstd.dot(P.T)
    Xhat = np.dot(T[:,0].reshape(-1,1),P[0,:].reshape(-1,1).T)
    mse[turb] = (np.square(Xstd - Xhat)).sum(axis=1)

# Figure XX Plot of reconstruction error using principal components
plt.figure(figsize=(12,8))
for turb in list(mse.keys()):
    plt.plot(mse[turb], label=turb)

# plt.title('Reconstruction Error when using two principal components', fontsize=18)
plt.xlabel('Time', fontsize=14)
plt.ylabel('Mean square reconstruction error',fontsize=14);
plt.legend(fontsize=10)
name = path_pictures + 'reconstruction_error.png'
plt.savefig(fname=name)
```

Listing 5: Code used to calculate reconstruction error, and produce plot in figure 5.3

```
# Figure XX Plot of values from perturbed WT versus non-perturbed
dict_perturbed_norm_dfs = copy.deepcopy(dict_norm_dfs)
dict_perturbed_norm_dfs['WTUR12'].columns
for idx in range(9000,11000):
    dict_perturbed_norm_dfs['WTUR12'].iat[idx,0] = 1
    dict_perturbed_norm_dfs['WTUR12'].iat[idx,2] = 0

df_to_plot = pd.DataFrame()
df_to_plot['Power perturbed'] = dict_perturbed_norm_dfs['WTUR12']['WTUR12/ActivePower (kW)']
df_to_plot['Speed perturbed'] = dict_perturbed_norm_dfs['WTUR12']['WTUR12/WindSpeed1']
df_to_plot['Power'] = dict_norm_dfs['WTUR12']['WTUR12/ActivePower (kW)']
df_to_plot['Speed'] = dict_norm_dfs['WTUR12']['WTUR12/WindSpeed1']
df_to_plot.plot(figsize=(12,8))
plt.xlabel('Time', fontsize=14)
plt.ylabel('Amplitude',fontsize=14);
plt.legend(fontsize=10)
name = path_pictures + 'perturbed_vs_unperturbed.png'
plt.savefig(fname=name)
```

Listing 6: Code used to produce plot in figure 5.4

```
cumulative_explained_variance = {}
for turb in list(dict_perturbed_norm_dfs.keys()):
    X = dict_perturbed_norm_dfs[turb].values
    pca = PCA().fit(X)  # project from 64 to 2 dimensions
    cumulative_explained_variance[turb] = np.cumsum(pca.explained_variance_ratio_)

#fig, ax = plt.subplots(7,2)
plt.figure(figsize=(12,8))
for turb in list(cumulative_explained_variance.keys()):
    plt.plot(cumulative_explained_variance[turb], label=turb)

# plt.title('Explained variance vs. Number of PCA components', fontsize=20)
plt.xlabel('Number of components', fontsize=14)
plt.ylabel('Cumulative explained variance',fontsize=14);
plt.legend(fontsize=10)
name = path_pictures + 'explained_variance_perturbed.png'
plt.savefig(fname=name)
```

Listing 7: Code used to calculate cumulative explained variance of perturbed data, and produce plot in figure 5.5

```
mse_pert = {}
for turb in list(dict_perturbed_norm_dfs.keys()):
    Xstd = dict_perturbed_norm_dfs[turb].values
    pca = PCA(n_components=2)
    pca.fit(Xstd)
    P=pca.components_
    T = Xstd.dot(P.T)
    Xhat = np.dot(T[:,0].reshape(-1,1),P[0,:].reshape(-1,1).T)
    mse_pert[turb] = (np.square(Xstd - Xhat)).sum(axis=1)

plt.figure(figsize=(12,8))
for turb in list(mse_pert.keys()):
    plt.plot(mse_pert[turb], label=turb)

# plt.title('Reconstruction Error when using two principal components', fontsize=20)
plt.xlabel('Time', fontsize=14)
plt.ylabel('Mean square reconstruction error',fontsize=14);
plt.legend(fontsize=10)
name = path_pictures + 'reconstruction_error_perturbed.png'
plt.savefig(fname=name)
```

Listing 8: Code used to calculate reconstruction error using perturbed data, and produce plot in figure 5.3

```
plt.figure(figsize=(12,8))
plt.plot(mse['WTUR12'], label='Original reconstruction error')
plt.plot(mse_pert['WTUR12'], label='Perturbed reconstruction error')

# plt.title('Reconstruction Error when using two principal components', fontsize=20)
plt.xlabel('Time', fontsize=14)
plt.ylabel('Mean square reconstruction error',fontsize=14);
plt.legend(fontsize=10)
name = path_pictures + 'pert_vs_unpert_reconstruction_error.png'
plt.savefig(fname=name)
```

Listing 9: Code used to produce plot in figure 5.6

Bibliography

[1] Espen Waaga. "Machine Learning for Automatic Classification of Wind Turbines". English. Master thesis. NTNU, 2019.

[2] Ml Hossain, A Abu-Siada, and SM Muyeen. "Methods for Advanced Wind Turbine Condition Monitoring and Early Diagnosis: A Literature Review". English. In: *Energies* 11.5 (2018). ISSN: 1996-1073.

[3] Henrique Dias Machado de Azevedo, Alex Maurício Araújo, and Nadège Bouchonneau. "A review of wind turbine bearing condition monitoring: State of the art and challenges". English. In: *Renewable and Sustainable Energy Reviews* 56 (2016), pp. 368–379. ISSN: 1364-0321.

[4] H. Wang et al. "Early fault detection of wind turbines based on operational condition clustering and optimized deep belief network modeling". In: *Energies* 12.6 (2019). ISSN: 19961073.

[5] P. Qian et al. "A novel condition monitoring method of wind turbines based on long short-term memory neural network". English. In: *Energies* 12.18 (2019). ISSN: 19961073.

[6] Alfredo Arcos Jiménez et al. "Linear and nonlinear features and machine learning for wind turbine blade ice detection and diagnosis". English. In: *Renewable Energy* 132 (2019), pp. 1034–1048. ISSN: 0960-1481.

[7] ASM Shihavuddin et al. "Wind Turbine Surface Damage Detection by Deep Learning Aided Drone Inspection Analysis". English. In: *Energies* 12.4 (2019), p. 676. ISSN: 1996-1073.

[8] Zifeng Qiu et al. "Automatic visual defects inspection of wind turbine blades via YOLO-based small object detection approach". English. In: *Journal of Electronic Imaging* 28.4 (2019), pp. 043023–043023. ISSN: 1017-9909.

[9] Alfredo Arcos Jiménez, Carlos Quiterio Gómez Muñoz, and Fausto Pedro García Márquez. "Dirt and mud detection and diagnosis on a wind turbine blade employing guided waves and supervised learning classifiers". English. In: *Reliability Engineering and System Safety* 184 (2019), pp. 2–12. ISSN: 0951-8320.

[10] Ningning Zhang, Chengzhi Lu, and Anmin Wang. "Study on wind turbine blade defect detection system based on imaging array". English. In: *E3S Web of Conferences* 118 (2019). ISSN: 25550403. URL: `http://search.proquest.com/docview/2301959230/`.

[11] Yinan Wang et al. "Unsupervised anomaly detection with compact deep features for wind turbine blade images taken by a drone". English. In: *IPSJ Transactions on Computer Vision and Applications* 11.1 (2019), pp. 1–7. ISSN: 1882-6695.

[12] Pierre Tchakoua et al. "Wind Turbine Condition Monitoring: State-of-the-Art Review, New Trends, and Future Challenges". English. In: *Energies* 7.4 (2014), pp. 2595–2630. ISSN: 19961073. URL: `http://search.proquest.com/docview/1537086932/`.

[13] A. Joshuva; V. Sugumaran. "A machine learning approach for condition monitoring of wind turbine blade using autoregressive moving average (ARMA) features through vibration signals: a comparative study". English. In: *Progress in Industrial Ecology, An Int. J.* 12.1/2 (2018). ISSN: 1476-8917. URL: http://www.inderscience.com/link.php?id=95867.

[14] Vahid Pashazadeh, Farzad R Salmasi, and Babak N Araabi. "Data driven sensor and actuator fault detection and isolation in wind turbine using classifier fusion". English. In: *Renewable Energy* 116.PB (2018), pp. 99–106. ISSN: 0960-1481.

[15] A. Joshuva and V. Sugumaran. "Improvement in wind energy production through condition monitoring of wind turbine blades using vibration signatures and ARMA features: a data-driven approach". English. In: *Progress in Industrial Ecology* 13.3 (2019), p. 207. ISSN: 1476-8917.

[16] Magda Ruiz et al. "Wind turbine fault detection and classification by means of image texture analysis". English. In: *Mechanical Systems and Signal Processing* 107.C (2018), pp. 149–167. ISSN: 0888-3270.

[17] Inturi Vamsi, G.R Sabareesh, and P.K Penumakala. "Comparison of condition monitoring techniques in assessing fault severity for a wind turbine gearbox under non-stationary loading". English. In: *Mechanical Systems and Signal Processing* 124 (2019), pp. 1–20. ISSN: 0888-3270.

[18] Vamsi Inturi et al. "Integrated condition monitoring scheme for bearing fault diagnosis of a wind turbine gearbox". English. In: *Journal of Vibration and Control* 25.12 (2019), pp. 1852–1865. ISSN: 1077-5463.

[19] Francesc Pozo, Yolanda Vidal, and Óscar Salgado. "Wind Turbine Condition Monitoring Strategy through Multiway PCA and Multivariate Inference". English. In: *Energies* 11.4 (2018), p. 749. ISSN: 19961073. URL: http://search.proquest.com/docview/2041094406/.

[20] Azzeddine Bakdi, Abdelmalek Kouadri, and Saad Mekhilef. "A data-driven algorithm for online detection of component and system faults in modern wind turbines at different operating zones". English. In: *Renewable and Sustainable Energy Reviews* 103 (2019), pp. 546–555. ISSN: 1364-0321.

[21] Wenna Zhang and Xiandong Ma. "Simultaneous Fault Detection and Sensor Selection for Condition Monitoring of Wind Turbines". English. In: *Energies* 9.4 (2016), p. 280. ISSN: 19961073. URL: http://search.proquest.com/docview/1780819136/.

[22] T. Regan, C. Beale, and M. Inalpolat. "Wind Turbine Blade Damage Detection Using Supervised Machine Learning Algorithms". In: *Journal of Vibration and Acoustics, Transactions of the ASME* 139.6 (2017). ISSN: 10489002.

[23] L. Fu et al. "Condition monitoring for the roller bearings of wind turbines under variable working conditions based on the Fisher score and permutation entropy." In: *Energies* 12.16 (2019). ISSN: 19961073.

[24] Wikipedia contributors. *Support-vector machine*. English. Dec. 1, 2019. URL: https://en.wikipedia.org/w/index.php?title=Support-vector_machine&oldid=928737848.

[25] Wikipedia contributors. *Restricted Boltzmann machine*. English. Dec. 12, 2019. URL: https://en.wikipedia.org/w/index.php?title=Restricted_Boltzmann_machine&oldid=916416667.

[26] Carl Edward Rasmussen. *Gaussian processes for machine learning*. eng. Cambridge, Mass., 2006.

[27] Elena Gonzalez et al. “Using high-frequency SCADA data for wind turbine performance monitoring: A sensitivity study”. English. In: *Renewable Energy* 131 (2019), pp. 841–853. ISSN: 0960-1481.

[28] Evangelos Papatheou et al. “Performance monitoring of a wind turbine using extreme function theory”. English. In: *Renewable Energy* 113.C (2017), pp. 1490–1502. ISSN: 0960-1481.

[29] Ravi Pandit and David Infield. “Gaussian Process Operational Curves for Wind Turbine Condition Monitoring”. English. In: *Energies* 11.7 (2018), p. 1631. ISSN: 19961073. URL: `http://search.proquest.com/docview/2108516073/`.

[30] Liang Tao et al. “Abnormal Detection of Wind Turbine Based on SCADA Data Mining”. In: *Mathematical Problems in Engineering* 2019 (2019). ISSN: 1024-123X.

[31] Majid Morshedizadeh et al. “Improved power curve monitoring of wind turbines”. In: *Wind Engineering* 41.4 (2017), pp. 260–271. ISSN: 0309-524X.

[32] Ravi Pandit and David Infield. “Comparative assessments of binned and support vector regression-based blade pitch curve of a wind turbine for the purpose of condition monitoring”. English. In: *International Journal of Energy and Environmental Engineering* 10.2 (2019), pp. 181–188. ISSN: 2008-9163.

[33] Peyman Mazidi et al. “A health condition model for wind turbine monitoring through neural networks and proportional hazard models”. English. In: *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability* 231.5 (2017), pp. 481–494. ISSN: 1748-006X.

[34] Hongshan Zhao et al. “Anomaly detection and fault analysis of wind turbine components based on deep learning network”. English. In: *Renewable Energy* 127 (2018), pp. 825–834. ISSN: 0960-1481.

[35] MA Rodriguez-Lopez et al. “Methodology for Detecting Malfunctions and Evaluating the Maintenance Effectiveness in Wind Turbine Generator Bearings Using Generic versus Specific Models from SCADA Data”. English. In: *Energies* 11.4 (2018). ISSN: 1996-1073.

[36] P Bangalore and M Patriksson. “Analysis of SCADA data for early fault detection, with application to the maintenance management of wind turbines”. English. In: *Renewable Energy* 115 (2018), pp. 521–532. ISSN: 0960-1481.

[37] P. Guo, J. Fu, and X. Yang. “Condition monitoring and fault diagnosis of wind turbines gearbox bearing temperature based on kolmogorov-smirnov test and convolutional neural network model”. In: *Energies* 11.9 (2018). ISSN: 19961073.

[38] Marcin Straczkiewicz and Tomasz Barszcz. “Application of Artificial Neural Network for Damage Detection in Planetary Gearbox of Wind Turbine”. In: *Shock and Vibration* 2016 (2016). ISSN: 1070-9622.

[39] Hsu-Hao Yang, Mei-Ling Huang, and Shih-Wei Yang. “Integrating Auto-Associative Neural Networks with Hotelling T2 Control Charts for Wind Turbine Fault Detection”. English. In: *Energies* 8.10 (2015), pp. 12100–12115. ISSN: 19961073. URL: `http://search.proquest.com/docview/1732948122/`.

[40] Peng Qian et al. “A novel wind turbine condition monitoring method based on cloud computing”. English. In: *Renewable Energy* 135 (2019), pp. 390–398. ISSN: 0960-1481.

[41] Abhishek Reddy et al. “Detection of Cracks and damage in wind turbine blades using artificial intelligence-based image analytics”. English. In: *Measurement* 147 (2019), p. 106823. ISSN: 0263-2241.

[42] Philip Cross and Xiandong Ma. “Model-based and fuzzy logic approaches to condition monitoring of operational wind turbines”. English. In: *International Journal of Automation and Computing* 12.1 (2015), pp. 25–34. ISSN: 1476-8186.

[43] Longting Chen et al. “Learning deep representation of imbalanced SCADA data for fault detection of wind turbines”. English. In: *Measurement* 139 (2019), pp. 370–379. ISSN: 0263-2241.

[44] Hongguang Yun et al. “An Adaptive Approach for Ice Detection in Wind Turbine With Inductive Transfer Learning”. English. In: *IEEE Access* 7.99 (2019), pp. 122205–122213. ISSN: 2169-3536.

[45] Dahai Zhang et al. “A Data-Driven Design for Fault Detection of Wind Turbines Using Random Forests and XGboost”. English. In: *IEEE Access* 6 (2018), pp. 21020–21031. ISSN: 2169-3536.

[46] H. Ren et al. “A new wind turbine health condition monitoring method based on VMD-MPE and feature-based transfer learning”. In: *Measurement: Journal of the International Measurement Confederation* 148 (2019). ISSN: 02632241.

[47] A. Agasthian, Rajendra Pamula, and L. Kumaraswamidhas. “Fault classification and detection in wind turbine using Cuckoo-optimized support vector machine”. English. In: *Neural Computing and Applications* 31.5 (2019), pp. 1503–1511. ISSN: 0941-0643.

[48] D Yu et al. “A radically data-driven method for fault detection and diagnosis in wind turbines”. English. In: *International Journal of Electrical Power and Energy Systems* 99 (2018), pp. 577–584. ISSN: 0142-0615.

[49] Sangryul Kim and Seo Yun-Ho. “Development of a Fault Monitoring Technique for Wind Turbines Using a Hidden Markov Model”. English. In: *Sensors* 18.6 (2018), p. 1790. ISSN: 14248220. URL: `http://search.proquest.com/docview/2108718625/`.

[50] Adrian Stetco et al. “Machine learning methods for wind turbine condition monitoring: A review”. English. In: *Renewable Energy* 133 (2019), pp. 620–635. ISSN: 0960-1481.

[51] Wenguang Yang, Chao Liu, and Dongxiang Jiang. “An unsupervised spatiotemporal graphical modeling approach for wind turbine condition monitoring”. English. In: *Renewable Energy* 127 (2018), pp. 230–241. ISSN: 0960-1481.

[52] Peter J. Brockwell and Richard A. Davis. *Time Series: Theory and Methods Second Edition*. Springer-Verlag New York, Inc., 1991.

[53] Saeed Aghabozorgi, Ali Seyed Shirkhorshidi, and Teh Ying Wah. “Time-series clustering - A decade review”. eng. In: *Information Systems* 53 (2015), p. 16. ISSN: 0306-4379.

[54] Hien Nguyen et al. “Maximum Pseudolikelihood Estimation for Model-Based Clustering of Time Series Data”. eng. In: *Neural Computation* (2017), p. 990. ISSN: 08997667. URL: `http://search.proquest.com/docview/1884823978/`.

[55] Peter J. Brockwell and Richard A. Davis. *Introduction to Time Series and Forecasting Third Edition*. Springer International Publishing Switzerland, 2016.

[56] Sheldon M. Ross. *Introduction to Probability Models 11th Edition*. Academic Press Elsevier Inc., 2014.

[57] Yu-Chia Hsu and An-Pin Chen. “A clustering time series model for the optimal hedge ratio decision making”. eng. In: *Neurocomputing* 138.C (2014), pp. 358–370. ISSN: 0925-2312.

[58] B Hulsegge and K.H de Greef. “A time-series approach for clustering farms based on slaughterhouse health aberration data”. eng. In: *Preventive Veterinary Medicine* 153 (2018), pp. 64–70. ISSN: 0167-5877.

[59] Jiechen Wang et al. “Relationship Between Urban Road Traffic Characteristics and Road Grade Based on a Time Series Clustering Model: A Case Study in Nanjing, China”. eng. In: *Chinese Geographical Science* 28.6 (2018), pp. 1048–1060. ISSN: 1002-0063.

[60] Gerrit Bode et al. “A time series clustering approach for Building Automation and Control Systems”. eng. In: *Applied Energy* 238 (2019), pp. 1337–1345. ISSN: 0306-2619.

[61] H. He and Y. Tan. “Pattern Clustering of Hysteresis Time Series with Multivalued Mapping Using Tensor Decomposition”. In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 48.6 (2018), pp. 993–1004. ISSN: 21682216.

[62] Omid Motlagh, Adam Berry, and Lachlan O’Neil. “Clustering of residential electricity customers using load time series”. eng. In: *Applied Energy* 237 (2019), pp. 11–24. ISSN: 0306-2619.

[63] Hailin Li. “Multivariate time series clustering based on common principal component analysis”. eng. In: *Neurocomputing* 349 (2019), pp. 239–247. ISSN: 0925-2312.

[64] Thelma Safadi. “Using independent component for clustering of time series data”. eng. In: *Applied Mathematics and Computation* 243.C (2014), pp. 522–527. ISSN: 0096-3003.

[65] Jafar Rahmanishamsi, Ali Dolati, and Masoudreza Aghabozorgi. “A Copula Based ICA Algorithm and Its Application to Time Series Clustering”. eng. In: *Journal of Classification* 35.2 (2018), pp. 230–249. ISSN: 0176-4268.

[66] H Zaifoglu, B Akintug, and A. M Yanmaz. “Regional Frequency Analysis of Precipitation Using Time Series Clustering Approaches”. eng. In: *Journal of Hydrologic Engineering* 23.6 (2018). ISSN: 1084-0699.

[67] Joao Francisco Barragan, Cristiano Hora Fontes, and Marcelo Embirucu. “A wavelet-based clustering of multivariate time series using a Multiscale SPCA approach”. eng. In: *Computers and Industrial Engineering* 95 (2016), pp. 144–155. ISSN: 0360-8352.

[68] Yongkyu Lee, Jonggeol Na, and Won Bo Lee. “Robust design of ambient-air vaporizer based on time-series clustering”. eng. In: *Computers and Chemical Engineering* 118 (2018), pp. 236–247. ISSN: 0098-1354.

[69] D. Muruga Radha Devi and P. Thambidurai. “Similarity measurement in recent biased time series databases using different clustering methods”. In: *Indian Journal of Science and Technology* 7.2 (2014), pp. 189–198. ISSN: 09746846.

[70] Narges Shafieian. “A Novel Method for Transforming XML Documents to Time Series and Clustering Them Based on Delaunay Triangulation”. eng. In: *Applied Mathematics* 6.6 (2015), pp. 1076–1076. ISSN: 2152-7385. URL: `http://search.proquest.com/docview/1718957579/`.

[71] Pedro C. Alvarez-Esteban, Carolina Euan, and Joaquin Ortega. “Time series clustering using the total variation distance with applications in oceanography”. In: *Environmetrics* 27.6 (2016), pp. 355–369. ISSN: 1180-4009.

[72] Jorge Caiado, Nuno Crato, and Pilar Poncela. “A fragmented-periodogram approach for clustering big data time series”. eng. In: *Advances in Data Analysis and Classification* (2019), pp. 1–30. ISSN: 18625347. URL: `http://search.proquest.com/docview/2239964127/`.

[73] Jane Harvill, Priya Kohli, and Nalini Ravishanker. “Clustering Nonlinear, Nonstationary Time Series Using BSLEX”. eng. In: *Methodology and Computing in Applied Probability* 19.3 (2017), pp. 935–955. ISSN: 1387-5841.

[74] Jiancheng Sun. “Clustering multivariate time series based on Riemannian manifold”. eng. In: *Electronics Letters* 52.19 (2016), pp. 1607–1609. ISSN: 0013-5194. URL: `http://search.proquest.com/docview/1845816549/`.

[75] Z. Izakian and M. Mesgari. "Fuzzy clustering of time series data: A particle swarm optimization approach". eng. In: *Journal of Artificial Intelligence and Data Mining* 3.1 (2015), pp. 39–46. ISSN: 2322-5211.

[76] Ibgtc Bowala and Mgnas Fernando. "A novel model for Time-Series Data Clustering Based on piecewise SVD and BIRCH for Stock Data Analysis on Hadoop Platform". eng. In: *Advances in Science, Technology and Engineering Systems* 2.3 (2017), pp. 855–864. ISSN: 2415-6698.

[77] Yongqiang Tang et al. "Tensor Multi-Elastic Kernel Self-Paced Learning for Time Series Clustering". eng. In: *IEEE Transactions on Knowledge and Data Engineering* (2019), pp. 1–1. ISSN: 1041-4347.

[78] Saeed Aghabozorgi and Teh Wah. "Clustering of large time series datasets". eng. In: *Intelligent Data Analysis* 18.5 (2014), pp. 793–817. ISSN: 1088-467X. URL: `http://search.proquest.com/docview/1620092812/`.

[79] Saeed Aghabozorgi and Teh Wah. "Approximate Clustering of Time-Series Datasets using k-Modes Partitioning". eng. In: *Journal of Information Science and Engineering* 31.1 (2015), pp. 207–228. ISSN: 1016-2364. URL: `http://search.proquest.com/docview/1686443066/`.

[80] Maria Ruiz-Abellon, Antonio Gabaldon, and Antonio Guillamon. "Dependency-Aware Clustering of Time Series and Its Application on Energy Markets". eng. In: *Energies* 9.10 (2016), p. 809. ISSN: 19961073. URL: `http://search.proquest.com/docview/1831861660/`.

[81] Francesco Finazzi et al. "A comparison of clustering approaches for the study of the temporal coherence of multiple time series". eng. In: *Stochastic Environmental Research and Risk Assessment* 29.2 (2015), pp. 463–475. ISSN: 1436-3240.

[82] Cassio M.M Pereira and Rodrigo F de Mello. "Persistent homology for time series and spatial data clustering". eng. In: *Expert Systems With Applications* 42.15-16 (2015), pp. 6026–6038. ISSN: 0957-4174.

[83] Pierpaolo D'Urso, Livia De Giovanni, and Riccardo Massari. "GARCH-based robust clustering of time series". eng. In: *Fuzzy Sets and Systems* 305 (2016), pp. 1–28. ISSN: 0165-0114.

[84] Sipan Aslan, Ceylan Yozgatligil, and Cem Iyigun. "Temporal clustering of time series via threshold autoregressive models: application to commodity prices". eng. In: *Annals of Operations Research* 260.1-2 (2018), pp. 51–77. ISSN: 0254-5330.

[85] Pierpaolo D'Urso, Livia De Giovanni, and Riccardo Massari. "Time series clustering by a robust autoregressive metric with application to air pollution". eng. In: *Chemometrics and Intelligent Laboratory Systems* 141 (2015), pp. 107–124. ISSN: 0169-7439.

[86] Pierpaolo D'Urso et al. "Autoregressive metric-based trimmed fuzzy clustering with an application to PM10 time series". eng. In: *Chemometrics and Intelligent Laboratory Systems* 161 (2017), pp. 15–26. ISSN: 0169-7439.

[87] Yongping Zeng et al. "Fuzzy clustering of time-series model to damage identification of structures". eng. In: *Advances in Structural Engineering* 22.4 (2019), pp. 868–881. ISSN: 1369-4332.

[88] Edoardo Otranto and Massimo Mucciardi. "Clustering space-time series: FSTAR as a flexible STAR approach". eng. In: *Advances in Data Analysis and Classification* 13.1 (), pp. 175–199. ISSN: 1862-5347.

[89] S. Deb. "VAR Model Based Clustering Method for Multivariate Time Series Data". eng. In: *Journal of Mathematical Sciences* 237.6 (2019), pp. 754–765. ISSN: 1072-3374.

[90] Jose Dias, Jeroen Vermunt, and Sofia Ramos. "Clustering financial time series: New insights from an extended hidden Markov model". eng. In: *European Journal of Operational Research* (2015), p. 852. ISSN: 03772217. URL: http://search.proquest.com/docview/1664477966/.

[91] Alvaro Gomez-Losada, Jose Carlos M Pires, and Rafael Pino-Mejias. "Time series clustering for estimating particulate matter contributions and its use in quantifying impacts from deserts". eng. In: *Atmospheric Environment* 117.C (2015), pp. 271–281. ISSN: 1352-2310.

[92] Shima Ghassempour, Federico Girosi, and Anthony Maeder. "Clustering multivariate time series using Hidden Markov Models". eng. In: *International journal of environmental research and public health* 11.3 (2014), pp. 2741–2763. ISSN: 1660-4601. URL: http://search.proquest.com/docview/1510403562/.

[93] Binoy B Nair et al. "Clustering stock price time series data to generate stock trading recommendations: An empirical study". eng. In: *Expert Systems With Applications* 70 (2017), pp. 20–36. ISSN: 0957-4174.

[94] Joao A Bastos and Jorge Caiado. "Clustering financial time series with variance ratio statistics". eng. In: *Quantitative Finance* 14.12 (2014), pp. 2121–2133. ISSN: 1469-7688. URL: http://www.tandfonline.com/doi/abs/10.1080/14697688.2012.726736.

[95] Marta Disegna, Pierpaolo D'urso, and Fabrizio Durante. "Copula-based fuzzy clustering of spatial time series". eng. In: *Spatial Statistics* 21 (2017), pp. 209–225. ISSN: 2211-6753.

[96] Leonardo N Ferreira and Liang Zhao. "Time series clustering via community detection in networks". eng. In: *Information Sciences* 326.C (2016), pp. 227–242. ISSN: 0020-0255.

[97] Lihua Zhou et al. "Clustering Multivariate Time Series Data via Multi-Nonnegative Matrix Factorization in Multi-Relational Networks". eng. In: *IEEE Access* 6 (2018), pp. 74747–74761. ISSN: 2169-3536.

[98] Sergio Martin-del-Campo, Fredrik Sandin, and Daniel Strömbergsson. "Dictionary learning approach to monitoring of wind turbine drivetrain bearings". In: (2019).

[99] Alberto Pliego Marugan, Ana Maria Peco Chacon, and Fausto Pedro Garcia Marquez. "Reliability analysis of detecting false alarms that employ neural networks: A real case study on wind turbines". English. In: *Reliability Engineering and System Safety* 191 (2019), p. 106574. ISSN: 0951-8320.

[100] Tianyang Wang et al. "Vibration based condition monitoring and fault diagnosis of wind turbine planetary gearbox: A review". English. In: *Mechanical Systems and Signal Processing* 126 (2019), pp. 662–685. ISSN: 0888-3270.

[101] Karen Kazor and Amanda Hering. "Assessing the Performance of Model-Based Clustering Methods in Multivariate Time Series with Application to Identifying Regional Wind Regimes". eng. In: *Journal of Agricultural, Biological, and Environmental Statistics* 20.2 (2015), pp. 192–217. ISSN: 1085-7117.