Predictive Maintenance: A Practical Perspective

Henrik Hviid Hansen[1,2]

December 2021

[1] Ørsted A/S,
Bioenergy, Digital Products and Processes

[2] Technical University of Denmark,
Department of Applied Mathematics and Computer Science,
Section for Statistics and Data Analysis

CONTENTS

1 INTRODUCTION

For as long as there have been humans, we have found ways of automating or improving our lives using tools we have created for a very specific purpose. Such tools could be simple devices or tools used by our hunter-gatherer ancestors to help them be more efficient in their daily tasks. It could also be complicated machinery at a modern power plant. Common to both is the wear and tear that such equipment is exposed to causing deterioration in performance and eventual breakdown. For this reason, maintenance of some form is eventually needed. The specifics of the particular maintenance task are beyond the scope of this work, as we are here more interested in how to determine *when* maintenance is needed.

Predictive maintenance (PdM), as the term is coined, is the newest approach that tries to solve this problem using process or performance data obtained from the equipment in question. Using the obtained data, mathematical models are typically used in trying to make sense of what the data represents; does it indicate that all is well, or is the machine on the brink of catastrophic failure? In practice, the problem is usually not as black and white as portrayed here and often it can prove difficult to make inference with such certainty. For the rest of this paper, we will delve into some high-level considerations associated with predictive maintenance (PdM) as well as advantages and challenges, prerequisites, other types of maintenance schemes. We will not delve deeply into the planning aspect, but simply acknowledge that it becomes a relevant task at some point when the PdM system has proven itself capable of useful predictions and enough historical data exists.

The practical perspectives in this paper are based primarily on the author’s experience from working with PdM for large scale power plant machinery within a large energy organization. Other more abstract parts are based on various academic literature. A good general introduction to the field can be found in both [18] and [19], which discuss both low as well as high-level aspects of introducing, building and operating a PdM system in an organization.

The rest of the paper is structured as follows. First, we introduce and discuss different maintenance philosophies. Then we delve more into the different aspects of PdM followed by a discussion of the requirements for practical PdM systems. Then we will look into some practical considerations of how the data flow and process should be structured, as well as a discussion on the organizational requirements for the adoption of a data-driven maintenance system. We finish with some thoughts on the mathematical modelling aspects and which models are useful for PdM before we conclude the paper. The last part can be skipped if you are in a hurry.

2 MAINTENANCE PHILOSOPHIES

In dealing with equipment that deteriorates over time it is inevitable and obvious that it needs to be maintained sooner or later in order to keep it running. But *when* should we maintain? This question does not have an obvious answer unless one is willing to postpone maintenance until the equipment breaks down, at which point maintenance becomes the only option to resume operation. This maintenance philosophy is usually referred to as *run-to-failure* (RTF). For equipment vital to industrial processes, this type of maintenance will usually not be a viable strategy as equipment might become very expensive to repair once fully broken, not to mention the lost production, which is usually the biggest concern. Therefore, some other way of determining maintenance intervals is needed. A good overview of maintenance philosophies can be found in both [3] and [5].

2.1 PREVENTIVE MAINTENANCE

The simplest way to maintain equipment is to maintain, repair or replace parts *before* it becomes necessary. This is called *preventive maintenance* and also referred to as *time-based maintenance* (TBM). It is based on elapsed time or some other measure of accumulated use, e.g., the number of kilometres a car has driven or the number of flights (cycles) an airplane has sustained since last it was maintained.

Traditionally, this has been the common way to plan maintenance, as it requires no effort other than keeping track of the accumulation of the appropriate number, such as kilometres, run hours, flights or number of uses. In almost any case, the prescription of maintenance after a fixed number has been reached is based on recommendations from the manufacturer who is assumed to have extensive knowledge of their equipment and may have gone to great lengths to determine appropriate maintenance intervals.

However, it can be viewed as a conservative approach, as the manufacturer's recommendations must ensure a very high proportion of their equipment will still be functional when the recommended time of maintenance is reached. Therefore, these recommendations are likely to be overly cautious and will often recommend maintenance much too early rather than a little too late. If no way of measuring degradation exists, this is a perfectly viable strategy and will in any case be better than the run-to-failure (RTF) approach for vital machinery.

In many cases, it is therefore quite likely that maintenance happens too early and too often versus what is actually needed to keep the equipment from breaking down. This results in excessive maintenance costs and unnecessary replacement of components which may still have a long *remaining useful life* (RUL). This maintenance strategy can therefore be thought of as over-maintaining. Although, as it requires very little effort and is easy to follow, it is applied ubiquitously in almost any place where maintenance is needed.

2.2 CONDITION BASED MAINTENANCE

Condition based maintenance (CBM), as PdM is also sometimes referred to, is perhaps the most meaningful way to determine the time of maintenance, as it prescribes maintenance when it is needed based on some condition indicator or other criterion. In its most simple form, it is the idea of neither maintaining too much, nor too little, but just enough and just in time (which it is sometimes referred to), contrary to TBM.

Broadly, there are three parts to this. First, the user must monitor the component in question to keep track of its current condition as faults can develop gradually over time or rapidly depending on the type of component and fault. In practice, this is a difficult problem and requires a good physical understanding of the component as well as how to properly analyse the data collected from it. There are many ways to go about this using different types of data and mathematical models, but common to all, is that anomalous data patterns should be detected in the case of a fault happening. This practice is commonly known as *condition monitoring* (CM) and draws heavily on the data mining field of anomaly detection.

The second part is the long-term prescription of maintenance based on the probability that the machine's condition will have deteriorated enough to justify taking it out of service to maintain it. This part is often the most difficult in practice as it requires extensive historical examples of a given type of fault. In the case that such data is available, the machine will in some cases have changed (parts have been replaced/fixed/adjusted) enough over time, that in practice it is no longer the same machine. Thus, the historical incidents will be of lesser use in modelling the probability of a future incident.

In the case of several identical machines, this becomes a more feasible problem to tackle as historical failure patterns will likely be similar across machines. This enables a faster accumulation of a useful failure history.

Third and lastly, the whole purpose of trying to infer the component condition is being able to better plan maintenance. This is where the benefits of a good PdM system are reaped, as optimal planning of maintenance can minimise factors such as component downtime, lost production, repairs outside ordinary workhours and stress to the repair crew. Furthermore, spare parts can better be ordered ahead of time which ultimately could mean a smaller need for keeping a stock of spare parts which may eventually grow old or no longer match component specifications.

3 ASPECTS OF PREDICTIVE MAINTENANCE

In the previous chapter we mentioned several different aspects of PdM, which together form a whole system for keeping components maintained, while maximizing the remaining useful life (RUL). The first part is monitoring the current condition of the component. Once this is in place, we would like a PdM system to be able to infer the right time to maintain or replace the component based on inference from historical fault patterns. Lastly, we should be able to better plan maintenance times in the future based on the information gathered from our short and long-term monitoring systems. We will delve into these three aspects in this chapter.

3.1 CONDITION MONITORING

To properly infer about component condition, one must be able to not only forecast the future, but also monitor the current state of the component. At the time of writing, many industrial PdM users are at the condition monitoring stage. That is, they are somewhat able to monitor for faults developing in real time, but much less able to forecast long-term maintenance needs. This makes sense, as here-and-now monitoring is intuitively a more manageable task with greater likelihood of correct inference, as uncertainty grows the longer the forecast horizon.

However, in practice it is not necessarily an easy task, as it often requires good knowledge of physical aspects of the component which may be heavy industrial machinery, e.g., pumps and motors are two very common examples, or simpler equipment such as rollers driving a conveyor belt. Common to both is that something must be known about the physical task that the component solves to model its behaviour – an expectation of what normal behaviour looks like. This can be obtained from a combination of sensor data and expert knowledge of the component.

In the case of a simple system, like the roller, the task can be relatively straightforward, as a single measurement such as a temperature can in many cases be enough to determine the operating condition (faulty or not). On the other hand, if a complex system like a heavy industrial pump is to be inferred about, it may not make sense to talk about a single maintenance state. Rather, as the system is put together of many smaller components, it will often lead to more accurate inference to treat the problem as such. That is, modelling the

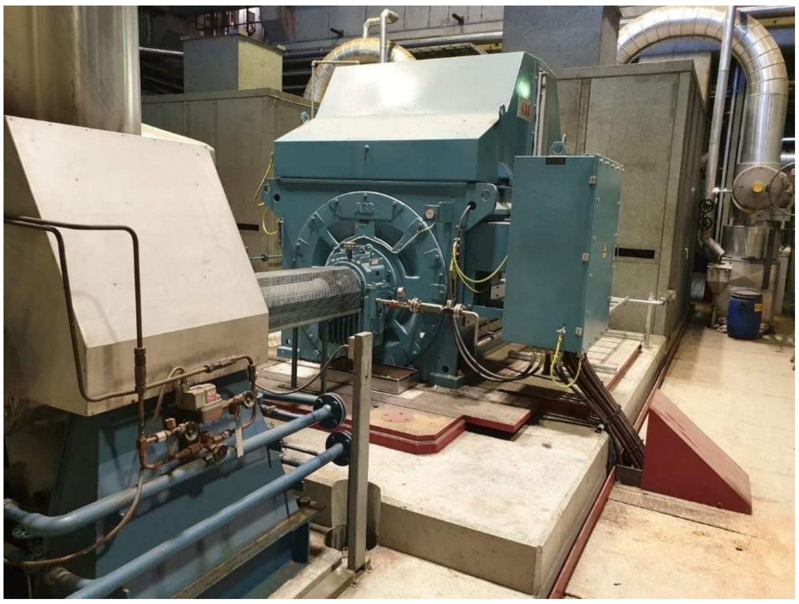


Figure 1: a large feedwater pump at a Danish combined heat and power plant. Source: Jonas Kim Fausing, 2020.

smaller components on their own rather than the whole system as one. This makes sense as the smaller components are the functioning parts of the machine and therefore will wear out over time.

This kind of subdivision of a complex system reduces the problem to modelling a simpler system. However, this still requires some knowledge of each component as to how they are expected to fail and how this may change the patterns of the data obtained from it. Despite this, inference is likely to be better and the modelling task more manageable.

Despite having divided the system into smaller parts and therefore being better able to monitor for faults, it may still prove challenging to narrow down the cause of the fault. This is especially the case if there is likely to be substantial correlation between the behaviour of several of the smaller components. This is the case since they work together to solve a bigger task in the complex system. Consequently, this could lead to a damaged component underperforming, successively affecting other components in the system. The damaged component is then camouflaged by the other components due to the correlation between them making it difficult to determine the root cause of the bad performance.

These are some of the challenges of monitoring the condition of a component, which show why condition monitoring may not be as straightforward at second glance. It also makes it clear why there is a need for domain knowledge about any system that one hopes to monitor. Finally, the right sensor data helps tremendously, since for example one cannot hope to quickly detect a broken pump impeller if only temperature measurements are available. The right data sources for particular fault conditions are often what makes the fault detectable or not.

3.2 PREDICTING FUTURE COMPONENT FAULTS

The second part of any whole PdM system is the ability to predict future breakdowns. This is the ability to analyse past failure patterns and determine how the probability of breakdown increases over time and according to what probability law.

When we say predicting breakdown, what is really meant is predicting the remaining useful life (RUL), as was introduced earlier. The idea of RUL is the assumption that at any given point in time a component has a predetermined number of hours/cycles/kilometres etc. remaining for which it will perform satisfactorily. Once this number is exceeded, we say that the component no longer has any RUL, since it can no longer carry out its task in a satisfactory way. Ideally, when the component reaches this point, it is the optimal time to replace it since its utility value has then been maximized over the course of its life. In other words, we avoid discarding a well working component with a lot of useful time left. Although, in practice we would prefer to replace the component a short while before its RUL is depleted to avoid a potential breakdown or damage to other parts when the component no longer works. It is this quantity that we would like to estimate in inferring about future events and when to maintain. Usually, this is difficult due to several challenges.

Starting off with the need to infer about future maintenance points (points in time when maintenance should optimally be carried out according to some rule), this is not a task that can be solved by merely observing the current condition of the component. Indeed, faults and slow deterioration is likely not detectable either visually or via sensor data until a short time before the component breaks down. Thus, trying to infer about component condition long into the future is not a feasible task only by observing the current condition.

Therefore, one must look at past failures and try to uncover the governing mechanism behind the failure patterns. This is usually in the form of a probability model, which can then associate with each future time point an estimate of the RUL for the component. When a certain lower threshold on the estimated RUL, i.e., when close enough to failure, has been reached the maintenance should then be carried out to avoid a breakdown.

The main challenge to this is typically a lack of past examples of failure, that is, when the true RUL has reached 0. The reasons for this are as follows.

First, the equipment may be critical to the broader operation of a complex system and has therefore routinely been maintained with fixed intervals, thus never or only rarely allowing a failure to occur. In the case of several identical machines, this problem can be somewhat alleviated by assuming they share the same governing failure mechanisms and then pooling the past failure examples to estimate a model.

Secondly, in the case of a completely new machine, naturally there is no failure history.

And lastly, the machine may have a failure history, but there may not be enough to estimate a very accurate model, in which case the model predictions will not be useful as they cannot be trusted.

Beyond the reasons just stated, another challenge is if failure examples are not documented properly. The time of failure as well as a detailed description of the nature of the failure are necessary to have, and are at less risk of disappearing, growing outdated or corrupted if kept in a structured and maintained database of historical incidents. This is an indicator of maintenance maturity of the organization hoping to employ PdM and is an important prerequisite to have in place before a useful RUL model can be estimated.

In the case that nothing is known about past failures (the machine might be completely new), the PdM user could turn to simulation of the system, if it is well understood. Pumping systems, motors, turbines, and similar rotating equipment are all governed by well-known physical relations. If these relations are known, a simulation model might be possible to make and could potentially help in shedding light on failure patterns. However, this requires vast amounts of knowledge of not only the physical relations, but also the type of disturbances and wear that it is exposed to over time. Not to mention the programming expertise required to design such a model.

The challenges described here are some of the major bottlenecks to most potential PdM users and serves in understanding why many users are stuck at condition monitoring of their essential equipment.

3.3 PLANNING MAINTENANCE

The third and last aspect of a PdM system is the planning. This is the ultimate goal that the PdM philosophy tries to achieve; better planning of future maintenance.

For the reasons stated in the previous two sections, not many PdM users have reached this level as it requires good estimates and faith in the RUL models estimated from past failures. In the case of a good RUL model, for critical equipment it may still not be deemed economically rewarding enough to stop periodic maintenance, simply due to the risk of the model overestimating the RUL and thus driving the equipment to failure.

However, assume that proven and trustworthy RUL and CM models are in place and that economical rewards are substantial enough to trust the predictions. The planning part is then the next natural step and the benefits are many: Planning the maintenance allows operators to work under less stressful conditions, spare parts be ordered ahead of time, minimization of down time and lost production, and more.

Planning of maintenance of a single or a few machines is not in itself a difficult task and is likely most easily done manually by a human planner. Therefore, if we further assume that many machines, possibly a whole industrial plant, are covered by proven PdM systems, optimal planning of maintenance becomes a more complicated task that might best be solved as an optimization problem. This is the field known as operations research. The problem of solving big optimization problems is covered extensively in [3] framing the problem to be solved as a Markov Decision Problem, which can be solved using dynamic programming methods.

4 DATA IN AN INDUSTRIAL PREDICTIVE MAINTENANCE SYSTEM

In this chapter, we delve into practical considerations revolving around the data, that organizations looking to employ PdM must have in place to get the most value.

4.1 SENSOR DATA REQUIREMENTS

The first thing to consider if one is to build and run a PdM system, be it for a single or an array of components, is data. Sensor data from the component to be monitored is the senses of the PdM system (as the eyes and ears of a human operator). Generally, there are several aspects to data that are important to consider: the right type of data for the component and the problem, the right amount as well as the quality of it. For this chapter, we will assume that sensor data from a given component exists as a discretized representation of a time-dependent continuous signal, that is, a time-series dataset.

4.1.1 Data types

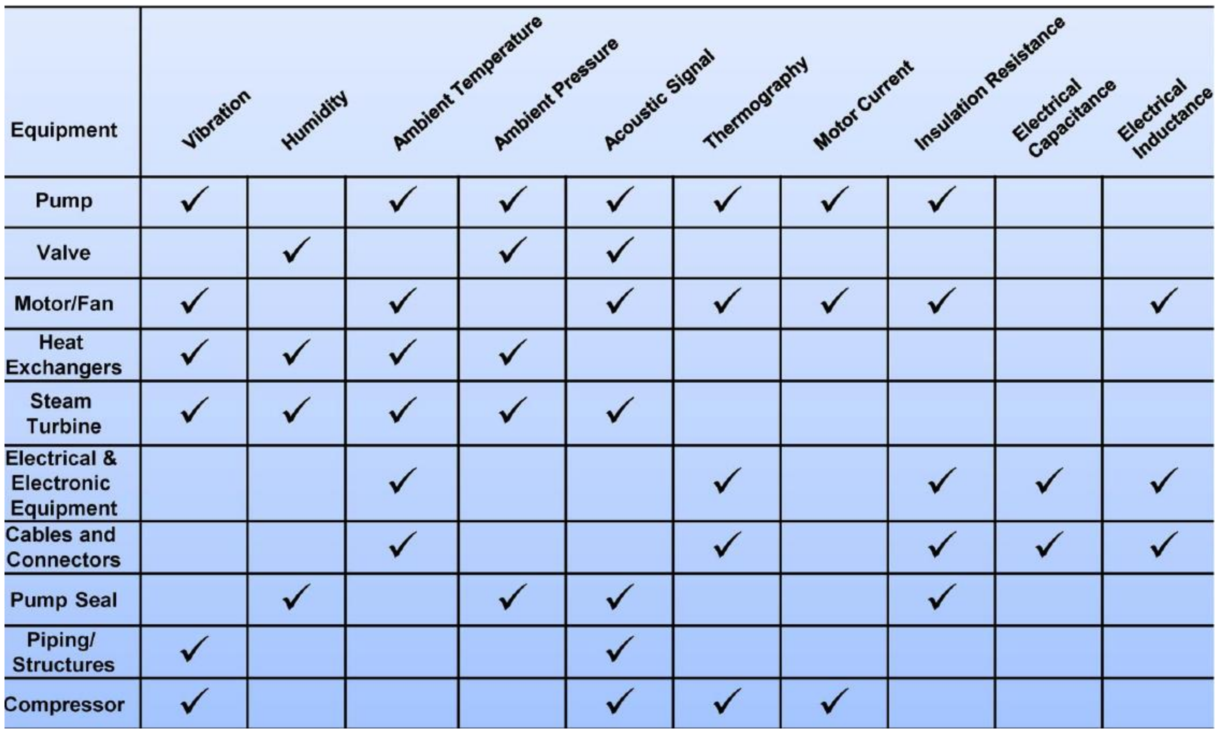
The type of data required to monitor component condition is perhaps the most important point to consider. Indeed, certain measurements are more appropriate for some equipment, processes, and fault types than others.

For rotating equipment such as motors, pumps, turbines, fans, etc., vibration and acoustic measurements will often prove very useful in detecting faults early on. This is due to small changes in the inner workings of the machine giving rise to slightly altered frequencies on which the equipment will tend to operate. This is generally the case for any moving parts and especially where bearings are located.

Temperature measurements can also prove useful in detecting high friction in moving parts as a result of poor lubrication (e.g., missing or wrong type of oil) as this will lead to excessive temperatures.

The above points serve as examples and should make clear the need to know the equipment and its main operating mechanisms. With such knowledge, it is in many cases possible to form expectations of what might happen in the case of a specific fault. For example, in the case of temperatures on ball bearings, one would expect a clear upward linear trend compared to the normal operation.

Therefore, it is important to consider if the right sensor data for detecting specific faults is available before trying to set up a monitoring system. If not, the system will in the best case not detect anything, and in the worst case it will detect a substantial number of false positives. That is, it will erroneously flag normal operation of the component as faulty. This can lead to the phenomenon known as *alarm fatigue*, which will be discussed in more detail later.



Equipment	Vibration	Humidity	Ambient Temperature	Ambient Pressure	Acoustic Signal	Thermography	Motor Current	Insulation Resistance	Electrical Capacitance	Electrical Inductance
Pump	✓		✓	✓	✓	✓	✓	✓		
Valve		✓		✓	✓					
Motor/Fan	✓		✓		✓	✓	✓	✓		✓
Heat Exchangers	✓	✓	✓	✓						
Steam Turbine	✓	✓	✓	✓	✓					
Electrical & Electronic Equipment			✓			✓		✓	✓	✓
Cables and Connectors			✓			✓		✓	✓	✓
Pump Seal		✓		✓	✓			✓		
Piping/ Structures	✓				✓					
Compressor	✓				✓	✓	✓			

Figure 2: table of measurements related to specific equipment condition. Source: "State-of-the-Art Predictive Maintenance Techniques", Hashemian, H.M., 2011.

4.1.2 Data sampling frequency

Another practical consideration for the data is the amount of it coming through the PdM system. Assume that there is a continuous signal of some measurement appropriate to the specific task. The overall question is then how often this signal must be sampled to obtain a discrete representation that is faithful enough to the true continuous signal to allow detection of significant changes over time.

One might argue that the more samples per time unit the better (increased sampling rate), as this is the discrete signal that is most true to the continuous original. However, the information gain from a higher sampling rate, is not always significant and may only serve to increase the computational and storage burden. Indeed, for most practical applications, there are diminishing returns to increasing the sampling rate, as the physical state of the component can only change so much in a short amount of time. For this reason, there is an upper limit on the sampling rate beyond which, more samples may even negatively contribute to the overall task, as it simply becomes noise that must be filtered through. For a thorough discussion of optimal sampling rate, see p. 457 of [20].

Therefore, it is desirable to come as close as possible to the optimal sampling rate. This is not a trivial matter, however, and is in most cases only possible if prior examples of faults exist and it is known exactly when it started and ended. In this case, an analyst can then determine how often the signal should be sampled in order to best detect the fault using a suitable CM scheme.

Note, that this is certainly the ideal, and somewhat unrealistic, case, as 1) in practice it will often not be possible to determine with complete accuracy when the fault started developing or became detectable, and 2) there is very likely not one optimal sampling rate, but only one that is optimal for the specific combination of one particular fault and monitoring scheme. In other words, the optimal sampling rate may change across faults and monitoring schemes.

Additionally, there may be matters of sensor battery life to consider, which may itself put an upper limit on the sampling rate. If this is not an issue, however, the best practical approach to sampling may be to sample as often as possible, or at least a little too often, as there is a lower chance of missing some crucial change in the equipment, than if sampling too rarely.

With a little too many samples, it is possible and likely even desirable to calculate certain derived versions of the signal. This could simply be an aggregation of data points into (disjoint or not) time windows for which a summary statistic, e.g., the mean or median, is then computed. One could also simply consider only every fixed number of data points for analysis, effectively lowering the sampling rate by pre-processing of the already sampled signal.

Lastly, depending on the monitoring approach, the analyst should consider the amount of time-dependence (autocorrelation) in the signal. Some time-dependence is desirable, as it will make model verdicts about new data points better, but too much may negatively affect the monitoring in the form of too many false positives depending on the CM scheme.

4.2 HISTORICAL MAINTENANCE DATA REQUIREMENTS

As was discussed earlier in the section about planning of maintenance, it is desirable to have a well-kept history of past historical faults on the equipment in question. This is a strictly necessary requirement for estimating RUL models. It may be of less importance for CM, but is certainly useful, as it can give an idea of what types of faults to look for in the data, and thus customize models specifically towards these.

With past faults in hand, domain knowledge of the machine can be used to infer about what the data patterns might look like in the case of a specific type of already observed fault. If past sensor data exists from the point in time when the fault occurred, these patterns can be directly observed. This may not only aid in the understanding of the data pattern behaviour in the case of the fault but could also serve as input data for a monitoring scheme, training it to look out for certain known fault patterns. The more knowledge exists about the possible fault patterns of a given machine, the easier it is to detect future occurrences.

Finally, it should be clear by now, that the more examples of past faults exist, the better the monitoring and RUL estimation will be. To keep a well-structured history of past examples, it is worth having one common database for storing such records. Ideally, when a new example of a fault is recorded, certain checks should be made ensuring the quality of the record for future use. Some requirements to a single record could be, but is certainly not limited to:

- a properly detailed free-text description of the incident
- ensuring it is classified into suitable predetermined categories
- strict time point identifiers are associated with it

- estimation of severity, lost-production hours, economic costs etc.
- performance indicators such as run-hours, number of start/stops since last maintenance/breakdown/installation
- other appropriate metadata.

This practice ensures that the fault is easily searchable in the future and that similar fault types can be easily found and compared. These recommendations to historical fault examples remain valid regardless of the number of machines desired to have a PdM system in-place.

5 DATA FLOW AND PROCESS MANAGEMENT

Up until now we have discussed requirements on good sensor data as well as past maintenance records. This chapter deals with more high-level aspects and can be thought of as the broad strokes of how a complete PdM system could be organized and the data flow through it.

5.1 DATA FLOW AND VALIDATION SHOULD BE AUTOMATED

There are two parts to the data flow. The first part is the CM of the current condition of the component. The flow of data through the CM system could look something like this:

1. Sensors on the machines sample a data point every fixed number of time units
2. The data points are then sent either via a machine control system or directly to some database of raw data
3. Some basic transforms and validation checks are carried out (could be some form of averaging or otherwise if sampling rate is very high)
4. The data is then sent to some database of processed/transformed data, where it is ready for analysis.
5. A server with the condition monitoring models fetches the transformed data at fixed time intervals (could be every 10 min, 1 hours, 8 hours, etc) and carries out the classification into normal/faulty using the defined monitoring models.
6. The data deemed faulty by the CM system is then analysed by human eyes. This should preferably be by a domain expert (e.g., a machine engineer) as well as a data analyst.
7. If the data was correctly classified as faulty appropriate action should be taken. If the data was a false positive, it should be used to update the CM system to avoid flagging similar data as faulty in the future.

The steps above delineate an automated process where data flows through the system without human intervention until the last steps. This is necessary to achieve the full benefit of the CM system. This process is also shown as a flowchart in figure 3. The benefits of the automated process are not only that it is less prone to error, but also that constant monitoring is ensured, thus minimizing the time to detection of new faults occurring. Furthermore, it relieves data analysts to do other meaningful work such as exploring new monitoring schemes or otherwise optimizing the system.

As mentioned in the last point, if the CM system made an incorrect classification which is overruled by human eyes, it means the data was difficult for the model to classify correctly. Therefore, it should ideally be used to update/retrain the CM model to prevent the same misclassification of similar data in the future. There is an added benefit of keeping track of misclassified data, namely, that this builds up a well-classified dataset over time, which can be useful in devising and experimenting with new monitoring schemes. Furthermore, if other identical machines exist, it may even help the monitoring of these, as the data and faults are assumed to come from the same generating mechanism.

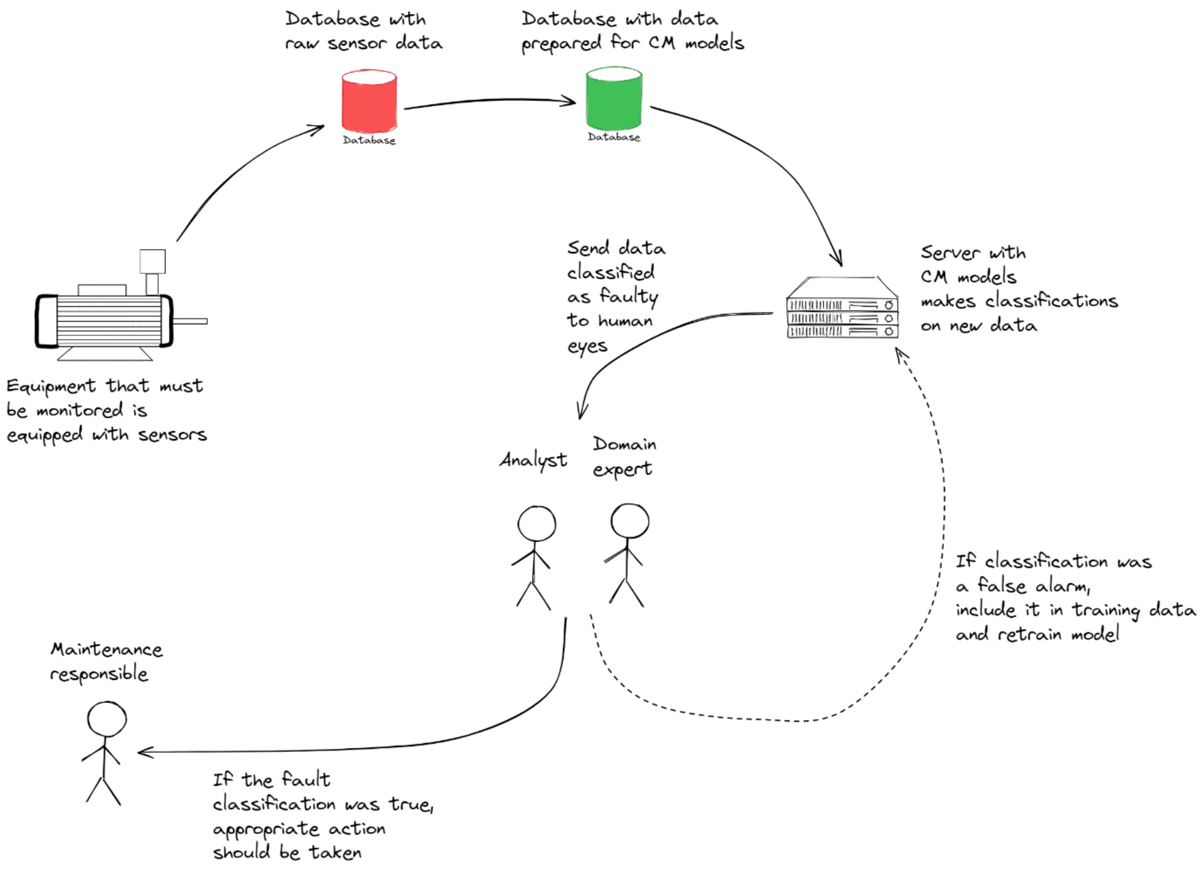

Database with raw sensor data
Database with data prepared for CM models
Database
Database
Server with CM models makes classifications on new data
Send data classified as faulty to human eyes
Equipment that must be monitored is equipped with sensors
Analyst
Domain expert
If classification was a false alarm, include it in training data and retrain model
Maintenance responsible
If the fault classification was true, appropriate action should be taken

Figure 3: flowchart of the sensor data's way through the PdM system, from machine to classification as a true or false alarm.

In the case of RUL estimation, this needs not be as automatic, as it should not take place continuously as in the case of CM. However, automation of this still comes with benefits, as it ensures that it is reliably carried out whenever relevant. Updating the RUL model should ideally happen every time a new fault example is added to the maintenance history. Even if the component is monitored by a CM system, the fault may not be detected and thus happen, or the fault is detected before it grows out of control and a root cause is determined. In either case, the fault can be added to the history of past incidents, and the RUL model should therefore be updated to make more accurate predictions.

5.2 RESPONSIBILITIES SHOULD BE CLEAR

Another important point to consider early on is the responsibilities of the people involved with the PdM system. For the highest impact of the PdM system, it should be made clear as early as possible who does what. The most essential roles involved with the PdM system are shown in table 1.

Who	Criticality / neccessity	Responsibility	Can overlap/combine with
Analyst / model developer / data scientist	Need to have	Designs and implements the mathematical models behind the PdM system with inputs from domain expert. Also analyses model classification output and general performance together with the domain expert	Data engineer
Data engineer	Need to have (can be less necessary in the beginning if these tasks can be solved by the analyst)	Makes sure data quality is satisfactory and ensures the stability of the automated data flow	Analyst / data scientist
Domain expert (SME)	Need to have	Finds, matures and describes new cases for monitoring by talking with component specific experts. Also assists the analyst in designing the models and analysing model output with physical knowledge of the equipment.	Maintenance responsible
Front-end / visualisation / dashboard developer	Nice to have (may become more necessary as maintenance solution scales up)	This person is focused solely on maximizing the interpretability and user-friendliness of the maintenance solution created by the team. Should be good at making useful data visualisations as well as dashboards showing them.	
Team/project leader	Need to have	Extroverted business-savy person who is good at promoting the maintenance solutions to the business. Very focused on increasing value and less on the technical aspects of what the team does.	
Equipment maintenance responsible	Need to have	The end-user. Receives the PdM system outputs and determines the appropriate action to take. Also gives feedback to the team as to how the outputs could be made better by e.g. showing specific or more information.	Domain expert

Table 1: The table shows an overview of the various responsibilities in operating an industrial PdM system.

Other roles may be involved with the overall functioning of the PdM system as well, but it is important to have defined at least the roles of table 1 early as possible. Usually, it is obvious who the analyst and data engineer are. However, the equipment/maintenance responsible person is the end-user of the PdM product, and as such, the value of the PdM system depends primarily on this person once the classifications are made. Additionally, it should be noted that for the system to work optimally, the domain expert and the maintenance responsible should ideally be working closely together and have the same goal in mind. This is due to the fact that the domain expert will help the analyst design the system and the outputs received from it, and the maintenance responsible will use these outputs for maintenance recommendations. The domain expert therefore needs to have the maintenance responsible person’s interests in mind in the design of the outputs.

5.3 INTERPRETABILITY IS IMPORTANT

An important aspect of PdM is the ability to understand how the data is used. This helps the non-data persons to understand the classifications better, which is likely to increase their trust in the system as well as overall use of it. Such personnel may not be accustomed to using data driven methods and may have primarily relied on manually sifting through relevant data. Consequently, classical visualization tools, e.g., scatter plots and time series plots, may well have been the primary means of understanding the data.

These points are important for the analyst as well, as a pure black box model is generally difficult to interpret. This makes it equally difficult to form and reject hypotheses of what may have triggered the model to classify data as faulty. In the end, this could lead to neither the analyst nor the maintenance person trusting the model’s classifications, as they cannot properly evaluate the output.

For these reasons, it is important to be able to understand and interpret model outputs. Some recommendations towards this, are to start with simple models and only increase complexity if necessary, to reach some desired level of classification skill. A common phenomenon at the time of writing is a focus on overly complex models due to the promises that come with them and their status as state-of-the-art models. In other words, the hype that surrounds the latest technology at the time.

If the PdM system is built around specific fault conditions, it is in many cases sufficient with simpler models, as knowledge of what the fault pattern looks like makes it possible to specify very problem specific thresholds.

Another recommendation is the continued use of the classical visualization tools of scatter and time series plots (among others). The intention of these is not to manually monitor all data as it comes through the system, but to diagnose specific model outputs when relevant. This would commonly be in the case where the model classifies data as representative of a faulty condition on the component. As mentioned in the responsibilities section, this should trigger the analyst and the domain expert to analyse the output and attempt to determine if it is a true or false positive. For this purpose, the visualization of data via plots is very useful since expectations of what a specific fault pattern will look like should have already been formed, if the system was designed to detect a certain fault condition.

5.4 False positives and alarm fatigue

An inevitable part of any system that tries to model a real system is wrong predictions. In the live application of the PdM system the number of false detections from the monitoring model must be low. This is due to the nature of the process, in that humans must analyse and determine the appropriate action on the output from the model. This means that if the system throws too many false alarms, the analyst and the domain expert could lose trust in the classifications over time. This results in a lower level of attention being given to each classification, which could in the worst case mean that a true detection might be overlooked.

Therefore, it is vital to control and reduce such false detections. There are some ways to go about this. First, designing the fault detection system to detect a very specific fault condition again pays off, in that more realistic and informed alarm thresholds can be set. For a universal model that is supposed to detect any deviation from normal, this is much more difficult as many different unknown fault conditions could exist, each varying in magnitude, data signature and so on. Therefore, domain knowledge of the equipment and its fault conditions allow the design of very focused detection models which often results in a reduced amount of false positives.

Secondly, it is inevitable that false positives will occur for any system as time progresses. Hence, it is essential to learn from these and update the system, so it will not flag a similar data pattern as a fault in the future. This is one of the reasons, why the faulty classifications from the model must be analysed by humans, so we can update our knowledge of the normal condition for the equipment. Once the false positive has been confirmed the data that was erroneously flagged as faulty must be included in the training dataset that was used to estimate the CM model parameters and the model retrained.

A third way of reducing false positives is to simply use less strict alarm thresholds. It is usually the case for any CM model, that a fixed threshold must be chosen as a binary cut-off value between what is deemed normal versus faulty (this could be over time or for each datapoint). This threshold is a hyperparameter of the model and cannot be learned without a good test dataset with past fault conditions to back test the model classifications for different thresholds. Such a dataset may not available, and therefore the threshold must be chosen in a more arbitrary way. In the case of prediction error models (more on these later), a value such as a certain percentile value, (e.g., the 95th, 99th, or even the maximum), of the training data prediction error can be useful. This value should then be chosen to ensure that new data is flagged as faulty only if the deviation from normal is substantial enough. However, this approach should not stand alone.

It remains to be said, that the three methods just described can and should be combined to get the best result.

5.5 How the PdM system creates value

Until now, we have discussed many data related details. We will now delve more on the business aspects of the PdM system.

There are two immediate things that must be given consideration in the creation of an industrial PdM system.

First, the *product* of the system should be decided upon. The product can be thought of as the process that converts raw sensor data into useful insights about the equipment under surveillance. In practical applications it may not be straightforward how this is achieved and what is understood by *useful insights*. To the analyst it may mean abstract things such as probabilities or model performance on some data, while to the domain expert and the maintenance responsible person, it may simply mean whether the machine is faulty or not. It is therefore necessary to define the PdM product and what comes out of it.

Ultimately, the end-product should be able to bring value and solve a problem and must therefore be tangible enough, that the insights can be understood and acted upon by the *end-user*, who we will discuss shortly. The product could be a dashboard of visualizations of the sensor data in the form of different useful plots. This is a good option since it also increases interpretability, which we discussed earlier. Model classifications could be visualized on the plots, making sure to bring attention to data classified as faulty. The visualizations also make the analysis of false positives easier as plotting the data is usually a good place to start with this.

When the PdM system has shown its worth and ability to make correct classifications over a period of testing, notifications to the control room or specific maintenance persons in the case of a fault detection could be an option. However, any detected fault should ideally still make it past human eyes before the end-user is alerted. Additionally, it is common practice to employ escalation levels, so that any detected fault must exceed several boundaries of increasing severity. Such boundaries could simply be as depicted in table 2.

Alarm level	**Severity**	**Action**
0	All good	Nothing
1	Low	Nothing
2	Medium	Observe
3	Critical	Maintain

Table 2: the table shows some recommendations for a possible escalation plan when faults are detected based on some measure of severity.

Another aspect is the *end-user*. This is the entity (person, department, control room, etc.) that must use the product, and can thus be thought of as the process that converts the useful insights from the product into actual value. Based on the recommendations from the PdM system, it must be decided what course of action to take (stop equipment immediately, wait a while and see or plan an optimal time of repair). The end-user should be clearly defined ahead of time, even before the PdM system is fully functional, as this ensures there is someone to realize the value that the PdM system can bring. As a last note, the output of the

PdM system / product should be decided upon in cooperation with the end-user, as this person is the one that must act on the recommendations and therefore can define what and how much information he needs in order to do so optimally.

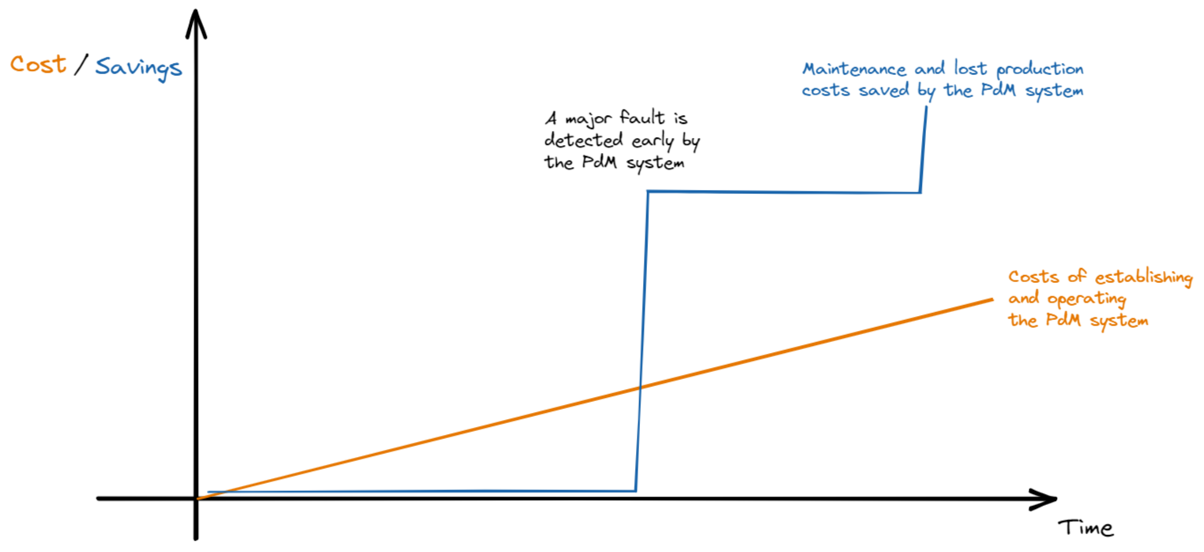


Figure 4: the graph shows how value from the PdM system (in the form of savings) develops over time versus the cost of establishing and operating the system.

Furthermore, it should be mentioned that contrary to the costs of establishing and operating the PdM system, the value (measured only by the economic savings) created by a functional PdM system is not linear over time. This is true, since critical faults will not develop often or constantly, but infrequently as time goes by. So long as there are no faults to detect, the system will not save any maintenance costs or minimize lost production. However, once a fault starts developing that could potentially become catastrophic if not detected early, the system will almost instantly have made major financial savings due to the fault not getting out of hand, optimal planning of down-time and replacement of parts. This is depicted visually in figure 4.

Finally, implementing and optimally using the PdM system is not just a data analysis or maintenance endeavour. Indeed, it is as much a matter of changing the way maintenance is done. Some old ways of maintaining will still be applied as the PdM system can and should not stand alone. But convincing an experienced maintenance department, that the PdM system will help them do their work and possibly replace some of their current practices requires time and a change in how maintenance is done and thought about. Thus, it is also a change management challenge that must be solved for a successful implementation of the system.

5.6 From Traditional to Predictive Maintenance

For many large industrial organizations, maintenance can be a conservative topic. Very experienced people will typically be overseeing the maintenance operations and indeed have a strong opinion about it. The adoption of a PdM system may be interpreted as a desire to replace these people with computers and automated monitoring. In most cases where a PdM system is introduced this is far from the ultimate goal of it, which in all its simplicity is better planning, higher machine availability and a better working environment for the maintenance personnel. Furthermore, the PdM system is only as good as the data being fed to it, and in the case of complex machinery, not every fault condition can be suitably detected via sensor data. Manual inspection and routine checks are therefore still necessary. It is thus apparent, that the PdM system cannot stand alone. An experienced and well-trained maintenance team is still required to maximize the availability of the machinery and thus the business value of the organization.

This is an important message to convey successfully, as it will help get some of the most important people, the maintenance personnel, to commit to helping achieve excellence with the PdM system, rather than fear for their job.

With this said, however, it should be noted that the process of adopting a modern PdM system is not something that is completed over night. Rather, it is an evolutionary process that takes time and effort. For the systems' successful adoption and optimal utilization in the organization, it requires full commitment and backing from the whole business. Such commitment to new data-driven ways may not be easily achieved and is largely a cultural change. This change should naturally start at the highest level (management) and propagate downwards, finally reaching the personnel carrying out the maintenance. This process takes time. Indeed, according to some estimates it may well take 3-5 years before the organization achieves noteworthy value from the system [18].

For a well-rounded discussion of these topics and much more, based on years of experience with maintenance solutions in industry, see especially Mobley's 2002 book, "*An Introduction to Predictive Maintenance"* [18].

6 MODELLING ASPECTS OF PREDICTIVE MAINTENANCE

For the final chapter, we will discuss some different modelling strategies and aspects in relation to a PdM system.

6.1 FEATURE ENGINEERING

We start with feature engineering. Briefly, feature engineering is the process of combining (e.g., adding, subtracting or dividing certain variables) or transforming (e.g., squaring, taking the logarithm or other linear/non-linear transforms) or scaling existing variables in the data to obtain stronger features with high correlation with some response variable of interest. It could also be extraction of new features using algorithms such as principal components analysis (PCA) or clustering. Other useful features in especially time-series data are the rate of change of some feature, extraction of parts of the date or categorical features based on logical rules about some of the other features.

Feature engineering is almost always useful, especially in the presence of domain knowledge which can be useful in determining appropriate combinations and transformations of the data. In essence, when looking for anomalies in the data, one tries to detect a signal that is especially powerful in revealing unusual data points. This is what feature engineering helps to accomplish. Indeed, it may be difficult if not impossible to detect any noteworthy deviations from normal with only the raw features of the data, yet easily visible with just the right combination or transformation of the data. This could be as simple as subtracting two features from each other or some other common operation.

As has been mentioned earlier, it is a good strategy to model specific fault conditions as opposed to trying to make one model that will detect everything. For this reason, it is also important to mention that strong features (raw or engineered ones) can play a vital part in the detection of specific faults. In fact, in some cases proper features with a strong correlation with the fault in question are all that is needed to detect unusual deviations from the normal condition. In other words, strong enough features could eliminate the need for a more advanced model. We will refer to such features as *fault indicator features*. In the simple case, a threshold on the particular indicator feature might be enough to signal that a fault is under development. In other cases, the inclusion of such features into a simple model (e.g., linear or logistic regression) could be enough to properly detect new developing faults.

Indeed, a simple thought experiment would be to imagine a quadratic relationship between a real-valued feature $\mathbf{x}$ and response $\mathbf{y}$. That is, $\mathbf{y} = \mathbf{x}^2$. We could model this non-linear relationship with a suitably powerful non-linear model such as a support vector machine (SVM) with a non-linear kernel. Or, we could choose a linear regression with a second order term. Both models would be able to capture the relationship, but a good general advice is to always opt for the simplest model, which in this case would be the linear regression, as it is more transparent and interpretable. There are other benefits to choosing the linear model in this case, which we will not delve into here.

However, we have demonstrated that a non-linear relationship can be appropriately modelled by a linear model with the right feature engineering, in this case the square transformation of the independent feature **x**.

One might also imagine that time dependence (autocorrelation) could be captured by the right set of features. It could simply be the inclusion of *time lagged features*, by which we mean simply taking a given feature and including the value of it one (or more) time unit back in time as a new feature in itself. Models based on this strategy and PCA have found use in areas such as statistical process control (SPC), see e.g., [9] for a discussion and application to simulated data.

In the presence of strong features – raw or feature engineered, with or without a model – the overall modelling task of monitoring the component becomes easier.

How to determine features useful for detecting a specific fault is situational and data dependent as well as based on domain knowledge of the problem at hand. However, if previous examples of the fault exist, these may be used as benchmark for testing the strength of each feature. This testing could be based on a combination of visual inspection of suitable plots and prediction error calculations. A brute force way of gauging the strength of each feature would be to simply have a model try to predict the fault and the normal data with and without each feature. Features whose inclusion yields much lower prediction errors are evidently the most useful ones according to the model.

Automated feature engineering libraries exist that could potentially make the feature engineering task somewhat easier, however, domain knowledge and other insights cannot be fully automated, and this task therefore remains mainly a manual one.

6.2 CONDITION MONITORING MODELS

In the pursuit of a system for monitoring the current state of a piece of equipment, some way of distinguishing data patterns from each other is needed. That is, data indicative of a healthy component versus data indicating that some fault has developed. This is the overall problem that must be solved in determining the component’s current state. To automate this task a mathematical model and processing of the data is needed. A field related to this task is the general field of *anomaly detection*, which has enjoyed much research over time, as it is a ubiquitous task in many different real-world applications. We will now discuss some common approaches to CM.

6.2.1 Simple models

Having discussed the importance of good feature engineering, we move on to simple models. More advanced models will be discussed later.

Some suggestions for simple ways to monitor a component are given here. It should be stressed that without some specific knowledge of what type of fault to look for or the right feature engineering in the form of a strong fault indicator feature, these models may not prove strong enough for the task. Also, if a great number of variables exist, it may be better to adopt another model more suited for multivariate problems.

The simplest model one can use is a fixed threshold based on data that represents a period of time in which the component was deemed healthy. A typical example of this is some integer multiple of the standard deviation or some percentile of the process, that future data must not exceed unless it represents some fault condition on the component. This type of modelling scheme is referred to as a Shewhart control chart and stems from the field of SPC [11]. In general, this strategy will prove too simple as it assumes that observations are independent and normally distributed, which are strong assumptions on real-world data that is often messy and strongly correlated through time. This causes the Shewhart chart to yield an excessive number of false alarms.

Generally, it is desirable to take the time-dependence into account somehow. A first suggestion is a simple moving average (MA) model. This could also be some other summary statistic of interest, e.g., the median or standard deviation. The idea is to capture the time dependence by aggregating a fixed past number of values, which also serves to stabilize the output statistic so random fluctuations (which are present in almost any real-world process) will not easily trigger a false positive.

This type of model is best used with a fixed threshold that the MA should not exceed unless there is an abnormal situation. The threshold could also be dynamic and thus depend on some external feature (e.g., weather temperature) in order to model the switching between different regimes over time. An alternative to this would be to take the first or second order difference of the time series and apply the MA model to, removing the need for a dynamic threshold. This is also a common strategy in classical time-series analysis.

The threshold for the model could be based on simply the standard deviation of the process in the training data, or it could be based on the prediction error between the forecasted and the observed value for each time step ahead.

The MA model usually comes with some modifications to adapt it better to the data at hand. One such modification is a weighting-scheme that gives decreasing importance to older data points, which is known as an exponentially weighted moving average (EWMA) in the SPC literature [11]**.** This modification makes the model faster at detecting deviations as they develop over time.

6.2.2 Time-series models

This class of models are perhaps not as simple as a fixed threshold, however, they are simple in the sense, that they are interpretable, unlike some of the later methods we will refer to.

We have already hinted at these models in the form of the moving average (MA) models mentioned earlier. This is a classical model within the time series literature and is often combined with *autoregressive* (AR) models, that we also briefly mentioned. Together, they form a very applied general model known as ARMA. A variant of this is the so-called *integrated ARMA* (ARIMA) model, which indicates an ARMA model fitted to the differenced time series data. The differencing is done to make the time series data stationary, which is an assumption of the ARMA model.

This classical time series model can be fitted to the data and forecasts can be done to predict values at future time steps. Uncertainty in the forecast is easily obtained since prediction intervals can be readily calculated. One can then compare predicted values with the observed, and if the latter fall outside the prediction interval, an anomalous situation might then be on the rise. This type of model may be best suited for a single or a few features (multivariate or vector ARIMA) as estimation of parameters can become infeasible for many features. For a good introduction and examples of these models, see [13].

6.2.3 SPC models

In the presence of many variables, the simple models will often not suffice. However, one should strongly consider if all these variables are in fact needed to properly monitor the desired equipment, or whether a subset of strong features – possibly obtained from feature engineering – can be used instead.

For several features it is often desirable to reduce the dimensionality by applying some transformation such as PCA, independent component analysis or even an autoencoder. These methods will attempt to extract new derived features that explain the other features of the data well. This can be seen as just another type of feature engineering.

After this is done, it is common to keep only a subset of the extracted features. A model based on this comes from the SPC field [7]. This particular model is based on PCA and instead of only retaining the strongest of the extracted features (the principal components or PC's) and discarding the rest, it uses all of them. The first n components deemed important are then monitored by the Hotelling T^2 statistic which is essentially just the multivariate distance from the mean scaled by the covariance matrix. The remaining PC's left out are monitored by another statistic often referred to as the Q statistic. Suitable thresholds can then be computed for both statistics based on the variance in the data from when the component was deemed healthy.

Together, these two statistics are computed for each observation over time. The T^2 statistic will then reveal changes in the mean whereas the Q statistic will tend to detect abnormal correlation patterns.

While this is a useful method when many features must be monitored, there are some drawbacks to this method. First and perhaps most importantly, it does not take into account the time dependence that there will almost surely be in the data. This could result in a high number of false alarms. Secondly, it does not capture non-linear feature relationships as it relies on PCA, which finds linear combinations of the features that maximize the data variation captured by the PC's.

Remedies to both of these problems have been suggested. A good way to tackle the time-dependence is by using the lagged PC's as features, which assumes that the data is an autoregressive process of some order and thus converts the time-dependence (autocorrelation) into feature correlations. This method is referred to as *dynamic PCA* and is discussed in greater detail with an application to simulated data in [9].

The second problem can be alleviated by the use of non-linear latent structure methods such as *autoencoders* instead of PCA. Other non-linear extensions of the classical PCA such as kernel PCA [14] are also viable alternatives.

The overall use of this method stems from the SPC field and thus also relies on control charts to monitor the T^2 and Q statistics. However, while human eyes observing these can strengthen the detection performance, the calculated thresholds will serve to alert whenever large enough deviations develop. The control charts are obvious candidates for a monitoring dashboard as discussed previously about interpretability of the PdM system.

Another fairly simple model is called the *Cumulative Sum* (CUSUM) control chart [11]. The idea is to compute for each time step of some quantity the (standardized) cumulated sum over time and tracking changes in this statistic as it evolves using predefined thresholds based on e.g., the standard deviation of the process. This takes into account the time-dependence by summing every data point up until the current one. It is similar in spirit to the Sequential Probability Ratio Test (SPRT), which is a bit more involved in that it computes the cumulative sum of the log-likelihood ratio of a statistic assuming some probability distribution [12].

6.2.4 Machine learning

First, for unsupervised problems, where little to no information about possible fault patterns exist, we have clustering algorithms, one-class classifiers (OCC) and isolation-based algorithms.

Clustering is a fairly standard approach for detecting anomalous points in a dataset. The intuition is to divide the data points into its *natural* clusters. A motor running at a manufacturing plant switching between to operating speeds would indicate two natural clusters. The clustering model is trained to recognize these two clusters from data representing a healthy motor. Over time, as the motor condition degrades, we might start to see outlying data points slowly forming a new cluster, separated from the two known ones. This approach can be useful and can benefit greatly from domain knowledge or some intuition about the equipment's number of natural clusters. However, the number of clusters is usually estimated from the data based on some selection criterion.

The clustering approach does not directly take into account the autocorrelation in the data, but is based on each new data point's distance from one of the natural clusters assuming a suitable distance metric (in many cases Euclidean). Some threshold must be set as cut off between normal and potentially anomalous points, which can be tricky if no prior examples of faults exist to estimate the threshold from. However, this remains a challenge for almost any model in PdM applications. Among noteworthy algorithms are k-means, DBSCAN and Archetype Analysis [6, 15, 16].

The other type of model for unsupervised problems is the one-class classifier (OCC). A popular variant is the one-class support vector machine [8]. The more recent one-class neural network classifier is also a popular choice [10]. In a sense, the OCC approach is a special case of the clustering approach mentioned above. This is true since it estimates a decision boundary between normal and anomalous given only normal data. Therefore, it essentially simplifies to a clustering with one single cluster considered normal where every new data point outside the decision boundary is considered anomalous.

As with clustering, there are generally two approaches to this; distance-based and density based. The distance-based approach was discussed already for clustering. The density approach is based on an estimation of a probability density of the normal data (this can be a computationally heavy task for many features and many data points). After this, new data points with a probability lower than some threshold according to the estimated density can then be considered anomalous. This cutting off based on probabilities can be a more intuitive way to set the threshold, however, it is still dependent on the data at hand. The best example of a density-based algorithm for the one-class approach is the kernel density estimator.

The third class of algorithms is said to be isolation-based, the best example being the Isolation Forest algorithm [4]. In this approach, classification trees are estimated and the number of branches (averaged over all the trees) required to isolate/classify each data point is then used as the anomaly score. If a given data point requires moving only one or a few branches down the tree, it is considered easy to isolate, and therefore possibly an anomaly, whereas normal points are difficult to isolate and thus require moving deep down the tree.

As with the other approaches, this does not directly take time into account, which could make other more time-based algorithms better choices if the data is highly autocorrelated.

6.2.5 Neural networks and deep learning

We now move on to another class of models typically known as neural networks. Deep learning models are considered a subset of these and are simply neural networks with deeper architectures (more layers).

One prominent model for anomaly detection among these is the *autoencoder* (AE). This model is a neural network which decreases the dimension of the data to extract the most important information in it and subsequently tries to take it back to the original dimension. In other words, it projects the original features into a lower dimensional latent space and back again into the original feature space. The idea with this is that the model will only reconstruct data points well if they are similar to data points seen in the training data. If not, the data point will be poorly reconstructed and thus have a high error (e.g. mean squared error or cross-entropy) when compared with the original data point. The reconstruction error is thus the anomaly score in this model.

The AE model is a very flexible and powerful model as it can model strongly non-linear interactions between many features. How well it reconstructs data points is a matter of finding the right parameters for the model. There are many parameters that could possibly be tuned for a neural network, but for this model, the most important ones are the number of layers and units, the so-called *activation function* for each unit, and the learning rate.

The downside is that it is more or less a black box, so debugging and analysing the outputs becomes quite difficult. Another downside is that in the standard edition (called *feed-forward*), it does not take time into account and therefore assumes every data point is independent, which is almost never the case in real-world PdM applications due to autocorrelated data.

However, more advanced variants exist known as *recurrent neural networks* (RNN) with the subclass *long short-term memory* (LSTM) networks being the most applied. These types of neural networks assume an autoregressive data structure and are thus capable of modelling the time-dependence in the data. When used for autoencoding these models are typically referred to as *sequence to sequence* (seq2seq) models, in that a time-sequence is taken as input and another time-sequence comes out, namely the reconstructed sequence. This type of AE should perform better than the feed-forward type if many features exist and autocorrelation is expected.

In general, autoencoders are very popular for anomaly detection for the reasons mentioned above and a great deal of research has gone into this yielding a large array of sophisticated modifications more or less powerful than the standard AE. As always, the best model is always data dependent.

6.2.6 Other types of analysis

Besides the analysis types already mentioned, other approaches exist that can be very useful for specific applications and machinery.

First, vibration analysis is considered to be one of the best ways to gather information about a machine as vibrations can potentially reveal faults a very long time before they become critical. For ball- and rolling bearings, which are present in any machinery with rotating parts, developing faults can be discovered by changes in the vibration frequencies several months (in some cases up to a year) before critical levels are reached.

Vibration analysis works by employing the Fourier transformation to the vibration data measured in units of displacement, velocity or acceleration. The transformation yields a spectrum of the different frequencies at which the machinery operates. It is in this spectrum that faults can more easily be revealed, as different fault patterns will have their own deviation in the measured frequencies. However, the analysis of vibrations is considered a whole field in itself and requires a great level of knowledge about different fault signatures. Moreover, knowledge of the different parts in the machinery, e.g., the gears, bearings, transmission system etc, also helps tremendously, as expected damage frequencies can usually be obtained through the manufacturer.

Vibration measurements can either be done every once in a while (offline), or mounted sensors could be placed on the machine for continuous, real-time monitoring (online). Both solutions can reveal faults ahead of time, but naturally, the online system yields the greatest benefits as changes can be detected as early as possible when they start to develop. This of course requires a setup of sensors and a system for constantly monitoring the vibrations. A good discussion of vibration analysis can be found in [5].

Somewhat related is the field of acoustics, which is based on the propagation of sound waves stemming from the different parts of the machinery. This is related to vibrations as both are essentially microscopic shockwaves travelling through a medium. Acoustic sensors in the form of small microphones pick up on the sound waves from the machine and analysis can then be performed to determine if there is a deviation in the sound patterns. This approach captures the intuition of human machine engineers since unusual noise from the machine is often how a potential fault is discovered upon human inspection of the machine.

However, at the point in time that the noise is noticeable to human ears, the fault will usually have developed for a long time and could possibly have been discovered much earlier.

Slightly related to both vibrations and acoustics is the field of electrical signature analysis (ESA), which uses electrical sensors mounted in the electrical cabinet of the (rotating) machine. These sensors constantly monitor the voltage and current going to into the machine. Small deviations in the measured electrical signatures can then reveal harmful fault patterns in the machinery with some precision. An example of this for PdM purposes is given in [17].

Other forms of equipment monitoring approaches exist, such as image analysis, oil analysis and thermography. The first is the process of algorithmically analysing images of some equipment that is to be monitored. An immediate example could be the inside of a boiler, a furnace or a silo, where human operators cannot safely inspect the inside for signs of degradation or piling up of residue. The last is related in that it uses infrared imaging to reveal leaks and other relevant fault types that could potentially be invisible to the naked eye. Oil analysis is also a somewhat situational method, as it can only be applied for machine parts that need lubrication. However, the chemical properties of the oil such as viscosity, contamination, oxidation, etc., can indicate certain types of wear in the part of the machine where the lubrication is needed. The field of oil analysis is also sometimes known as *tribology*.

A discussion of thermography and lubrication oil analysis can be found in [18]. As stated, these types of monitoring depend heavily on the equipment and may not make sense to use at all for some applications.

6.3 Inferring future fault patterns

Some requirements for estimating RUL models were alluded to earlier. We will not delve deeply into this, but briefly mention some models that are useful for this.

In the pursuit of a good PdM system, there are two common situations (and whatever is between these two). One might have only a single piece of equipment that is desired to have a PdM system in place, or one might have many identical. The definition of *many* will be entirely dependent on the situation and the history of past fault examples. For the purpose of this section, we thus let this remain vague for now and simply assume that a failure history exists that is useful enough for establishing a model. The failure history then serves to obtain the time-to-failure of the equipment in question, which is necessary for the models to work.

6.3.1 Poisson models

The first type of model for quantifying the incidence rate of faults over time is a Poisson model. This model assumes that faults happen over time and that the time between each of them is identically distributed according to some exponential distribution [2]. This is a very simple model and should be used with some caution, but it may be a decent building block or a good guide for more advanced models.

6.3.2 Survival models

The goal of survival analysis is to make statistical inference about the time until some event occurs. The perhaps most well-known model for this is the Kaplan-Meier (KM) estimator, which is a classical, non-parametric way to estimate a survival function given data with relevant lifetimes. This model has enjoyed extensive use for estimating lifetimes of patients in medical research, but many applications have also been focused on estimating the RUL of machine components for industrial applications.

The KM model is far from the only model useful for survival analysis. There is a plethora of models, and many machine learning algorithms have also been adapted for this task, which requires the algorithm to be able to work with censored data.

A good overview of many different methods, performance metrics, survival functions and survey of the field of survival analysis in general is given in [1].

7 CONCLUSION

In this paper, we have given an overview of some practical considerations related to running an industrial PdM system. We have covered the different maintenance philosophies as well as essentials of what makes a PdM system work as well as the requirements for data and historical maintenance records. We concluded that time-based maintenance is and has been the most commonly applied form of maintenance, but that it is not optimal in that potentially unnecessary maintenance or replacement is likely to result from it. Condition-based maintenance tries to solve this problem by monitoring equipment condition and making forecasts as to when maintenance should optimally be carried out.

We found that many challenges are associated with monitoring equipment, especially in the requirements for data and suitable mathematical models. Furthermore, domain knowledge about the monitored equipment is necessary to optimally monitor the condition, as we found that fault specific monitoring models are likely to yield the most focused and early warnings of developing faults.

The need for an automated data and model flow was emphasised and thus, a recommendation for an automated process of data flow and model output was given. Responsibilities associated with running the PdM system and acting on the outputs from it were also discussed. In relation to this, the interpretability of the PdM system and its mathematical models was stressed, as this helps the maintenance personnel understand the system better, which could help in increasing their trust in it.

Lastly, when modelling the condition of the equipment, good feature engineering can reduce the need for very complex mathematical models and allow better detection ability of certain faults. This combined with the need for an interpretable system led us to discuss and stress simplicity of the mathematical models and only increasing complexity if necessary.

REFERENCES

[1] Ping Wang, Yan Li, and Chandan k. Reddy. 2019. “Machine Learning for Survival Analysis: A Survey”. *ACM Comput. Surv. 51, 6, Article 110 (February 2019), 36 pages*.

[2] Mark A. Pinsky, Samuel Karlin. 2011. “An introduction to stochastic modelling” (4th ed.) *edition, Academic Press*.

[3] Jesper Fink Andersen. 2021. Maintenance optimization for multi-component systems using Markov decision processes, *PhD thesis, Technical University of Denmark*.

[4] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. 2008. "Isolation forest." *2008 eighth ieee international conference on data mining*. IEEE.

[5] Robert Bond Randall. 2021. “Vibration-based condition monitoring: Industrial, automotive and aerospace applications” (2nd ed.). *Wiley-Blackwell, Hoboken, NJ.*

[6] Max Kuhn, Kjell Johnson. 2019. “Applied predictive modelling”. *Springer, New York, NY.*

[7] Alberto Ferrer. 2014. “Latent Structures-Based Multivariate Statistical Process Control: A Paradigm Shift”, *Quality Engineering, 26:1, pp. 72-91*

[8] Bernhard Schölkopf, Robert Williamson, Alex Smola, John Shawe-Taylor, and John Platt. 1999. “Support vector method for novelty detection”. *In Proceedings of the 12th International Conference on Neural Information Processing Systems (NIPS'99). MIT Press, Cambridge, MA, USA, 582–588.*

[9] Erik Vanhatalo, Murat Kulahci, Bjarne Bergquist. 2017. “On the structure of dynamic principal component analysis used in statistical process monitoring”. *In: Chemometrics and intelligent laboratory systems, Vol. 167. pp. 1-11.*

[10] Raghavendra Chalapathy, Aditya Krishna Menon, and Sanjay Chawla. 2018. "Anomaly detection using one-class neural networks." *arXiv preprint arXiv:1802.06360.*

[11] Douglas C. Montgomery. 2013. “Introduction to statistical quality control”*. Hoboken, NJ: Wiley.*

[12] Hiromichi Kawano, Tetsuo Hattori, Ken Nishimatsu. 2008. “Structural Change Point Detection Method of Time Series Using Sequential Probability Ratio Test”*.*

IEEE Transactions on Electronics, Information and Systems.

[13] Søren Bisgaard, Murat Kulahci. 2011. “Time Series Analysis and Forecasting by Example”, *Wiley, 2011. 392 p. (Wiley Series in Probability and Statistics).*

[14] Bernhard Schölkopf, Alex Smola, Klaus-Robert Müller. 1998. "Nonlinear Component Analysis as a Kernel Eigenvalue Problem"*. Neural Computation. 10 (5): 1299–1319*

[15] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, 1996. ”A density-based algorithm for discovering clusters in large spatial databases with noise”. *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96). AAAI Press. pp. 226–231*

[16] Adele Cutler and Leo Breiman. 1994. “Archetypal analysis”. *Technometrics, 36(4):338–347, November 1994*

[17] Camila Paes Salomon, et al. 2019. "A study of fault diagnosis based on electrical signature analysis for synchronous generators predictive maintenance in bulk electric systems." *Energies 12.8 (2019): 1506*.

[18] R. Keith Mobley. 2002. “An introduction to predictive maintenance”, 2nd edition*. Elsevier*.

[19] Rolf Isermann. 2011. “Fault-diagnosis applications: model-based condition monitoring: actuators, drives, machinery, plants, sensors, and fault-tolerant systems”. *Springer Science & Business Media.*

[20] George E. P. Box, et al. 2015. “Time Series Analysis: Forecasting and Control”, 5th edition. *Wiley.*