Fault Diagnosis of Wind Turbine Drive Train Faults based on Dynamical Clustering.

Antoine Chammas$^1$ and Eric Duviella$^1$ and Stéphane Lecoeuche$^1$

Abstract—In this paper, a fault diagnosis architecture based on a dynamical clustering algorithm is developed to detect and isolate faults in wind turbines. The challenge is to deal with different kinds of faults. Constraints on the time of detection are also added in the sense that a fault must be detected as soon as possible. Also, limited historical data corresponding only to normal operating modes are available. Our methodology is based on a data-driven model and is therefore not dependent on the physical models in the wind turbine. It consists of extracting, from sensor measurements, features that are fed into a dynamical clustering algorithm. The latter learns process behaviors characterized by clusters with the ability to update, recursively, the parameters of these clusters. These parameters are used to create detection signals and health indicators used for diagnosis.

I. INTRODUCTION

Despite that wind energy is becoming more and more demanded in modern societies. It has been stated that one of the major reasons for its high prices is the functioning downtime [1], which is mainly caused by the failure of some components in a wind turbine. Therefore, it is more and more crucial to increase investment in more reliable Fault Detection and Isolation (FDI) diagnosis schemes.

This subject has attracted a lot of interest recently. A lot of papers contributed with diverse methodologies to the FDI problem in wind turbines. In [2], a survey on fault detection and condition monitoring for wind turbines is presented. In [3], an overview of fault detection in wind turbines is given. The authors studied and analyzed the faults and their root-causes. Methodologies for fault detection cover observer based schemes like [4], [5], parity equations and set membership approaches [6], [7]. In [8], the authors studied the blade root moment sensor fault detection and isolation issue. They considered the four scenarios of sensor faults: additive faults, multiplicative faults, output stuck and sensor drift. These methodologies are physical-based and thus have one major disadvantage which is the knowledge of the physical model of the process. Alternatively, data-driven approaches do not require physical modeling but make direct use of data measured on a process. This makes them a more suitable choice for a complex system like wind turbines. In [9], [10], [11], [12] data driven approaches have been used. For instance, a SVM based scheme is developed in [12].

The authors extracted features from sensor measurements without any use of a physical model.

In this paper, a data-driven architecture for FDI for wind turbines is proposed. It is tested on a benchmark that was originally developed for a worldwide FDI competition [13]. The results we obtained are compared to two highly ranked papers from this competition. The considered faults in these papers (and our paper) are abrupt faults. The tool that we use is a dynamical clustering algorithm. This tool is used to model operating modes of the wind turbine in a feature space by clusters [14]. The parameters of these clusters allow us to compute indicators on the health of the process, which then are processed to produce FDI results. This work concerns the drive train of a wind turbine. We finally note that in our previous work [15], [16], slowly developing faults or drifts were handled. In this work, the methodology is adapted and developed for abrupt faults, and successfully applied.

The rest of the paper is as follows. In section II, a general description of the drive train and the fault scenarios are shown. In section III, a detailed description of the methodology is given. In section IV, the results on the wind turbine benchmark are presented and in section V, the paper is concluded with some perspectives.

II. DESCRIPTION OF WIND TURBINE AND FAULT SCENARIOS

The benchmark, presented in [13], simulates a realistic generic three blades horizontal variable speed wind turbine with a full converter coupling.

The drive train is a mechanical part that relates the blades (or commonly known as the rotor) to the generator. This system can be modeled by two masses connecting over a shaft with finite torsion stiffness, subject to torsion damping and imperfect transmission efficiency. A gearbox is used to increase the angular speed of the rotor (low speed shaft) into the generator (high speed shaft). The model is represented in Fig. (1) where the main input-output variables are indicated.

![Fig. 1. Drive train box model.](image)

The instrumentation on the drive train is as follows:
1) \( \omega_r \) and \( \omega_g \) are measured each with two sensors, namely \( \omega_{m1}^r \), \( \omega_{m2}^r \), \( \omega_{m1}^g \) and \( \omega_{m2}^g \).

2) \( \tau_g \) is measured by only one sensor, namely \( \tau_g^m \). However, \( \tau_r \) is not measured.

There are also measurements outside the drive train that are interesting to us. For instance, the power extracted from the wind turbine, denoted \( P_g \) is also measured by only one sensor, namely \( P_m^g \). In addition, the wind speed \( v_w \) is measured by only one sensor also \( v_{m} \).

We assume that sensors are corrupted with white Gaussian noise. In general, let \( S_{Ph}^m \) be a sensor measurement of a physical value \( P_h \), then we have:

\[
S_{Ph}^m = S_{Ph} + \epsilon_S^m,
\]

where \( S_{Ph} \) is the real value and \( \epsilon_S \) is white Gaussian noise that depends on the sensor.

In [13], several Fault Scenarios (FaSc) were considered covering different parts of wind turbine. The faults were categorized as sensor faults, actuator faults and system faults. Sensor faults are of two possible types: fixed value (FV) or gain factor (GF). In this work, since we are interested in the drive train, we retain from these fault scenarios only three that are related to it. These are FaSc 4, 5 and 9 in [13]. FaSc 4 and 5 are sensor faults whereas FaSc 9 is a system fault:

- FaSc 4: FV on the sensor \( \omega_r^m \), i.e. \( \omega_r^m = C_o \), where \( C_o \) is a constant.
- FaSc 5: GF on the sensors \( \omega_r^m \) and \( \omega_g^m \). \( \omega_r^m = 1.1 \omega_r + \epsilon_{\omega_r^m} \) and \( \omega_g^m = 0.9 \omega_g + \epsilon_{\omega_g^m} \).
- FaSc 9: change in the friction parameter \( \nu_d \) from its nominal value to a faulty value, denoted \( \nu_d^f \).

### III. Methodology

The architecture for fault detection and diagnosis is depicted in Fig. (2). It consists of five phases. The phase A consists in detecting quickly FV type sensor faults. The reason is that these faults are the easiest to detect and can be done directly once sensor measurements are available. Secondly, features are extracted from sensor measurements (phase B). These features are essential to build feature spaces on which the operating modes of the wind turbine are modeled by clusters. The modeling tool is an algorithm called AuDyC (Auto-Adaptive Dynamical Clustering). Two feature spaces are constructed via AuDyC. One that is used to raise an alarm in case of fault and to decide whether it is a system fault or a sensor fault, and it will be called detection feature space. Another one that investigates more in finding the particular faulty component or sensor, and it will be called isolation feature space. The reason for this choice is performance and robustness. The clustering algorithm will perform better (faster convergence, easier to handle) when the features needed for the detection and separation of the type of faults are separated from the features needed for isolation. The outputs of phase C are dynamically updated cluster parameters. These are used in phase D to compute health indicators. These are the basis of the FDI decision module in phase E. In the rest of this section, these steps are described in more detail.

This methodology is general but must be adapted to each application. This is done by correctly choosing the features.

#### A. Phase A: initial processing on sensor measurements

The purpose of phase A is to quickly detect sensor faults of type FV. This is done by differencing the sensor measurements. The reason is that, in case of FV sensor fault, the differentiation yields 0. In fact, let \( \Delta \) be the differencing operator, then we have:

\[
\Delta(S_{Ph}^m(t)) = S_{Ph}^m(t) - S_{Ph}^m(t - 1).
\]

In case of FV sensor fault, the fault can thus be detected and isolated two time steps after its appearance. This is applied to all the considered sensors.

#### B. Phase B: feature estimation

Features are signatures extracted from sensor measurements. They must be carefully chosen for FDI purposes. They must be sensitive to faults in such a way that they will also allow isolation of faults. In this paper, 7 features are estimated from sensor measurements. The estimated features are:

\[
R_{\omega_r} = \omega_r^m - \omega_r^m,
\]

\[
R_{\omega_g} = \omega_g^m - \omega_g^m,
\]

\[
R_{\nu_d} = 10^{-10}(v_{m}^d)^6(\nu_d^d - 1/2(\omega_g^m + \omega_g^m)),
\]

\[
t_{pw1} = \frac{\tau_{g}^m}{P_m^g} \omega_r^m,
\]

\[
t_{pw2} = \frac{\tau_{g}^m}{P_m^g} \omega_r^m,
\]

\[
t_{pwg1} = \frac{\tau_{g}^m}{P_m^g} \omega_g^m,
\]

\[
t_{pwg2} = \frac{\tau_{g}^m}{P_m^g} \omega_g^m.
\]

The feature \( R_{\omega_r} \) is the difference between two redundant sensor measurements. It allows the detection of FV faults.
on the sensors \( \omega_{m1}^r \) and \( \omega_{m2}^r \). However, alone, this feature is not enough to decide which the faulty sensor is. Thus, \( tpwr1 \) and \( tpwr2 \) are used to isolate between the sensors. The same logic goes for \( R_{owg} \) and the features \( tpwg1 \) and \( tpwg2 \). It is worthy noting that it is essential to consider the four residuals Eq. (6-9) to be able to isolate between the four possible sensor faults. Only the feature \( R_{owg} \) is inspired from [12]. The authors of [12] argued that including the wind speed measurement in \( R_{owg} \) helped building more robust feature in low wind speed situations. In fact, let \( P_r \) be the nominal reference power of the wind turbine (constant) and \( r^g_f \) be the reference generator torque given by the controller, then \( \omega_g^r = \frac{P_r}{r^g_f} \) is called the desired generator speed. For this reason, \( R_{owg} \) is very sensitive to the system fault described in section II. It is much less sensitive to faults defined in FaSc 5. For this reason, it is retained. The Table I shows which features are the most sensitive to which fault. Since \( R_{owg} \), \( R_{owg} \) and \( R_{owg} \) are enough to detect a sensor fault or a system fault, they are used to span a detection feature space, denoted \( E_D \) (dimension 3). The four features \( tpwr1 \), \( tpwr2 \), \( tpwg1 \) and \( tpwg2 \) are used to isolate the faulty sensor, thus they span an isolation feature space, denoted \( E_I \) (dimension 4).

<table>
<thead>
<tr>
<th>Fault</th>
<th>Sensitive features</th>
</tr>
</thead>
<tbody>
<tr>
<td>PV on ( \omega_{m1}^r )</td>
<td>( R_{owg} ), ( tpwr1 )</td>
</tr>
<tr>
<td>GF on ( \omega_{m2}^r )</td>
<td>( R_{owg} ), ( tpwr2 )</td>
</tr>
<tr>
<td>GF on ( \omega_{m2}^r )</td>
<td>( R_{owg} ), ( tpwg2 )</td>
</tr>
<tr>
<td>System fault</td>
<td>( R_{owg} )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Old pattern</th>
<th>Newest arrived pattern</th>
<th>Oldest pattern forgotten – Updating of the parameters cluster</th>
</tr>
</thead>
<tbody>
<tr>
<td>At time 1</td>
<td>At time t+1</td>
<td>Dynamical adaptation</td>
</tr>
</tbody>
</table>

C. Phase C: dynamical clustering

1) AuDyC: dynamical clustering algorithm: AuDyC uses a technique that is inspired from the Gaussian mixture model [14], [17]. Let \( \mathbb{R}^d \) be a \( d \)-dimensional feature space. Each feature vector \( x \in \mathbb{R}^d \) is called a pattern. The patterns are used to model Gaussian prototypes \( P_i \) characterized by a center \( \mu_{P_i} \in \mathbb{R}^{d \times 1} \) and a covariance matrix \( \Sigma_{P_i} \in \mathbb{R}^{d \times d} \). Each gaussian prototype characterizes a cluster. A minimum number of \( N_{win} \) patterns are necessary to define one prototype, where \( N_{win} \) is a user-defined threshold. A cluster models an operating mode and groups patterns that are similar one to each other. The similarity criterion that is used is the Gaussian membership degree. AuDyC is dynamical by nature in the sense that it continuously updates the parameters of the clusters as new patterns arrive (see Fig. (3)).

This makes it a suitable choice to model streams of patterns since it reflects always the final distribution of patterns in the features space. Faults will affect directly this distribution and this will be seen on the continuously updated parameters. The rules for recursive adaptation of parameters (as in Fig. (3)) are:

\[
\begin{align*}
\mu_{P_i}(t) &= \mu_{P_i}(t-1) + f(\mu_{P_i}(t-1), X^{new}, X^{old}, N_{win}), \\
\Sigma_{P_i}(t) &= \Sigma_{P_i}(t-1) + g(\Sigma_{P_i}(t-1), \mu_{P_i}(t-1), X^{new}, X^{old}, N_{win}),
\end{align*}
\]

where \( X^{new} \) and \( X^{old} \) are the newest arrived pattern and the oldest pattern in the time window \( N_{win} \) respectively. For more details on the functionalities (there are other functionalities then adaptation like merging clusters, splitting clusters, ...), the rules of recursive adaptation and the similarity criteria in AuDyC, please refer to [17], [14].

Initial offline modeling allows the construction of initial clusters that characterize knowledge from historical data. The latter are usually sensor data that are saved. Feature estimation, as in section II-B, applied on these saved sensor data, are used to initialize the two feature spaces that later on will be dynamically updated. Knowledge of failure modes given from (labeled) historical data can help building a classification scheme for fault diagnosis. However, in reality, these data are hard to obtain.

In this work, we suppose we only have data corresponding to normal operating modes. We have 7 features that are extracted from historical sensor data, as in section II-B. We start the training process by applying AuDyC. Once finished, one cluster corresponding to normal operating mode is retained in each of the feature spaces \( E_D \) and \( E_I \) (since only data corresponding to normal operating modes are considered available). In \( E_D \) we denote this cluster by \( C_D^{E_D} = (\mu_D^{E_D}, \Sigma_D^{E_D}) \). In \( E_I \), we denote it by \( C_I^{E_I} = (\mu_I^{E_I}, \Sigma_I^{E_I}) \).

In online fashion, the parameters of \( C_D^{E_D} \) and \( C_I^{E_I} \) are dynamically updated by AuDyC s each new pattern arrives. This yields changes in the cluster parameters which continuously reflect the distribution of the newest arriving patterns. We denote by \( C_D^{E_D}(t) = (\mu_D^{E_D}(t), \Sigma_D^{E_D}(t)) \) and \( C_I^{E_I}(t) = (\mu_I^{E_I}(t), \Sigma_I^{E_I}(t)) \) the evolving clusters in \( E_D \) and \( E_I \) respectively. We have \( C_D^{E_D}(t = 0) = C_D^{E_D} \) and \( C_I^{E_I}(t = 0) = C_I^{E_I} \).

D. Phase D: health indicators

The proposed methodology makes use of cluster parameters which are dynamically updated at each time stamp \( t \).
Health indicators are signals extracted from these parameters and FDI decisions will be made based on their values. We define \( \xi_{E_D}(t) \) as:

\[
\xi_{E_D}(t) = d_{KL}(C_N^{E_D}, C_e^{E_D}(t)),
\]

where \( d_{KL} \) is the symmetrised Kullback-Leibler divergence. Its mathematical expression, between the two Gaussian multivariate distributions \( C_N^{E_D} = (\mu_N^{E_D}, \Sigma_N^{E_D}) \) and \( C_e^{E_D}(t) = (\mu_e^{E_D}(t), \Sigma_e^{E_D}(t)) \) is given by:

\[
d_{KL}(C_N^{E_D}, C_e^{E_D}(t)) = d_{KL}(C_e^{E_D}(t), C_N^{E_D}) = \frac{1}{2} \times \left( \text{tr}(\Sigma_N^{E_D}) - \text{tr}(\Sigma_e^{E_D}(t)) + (\Sigma_e^{E_D}(t))^{-1} \times \Sigma_N^{E_D} \right) + (\mu_N^{E_D} - \mu_e^{E_D}(t))^T (\Sigma_N^{E_D})^{-1} + (\Sigma_e^{E_D}(t))^{-1} \times (\mu_N^{E_D} - \mu_e^{E_D}(t)) - d.
\]

\( \xi_{E_D}(t) \) is called the Severity Indicator. Another two indicators, called Direction Indicators (DI) are also calculated. They are denoted \( \Xi_{E_D}(t) \) and \( \Xi_{E_I}(t) \) and are given by:

\[
\Xi_{E_D}(t) = (\mu_e^{E_D}(t) - \mu_N^{E_D})^T, \quad \Xi_{E_I}(t) = (\mu_e^{E_I}(t) - \mu_N^{E_I})^T.
\]

We have \( \Xi_{E_D}(t) \in \mathbb{R}^3 \) and \( \Xi_{E_I}(t) \in \mathbb{R}^4 \). Each component of \( \Xi_{E_D}(t) \) correspond to one of the features that were used to span \( E_D \). The same goes for the components of \( \Xi_{E_I} \) with \( E_I \). The meaning of these indicators and the reason why they were calculated this way is explained in the next subsection.

E. Phase E: decision scheme for FDI

This step achieves the FDI decisions. Reaching this phase means that FV sensor faults did not occur. The possible faults are GF on sensors \( \omega_{r,m}^{m_1}, R_{r,m_2}^{m_2}, \omega_{g,m}^{m_1}, \omega_{g,m}^{m_2} \) (FaSc 5) or the system fault (FaSc 9). The logic behind the decision algorithm is based on the sensitivity of the features to the faults (table I). It is depicted graphically in Fig.4. If a GF sensor fault has occurred, the features \( R_{\omega_r} \) and \( R_{\omega_g} \) are excited. If the system fault occurred, \( R_{\omega_{cg}} \) is affected. In both cases, this will affect \( \xi_{E_D}(t) \) and an alarm should be launched. The alarm launching mechanism is based on the value of \( \xi_{E_D}(t) \) when it exceeds a certain threshold \( \eta \):

\[
\xi_{E_D}(t) \begin{cases} < \eta, \text{ No alarm} \\ \geq \eta, \text{ Launch alarm} \end{cases}
\]

In the case of positive alarm launch, the next step will be to decide whether it is a GF sensor fault or a system fault. For this reason the signal \( \Xi_{E_D}(t) \) is used. Depending on the excited components of this vector, we can know in which faulty case we are. For instance, if the value of the components corresponding to \( R_{\omega_r} \) or \( R_{\omega_g} \) exceeds a threshold, then we can conclude that it is a GF sensor fault. If the value of the component corresponding to \( R_{\omega_{cg}} \) exceeds a threshold, then it is the system fault. If a GF sensor fault is detected, then the signal \( \Xi_{E_I}(t) \) is used to isolate the faulty sensor. For instance, in the case of a GF on \( \omega_{g,m}^{m_2} \), only the component corresponding to \( tpwg2 \) will exceed a threshold. Finally we note that, in this paper, the thresholds are experimentally chosen in order to remove false alarms. The thresholds for each component of \( \Xi_{E_D}(t) \) and \( \Xi_{E_I}(t) \) can change from one to another. It can be argued that these thresholds could be found by other statistical means, or by expert knowledge, or by availability of more data, particularly failure data.

![Decision scheme for FDI](image_url)

**IV. RESULTS**

The developed methodology for FDI is tested on the benchmark [13]. The sampling period is \( T_s = 0.01 \text{s} \). The results are compared with two highly ranked papers [12], [4] (ranks 1 and 3 respectively), and summarized in Table II and Table III.

<table>
<thead>
<tr>
<th>TABLE II</th>
</tr>
</thead>
<tbody>
<tr>
<td>Summary of results</td>
</tr>
<tr>
<td>Fault time span</td>
</tr>
<tr>
<td>FDI requirements</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>FDI time results</th>
<th>[12]</th>
<th>[4]</th>
<th>this paper</th>
</tr>
</thead>
<tbody>
<tr>
<td>21s</td>
<td>12.47s</td>
<td>21s</td>
<td></td>
</tr>
</tbody>
</table>

It can be considered from the results that our methodology gives promising results. In most of the cases (except the GF on \( \omega_{g,m}^{m_2} \), our algorithm seems to give better results then [4], in which a model-based methodology is defined. Compared to [12], our methodology has also given comparable results. For FaSc 4, where the sensor \( \omega_{g,m}^{m_1} \) is affected by a FV fault, the FDI is done directly by phase A. Fig.(5) shows the behavior of the signal \( \Delta(\omega_{g,m}^{m_1}) \) versus time, enlarged at the time of occurrence of the FaSc 4, which is 1200s.

FaSc 5 contains two GF sensor faults affecting \( \omega_{g,m}^{m_2} \) and \( \omega_{g,m}^{m_2} \). The papers to which our approach was compared successfully diagnosed these two faults. In our approach, the FDI of these sensor faults is done following the phases from B to E. The first studied signal is \( \xi_{E_D}(t) \) that allows an initial alarm launching. Fig. (6) shows this signal at the time of FaSc 5. Fig. (7) shows the same signal but enlarged over the time occurrence of the FaSc 5, which is 1700s. Then,
the signal $\Xi_{ED}$ is used to firstly decide if it is the system fault or a sensor fault. In Fig. (8), the three components of $\Xi_{ED}$ are shown. It is obvious that, at the time of FaSc 5, the excited components correspond to $R_{\omega_r}$ and $R_{s}$ (with a time difference). This suggests that at least one of the two sensors for each angular speed (the generator and the rotor) is faulty.

At this point, the vector $\Xi_{E_{I_{i}}}(t)$ will be used to isolate the faulty sensor. The components of the vector $\Xi_{E_{I_{i}}}$ are depicted in Fig. (9). It is clearly shown that two components (corresponding to the $tpwg2$ and $tpwr2$) are excited suggesting that the faulty sensors are $\omega_{g}^{m2}$ and $\omega_{s}^{m2}$. Fig. (10) show the component corresponding to $tpwr2$ and the thresholds that were chosen for it. We should remind that the choice of the thresholds is done to remove false alarms.

FaSc 9 is the only scenario for the system fault. In the same manner, the signal $\Xi_{ED}(t)$ detects an anomaly at the
Fig. 10. The component of the direction indicator $\Xi_{E_1}(t)$ corresponding to $tpwr2$. 

beginning. However, in this case, the vector $\Xi_{E_1}(t)$ isolates it immediately (because it is the only considered system fault). Fig. (11) shows the SI $\xi_{E_D}(t)$ at the time it was detected. We remind that this fault begins at 3000s. Fig. (12) shows the corresponding component of $\Xi_{E_D}(t)$ and the threshold chosen for it.

Fig. 11. Severity indicator $\xi_{E_D}(t)$ for FaSc 9.

Fig. 12. The component of the direction indicator $\Xi_{E_D}(t)$ corresponding to $R_{\omega_{gc}}$. 

V. CONCLUSION AND PERSPECTIVES

In this paper, we propose an architecture for fault detection and isolation for wind turbines. It was tested on a benchmark that was used in the context of a worldwide competition. Our methodology showed good results compared to the best ranked papers from this competition. The data available for our algorithm to work does not need to include failure data, but needs a good adjustment of thresholds.

Since this approach can handle drifts also, an interesting perspective is in the direction of prognosis. The architecture developed here could be part of a larger architecture of Condition-Based Maintenance (CBM) in which prognosis plays an important role.

REFERENCES


