



Delft University of Technology

## Simulating run-to-failure SCADA time series to enhance wind turbine fault detection and prognosis

Eftekhari Milani, A.; Zappalá, D.; Castellani, Francesco; Watson, S.J.

**DOI**

[10.5194/wes-10-2563-2025](https://doi.org/10.5194/wes-10-2563-2025)

**Publication date**

2025

**Document Version**

Final published version

**Published in**

Wind Energy Science

**Citation (APA)**

Eftekhari Milani, A., Zappalá, D., Castellani, F., & Watson, S. J. (2025). Simulating run-to-failure SCADA time series to enhance wind turbine fault detection and prognosis. *Wind Energy Science*, 10(11), 2563–2576. <https://doi.org/10.5194/wes-10-2563-2025>

**Important note**

To cite this publication, please use the final published version (if applicable).

Please check the document version above.

**Copyright**

Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

**Takedown policy**

Please contact us and provide details if you believe this document breaches copyrights.

We will remove access to the work immediately and investigate your claim.



# Simulating run-to-failure SCADA time series to enhance wind turbine fault detection and prognosis

Ali Eftekhari Milani<sup>1</sup>, Donatella Zappalà<sup>1</sup>, Francesco Castellani<sup>2</sup>, and Simon Watson<sup>1</sup>

<sup>1</sup>TU Delft, Kluyverweg 1, Delft, 2629 HS, the Netherlands

<sup>2</sup>Department of Engineering, University of Perugia, Via Duranti, 06125, Perugia, Italy

**Correspondence:** Ali Eftekhari Milani (a.eftekharimilani@tudelft.nl)

Received: 4 April 2025 – Discussion started: 13 June 2025

Revised: 4 August 2025 – Accepted: 30 September 2025 – Published: 12 November 2025

**Abstract.** Wind turbine supervisory control and data acquisition (SCADA) datasets available for research usually contain a limited number of failure events. This limitation hinders the successful application of deep learning (DL) methods for fault detection and prognosis, as they require large datasets for robust training and generalisation. This work proposes a method using conditional generative adversarial networks (cGANs) to generate synthetic SCADA time series that replicate wind turbine behaviour under controllable operational, environmental, and degradation conditions. Given a set of SCADA time series representing these conditions, the cGAN generates temperature and pressure time series simulating gearbox operation. Results show that augmenting the training set of an artificial neural network (ANN) fault detection model with synthetic time series reduces false positives in the detected gearbox faults by 84 % on average, enabling the model to blindly detect a fault in a test wind turbine without prior knowledge of the event. Furthermore, training a convolutional autoencoder-based unsupervised health indicator (HI) model with both real and synthetic SCADA time series leads to an HI that more accurately captures the expected degradation trend. Using this HI, the gearbox's remaining useful life (RUL) can be predicted within the defined error bounds from around 4.5 months before the detection of the fault, while the HI obtained without the synthetic data fails to produce reliable RUL estimations.

## 1 Introduction

State-of-the-art deep learning (DL) methods for wind turbine fault detection and prognosis rely on large datasets for robust training and generalisation. However, component failures in wind turbines are rare events (Spinato et al., 2009), and wind farm operators are often reluctant to disclose detailed information about them due to privacy concerns (Chatterjee and Dethlefs, 2021). Therefore, supervisory control and data acquisition (SCADA) datasets available for research usually include very few failure events, limiting the successful implementation of DL methods. A viable solution to this challenge is to simulate new failure events within SCADA datasets. This involves generating time series data that mimic sensor signals reflecting turbine component behaviour as degradation progresses over a specific time window leading to failure. Rather than merely replicating existing failure event sig-

nals, these synthetically generated run-to-failure time series should instead capture diverse degradation scenarios under varying operational and environmental conditions. This diversity is crucial for enhancing the training data of DL fault detection and prognosis models, improving their robustness and practicality.

Existing approaches for generating synthetic signals have mostly used physics-based and hybrid physics-data-driven models of wind turbines. They often simulate damage by methods such as inserting additional mass or reducing local structural stiffness. For example, synthetic vibrational signals generated using the OpenFAST software (Jonkman et al., 2022) have been used to validate condition monitoring methods in Tatsis et al. (2017), Tatsis et al. (2021), and Song et al. (2024). However, they are unsuitable for training DL methods to be applied to real wind turbines with different configurations. In Pujana et al. (2023), a hybrid digital twin

of a wind turbine drivetrain is developed to generate synthetic stator winding temperature signals, with the temperature increase due to a generator failure modelled as a heat exchanger. These synthetic signals are used to train a fault detection model. While useful, these methods oversimplify the actual component behaviour and cannot model gradual degradation. Therefore, they fail to generate realistic run-to-failure sequences across multiple SCADA signals, which are critical for prognostic applications.

Data-driven approaches for generating synthetic SCADA time series are rare in the literature. An artificial neural network (ANN)-based framework for generating synthetic SCADA signals, given operational, environmental, and degradation conditions, is proposed in Eftekhari Milani et al. (2024a). While the generated synthetic signals are in good agreement with the corresponding field data, this approach assumes that sensor signals at each timestamp are deterministic functions of the current conditions. This assumption overlooks the inertia and temporal dependencies present in SCADA signals. These signals, especially temperature data, often exhibit significant inertia, with measurements strongly dependent on their previous values (Mello et al., 2021). Furthermore, this approach does not address the inherent stochasticity of SCADA signals and does not demonstrate whether it can simulate new failure events.

Generative adversarial networks (GANs) (Goodfellow et al., 2020) have been proven successful in generating diverse and realistic synthetic data across many domains, including images (Shorten and Khoshgoftaar, 2019), text (Li et al., 2018), audio (Liu et al., 2022), and video (Chu et al., 2020). These models consist of two neural networks trained in a competitive framework: a generator generating realistic-looking synthetic samples and a discriminator evaluating whether the data are real or synthetic. In the wind turbine SCADA data domain, the application of GANs has been mostly limited to addressing class imbalances in fault detection tasks by augmenting faulty data instances. For example, faulty data instances are generated in Liu et al. (2019) to enhance fault detection performance. In Wang et al. (2022), a variant of GAN called the least squares GAN is used to generate synthetic data instances and improve the performance of an autoencoder-based condition monitoring framework. Similarly, in Liu et al. (2023), a GAN is used to overcome the limitation of scarce faulty data by generating synthetic faulty instances, which are then used to enhance the performance of an autoencoder-based anomaly detection method. These methods outperform more traditional oversampling approaches (Antoniou et al., 2017), such as those based on the synthetic minority oversampling technique (SMOTE) (Chawla et al., 2002), which have been extensively used to address the problem of class imbalance in SCADA datasets (Peng et al., 2020; Yang et al., 2021; Li et al., 2023; Tao et al., 2024). However, they are limited to generating individual signal instances rather than run-to-failure time series. Another limitation of these methods is that they cannot sim-

ulate entirely new failure events and are limited to oversampling faulty data instances corresponding to existing failure events in SCADA datasets (Chesterman et al., 2023). Extending these approaches to generate entire time series is challenging, as a generative model must learn not only the feature distributions but also their temporal dynamics. Furthermore, it must generate a diverse set of time series under predefined operational, environmental, and degradation conditions to be useful for wind turbine fault detection and prognosis.

This work addresses these limitations by developing a method based on a conditional GAN (cGAN) (Mirza and Osindero, 2014). Unlike the standard vanilla GAN, the proposed model allows conditioning each generated signal instance to a vector of predefined conditions, including component degradation levels and SCADA measurements related to environmental conditions and the operational states of the wind turbine. To capture temporal dynamics of the signal instances, gated recurrent unit (GRU)-based recurrent neural networks (Cho et al., 2014) are used for both the generator and the discriminator networks, enabling the model to retain a memory of condition vectors from previous timestamps. Furthermore, as suggested by Yoon et al. (2019), a supervised loss term is added to the generator's training loss function to enhance the cGAN's ability to generate realistic SCADA time series. The effectiveness of this approach for fault detection and remaining useful life (RUL) prediction is demonstrated using field SCADA data. The results show that augmenting field data with synthetic time series generated by the cGAN significantly reduces false positives caused by the scarcity of failure events in training data. This enables the model to blindly detect a fault in one of the test wind turbines without prior knowledge of the event. Furthermore, including synthetic time series enhances the performance of a health indicator (HI) construction model. The resulting HIs better capture the degradation trend than those generated without the synthetic data, leading to more accurate RUL predictions. The rest of this paper is organised as follows. Section 2 describes the method developed for synthetic SCADA time series generation. Section 3 introduces the SCADA dataset used, data preprocessing, selected signals, and the healthy and faulty wind turbines. Sections 4 and 5 present the application of the proposed method in fault detection and RUL prediction, respectively. Finally, conclusions are drawn, and future work is discussed in Sect. 6.

## 2 Generation of synthetic SCADA signals

A set of SCADA signals representing the operation of a wind turbine component, such as gearbox temperature and pressure signals, is a discrete multivariate time series  $\mathbf{S} = \{s_{i,t}\}$  with distribution  $p_s$ , where  $t = 1, \dots, T$ ,  $i = 1, \dots, N_s$ ,  $N_s$  is the number of component signals,  $t$  is the time instance, and  $T$  is the length of the dataset. Similarly,  $\mathbf{O} = \{o_{i,t}\}$ , where  $i = 1, \dots, N_o$ , and  $\mathbf{E} = \{e_{i,t}\}$ , where  $i = 1, \dots, N_e$ , are sets of

SCADA signals representing operational and environmental conditions over the same time interval  $t = 1, \dots, T$ , such as rotor speed and ambient temperature, with  $N_o$  and  $N_e$  being the number of signals representing these conditions, respectively.  $D$  is an HI representing the degradation of the component at each point in time, which can be extracted from the SCADA signals. The set of  $\{\mathbf{O}, \mathbf{E}, D\}$  is called the condition time series  $\mathbf{C} = \{c_{i,t}\}$ , where  $i = 1, \dots, N_c$ , and  $N_c = N_o + N_e + 1$  is the number of condition signals.

It is assumed that  $\mathbf{S}$  is a function of  $\mathbf{C}$  (Eftekhari Milani et al., 2024a), with some stochasticity inherent in the SCADA signals. At each instance  $t$ ,  $s_t$  is a function of not only  $c_t$  but also all the previous instances due to the inertia inherent in the SCADA signals. However, in practice, this dependence becomes negligible beyond a certain time window. This relationship can be expressed through a stochastic generative function  $\mathcal{F}$ :

$$s_t = \mathcal{F}(\mathbf{c}_{1:t}, \mathbf{z}_t), \quad (1)$$

where  $\mathbf{z}_t$  is a vector of random noise with distribution  $p_z$ . The objective of this work is to model  $\mathcal{F}$  through a GAN-based framework and use it to generate a set of synthetic SCADA signals  $\mathbf{S}^s$  given any  $\mathbf{C}$ .

## 2.1 HI construction

As mentioned in the previous section,  $D$  is an HI extracted from the SCADA dataset and, together with  $\mathbf{O}$  and  $\mathbf{E}$ , forms the condition time series  $\mathbf{C}$ . In this work,  $D$  is obtained using the unsupervised method developed in Eftekhari Milani et al. (2024b), where it is demonstrated that it can construct HIs that track the true component degradation trend more accurately than other methods proposed in the literature. This approach adopts a convolutional autoencoder (CAE), which is trained using a hybrid of particle swarm optimisation (PSO) and backpropagation to simultaneously maximise the HI monotonicity built in the middle layer and minimise the reconstruction error. Due to the generally irreversible nature of component degradation, an HI is expected to demonstrate a monotonic trend, and monotonicity has been widely used as one of the main criteria to build HIs (She and Jia, 2019; Yang et al., 2022). Therefore, maximising monotonicity leads to an HI that better represents the component degradation, leading to a more accurate RUL prediction. The training fitness function to be maximised is

$$f = f_M - f_R - f_0 - f_1, \quad (2)$$

where  $f_M$  is the monotonicity of the HI built in the middle layer of the CAE, measured using the Mann–Kendall (MK) metric (Pohlert, 2015);  $f_R$  is the CAE reconstruction loss, i.e., the mean squared error (MSE) of the difference between the CAE input and output;  $f_0 = |\text{HI}(0)|$  is the absolute HI value at its initial timestamp; and  $f_1 = |1 - \text{HI}(\text{end})|$ , where  $\text{HI}(\text{end})$  is the HI value at its final timestamp. Maximising  $f$

corresponds to minimising both  $f_0$  and  $f_1$  and training the CAE to associate the healthy state at the initial timestamp with an HI value of 0 and the failed state at the final timestamp with an HI value of 1. In this work,  $f_0$  is removed from  $f$  because the component is not necessarily in a pristine state at the initial timestamp of a run-to-failure SCADA time series.

SCADA measurements are characterised by high noise levels and varying operational and environmental conditions. These factors make it more challenging for the CAE to effectively reconstruct and denoise the signals compared to more controlled vibration signals obtained from bearing test beds used in Eftekhari Milani et al. (2024b). Therefore, in this work, an additional term  $f_{SD}$  is considered, which measures the average weekly rolling standard deviation of the HI, and minimising this term leads to a less noisy HI. An equal weight of 1 for the four terms  $f_M$ ,  $f_R$ ,  $f_1$ , and  $f_{SD}$  results in slow convergence during the training process due to the slow minimisation of the reconstruction error, and the obtained training HI tends to be noisy. For this reason, the weights of the  $f_R$  and  $f_{SD}$  terms are set to 3 using trial and error to balance the four terms and resolve these issues.

The CAE training fitness function thus used in this work is

$$f = f_M - 3f_R - f_1 - 3f_{SD}. \quad (3)$$

The CAE architecture and the training algorithm hyperparameters are set according to those proposed in Eftekhari Milani et al. (2024b).

Since wind turbines operate under highly variable operational and environmental conditions, the constructed HIs usually exhibit local variations, which can reduce the RUL prediction accuracy. To mitigate these variations, a post-processing algorithm is developed, leveraging the usual irreversible nature of component degradation. As shown in Algorithm 1, a curve is fitted to the HI using the non-parametric locally weighted scatterplot smoothing (LOESS) regression approach (Cleveland and Devlin, 1988). This curve is then subtracted from the HI, and subsequently, the cumulative maximum of the curve is re-added to the HI.

---

### Algorithm 1 HI post-processing algorithm.

---

1. Fit a *curve* to the HI using LOESS.
2. Compute *residue*:  $\text{residue} = \text{HI} - \text{curve}$ .
3. Compute *curve<sub>cm</sub>*:  $\text{curve}_{cm}(t) = \max[\text{curve}(1:t)]$ .
4. Compute the post-processed HI:  $\text{HI}_{pp} = \text{residue} + \text{curve}_{cm}$ .

---

Figure 1 provides a visual explanation of this algorithm and its impact on RUL prediction. Figure 1a shows a hypothetical HI and a curve fitted using LOESS. In Fig. 1b and c, which show the HI at  $t = 200$  s and  $t = 300$  s, respectively,

it is evident that the slope of the regression line oscillates between negative and positive values. This undesirable behaviour is resolved after post-processing the HI with Algorithm 1, as shown in Fig. 1d–f.

## 2.2 Vanilla GAN

GANs consist of two neural networks: a generator ( $\mathcal{G}$ ), which takes an input vector of unit Gaussian random noise and generates a synthetic sample, and a discriminator ( $\mathcal{D}$ ), which, given any sample, outputs the probability of it being real (not synthesised). These neural networks are trained in an adversarial framework where the generator is trained to generate increasingly realistic-looking synthetic samples to deceive the discriminator and the discriminator is trained to detect these synthetic samples with higher accuracy (Goodfellow et al., 2020). In the context of SCADA signals, the generator receives a noise vector  $\mathbf{z}_t$  as input and is trained to generate synthetic signals  $\mathbf{s}_t^s$  that match the distribution of the real signals  $\mathbf{s}_t$  used to train the GAN. The discriminator is trained to output a probability of 1 when the data are real and 0 when they are synthetic, minimising the loss function:

$$\mathcal{L}_{\mathcal{D}}^v = -\mathbb{E}_{\mathbf{s}_t \sim p_s} [\log \mathcal{D}(\mathbf{s}_t)] - \mathbb{E}_{\mathbf{z}_t \sim p_z} [\log (1 - \mathcal{D}(\mathcal{G}(\mathbf{z}_t)))] \quad (4)$$

where  $\mathbb{E}$  denotes the expected value. The generator is trained to deceive the discriminator by generating synthetic samples that it classifies as real, assigning a label of 1. It minimises the loss function:

$$\mathcal{L}_{\mathcal{G}}^v = -\mathbb{E}_{\mathbf{z}_t \sim p_z} [\log \mathcal{D}(\mathcal{G}(\mathbf{z}_t))] \quad (5)$$

The overall objective is a min–max game between the generator and the discriminator:

$$\begin{aligned} \min_{\mathcal{G}} \max_{\mathcal{D}} V(\mathcal{D}, \mathcal{G}) = & \mathbb{E}_{\mathbf{s}_t \sim p_s} [\log \mathcal{D}(\mathbf{s}_t)] \\ & + \mathbb{E}_{\mathbf{z}_t \sim p_z} [\log (1 - \mathcal{D}(\mathcal{G}(\mathbf{z}_t)))] \end{aligned} \quad (6)$$

where the discriminator tries to maximise the value function  $V$ , and the generator tries to minimise it.

## 2.3 Proposed method for generating synthetic SCADA signals

A vanilla GAN treats  $\mathbf{S}$  as a group of independent samples  $\mathbf{s}_t$  and models only the signal distribution  $p(\mathbf{s}_t)$ , neglecting the temporal dependencies (Yoon et al., 2019). Furthermore, signals are generated based only on a random noise vector. Therefore, the objective of this work is to develop a GAN-based framework that effectively models  $\mathcal{F}$  in Eq. (1):

1. by conditioning the signal generation on both the condition vector  $\mathbf{C}$  and a random noise vector and

2. by modelling the temporal dynamics of SCADA signals by conditioning the signal generation at each timestamp on information from previous timestamps, in addition to the current one.

In other words, rather than modelling the marginal distribution of the signals  $p(\mathbf{s}_t)$ , the framework should model the conditional distribution  $p(\mathbf{s}_t | \mathbf{c}_{1:t})$ .

The first task above is achieved by adopting a cGAN framework (Mirza and Osindero, 2014), which allows the input of a condition vector at each timestamp  $t$  to the generator and the discriminator, based on which the samples are generated. The loss functions of the discriminator and generator then become

$$\begin{aligned} \mathcal{L}_{\mathcal{D}} = & -\mathbb{E}_{\mathbf{s}_t \sim p_s} [\log \mathcal{D}(\mathbf{s}_t | \mathbf{c}_t)] \\ & - \mathbb{E}_{\mathbf{z}_t \sim p_z} [\log (1 - \mathcal{D}(\mathcal{G}(\mathbf{z}_t | \mathbf{c}_t)))] \end{aligned} \quad (7)$$

$$\mathcal{L}_{\mathcal{G}} = -\mathbb{E}_{\mathbf{z}_t \sim p_z} [\log \mathcal{D}(\mathcal{G}(\mathbf{z}_t | \mathbf{c}_t))] \quad (8)$$

The second task is achieved by using a recurrent neural network (RNN) for both the generator and the discriminator, allowing them to retain information from previous timestamps at each  $t$ . Additionally, as suggested in Yoon et al. (2019), a supervised loss term  $\mathcal{L}_S$  is incorporated into the GAN generator loss function to further enhance temporal consistency.

$$\mathcal{L}_S = \mathbb{E}_{\substack{\mathbf{s}_t \sim p_s \\ \mathbf{z}_t \sim p_z}} \left[ \|\mathbf{s}_t - \mathcal{G}(\mathbf{z}_t | \mathbf{c}_t)\|_2^2 \right] \quad (9)$$

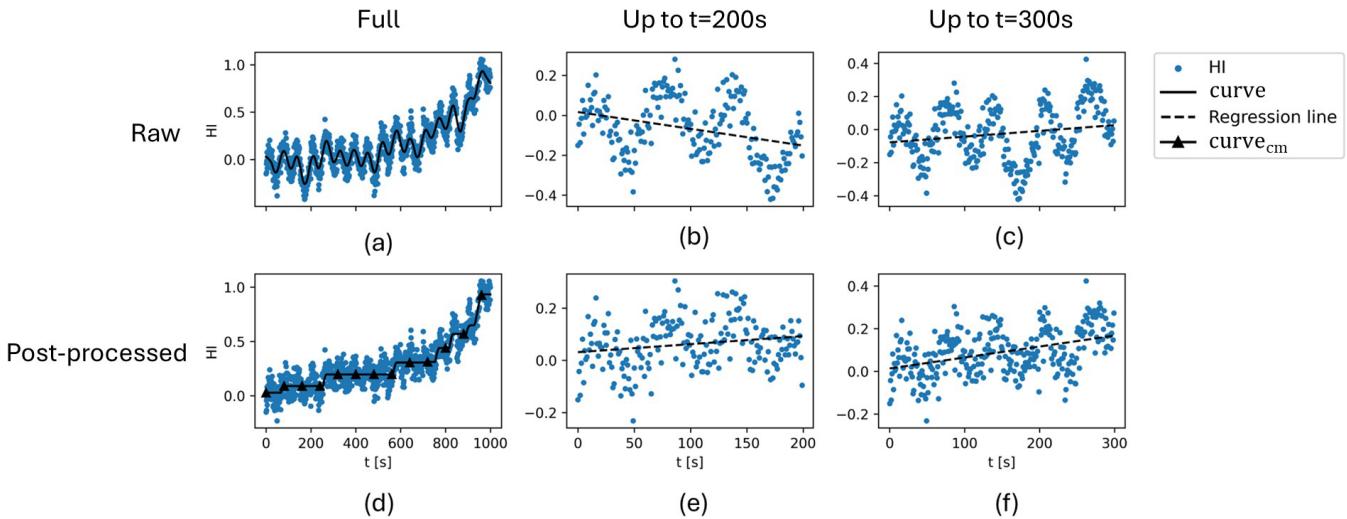
This loss term minimises the Euclidean distance between the true and generated signals during the training phase. The modified generator loss function then becomes

$$\mathcal{L}'_{\mathcal{G}} = \mathcal{L}_{\mathcal{G}} + \mathcal{L}_S \quad (10)$$

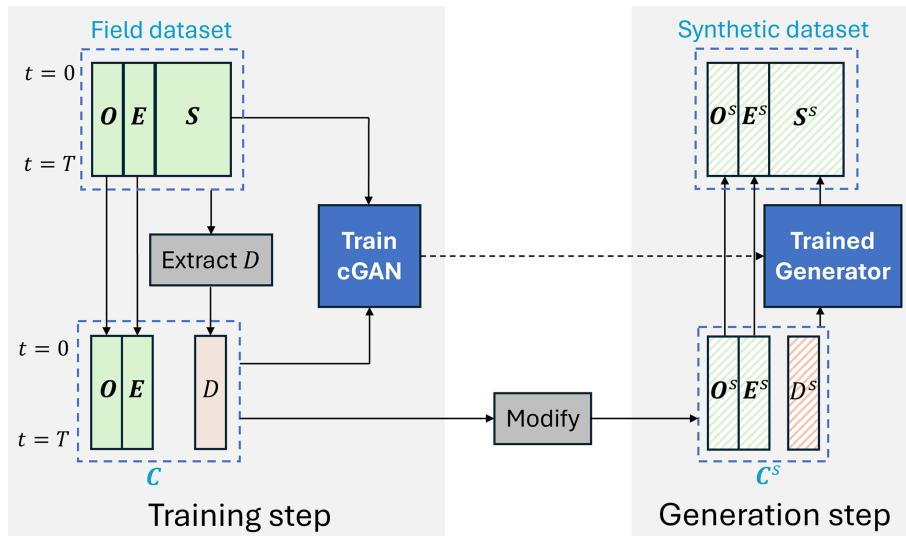
The flowchart of the cGAN-based synthetic SCADA signal generation framework is shown in Fig. 2. In this work, gated recurrent unit (GRU)-based RNNs have been used, as GRU cells achieve performance comparable to Long Short-Term Memory (LSTM) cells while having fewer parameters. This makes them more efficient, faster to train, and less prone to overfitting (Chung et al., 2014). The flowchart of the cGAN model is shown in Fig. 3, where  $y = p(\mathbf{x} = \mathbf{s}_t | \mathbf{c}_t)$ , with  $\mathbf{x}$  being the input to the discriminator.

The architectures of the generator and the discriminator are shown in Fig. 4:

1.  $w$  is the length of a rolling time window applied to  $\mathbf{C}$  before being fed into the GRU layer in both networks. This transforms the shape of  $\mathbf{C}$  from  $(T, N_c)$  to  $(T - w + 1, w, N_c)$  and is shown in Fig. 5. The window length  $w$  introduces a trade-off between model complexity and performance. A larger window allows more information to be captured at each time frame from earlier signal values, at the expense of increased computational burden. In this study, this parameter is set to



**Figure 1.** Example of HI post-processing on a hypothetical HI: (a–c) raw HI, its first 200 s, and its first 300 s and (d–f) post-processed HI, its first 200 s, and its first 300 s.



**Figure 2.** Flowchart of the cGAN-based synthetic SCADA signal generation framework.

10, since the HI obtained from the training wind turbine remained mostly unchanged for larger window length values.

2.  $\Lambda$  denotes the leaky ReLU activation function defined below with  $\alpha = 0.2$ .

$$\Lambda(x) = \begin{cases} x & \text{if } x \geq 0 \\ \alpha x & \text{if } x < 0 \end{cases} \quad (11)$$

In both the generator and the discriminator, the condition signals  $\mathbf{C}$  pass through two recurrent layers with 64 GRU cells. The output vector is concatenated with the random vector in the generator and with the real or synthetic component signals,  $\mathbf{S}$  or  $\mathbf{S}^s$ , in the discriminator. The result is then in-

put to a dense layer with 64 neurons, followed by the output layer, which in the generator outputs the  $\mathbf{S}^s$  signals and in the discriminator the probability of its input signal being real. The hyperparameters are set through trial and error using the training wind turbine data to find a setting that ensures a stable training process, with the generator and the discriminator being trained in tandem and with consistent speeds.

### 3 Dataset

The field dataset used in this work consists of 10 min SCADA signals collected from 1 January 2017 to 1 August 2022 from nine 2 MW wind turbines (WT1–9) in a wind farm operated by Lucky Wind SpA (<https://www.luckywind.it/>, last ac-

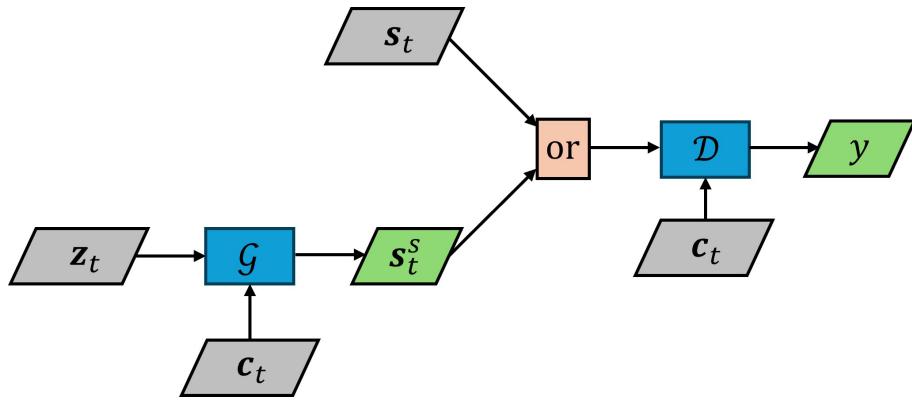


Figure 3. Flowchart of the cGAN model.

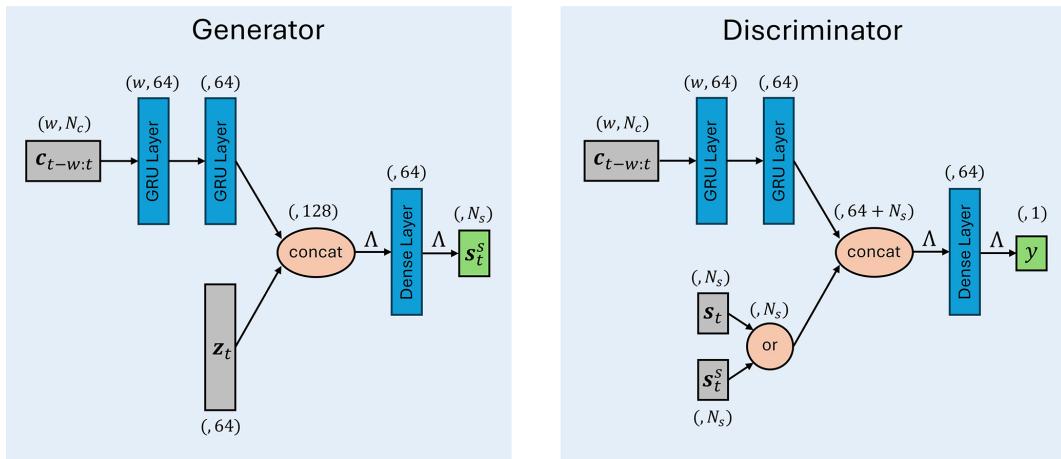


Figure 4. Architecture of the generator (left) and the discriminator (right).

cess: 6 November 2025). Each turbine has a rotor diameter of 100 m, a hub height of 80 m, and a three-stage gearbox with one planetary stage and two parallel stages. Maintenance logs report that one of the wind turbines, WT8, experienced a gearbox failure on 23 February 2022. Inspection logs report widespread debris indentation and circumferential marks in multiple gearbox bearings and abrasion in several gears. A total of 10 signals are selected for analysis: seven related to gearbox operation, two to environmental conditions, and one to operational conditions. These signals are reported in Table 1. The SCADA signals from WT8 during the year leading to the gearbox failure are used for training purposes. In addition, WT9 data are used as the validation set, and the remaining seven wind turbines are used as the test set.

### 3.1 Data preprocessing

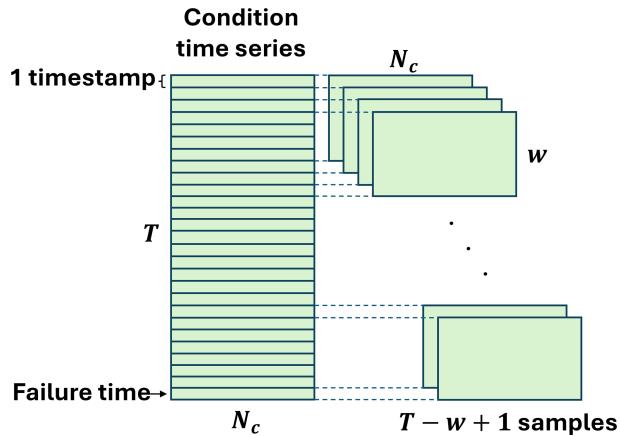
Data preprocessing in this study is limited to omitting non-physical signal values and those corresponding to non-operational turbine conditions. Non-physical signal values refer to instances where gearbox-related temperatures are

lower than the ambient temperature, gearbox oil pressure is equal to zero, and rotor speed is negative. These outliers, likely caused by sensor malfunctions, constitute around 5 % of the total data. Non-operational values occur when the turbine idles and the produced power is negative. They constitute around 20 % of the total data. As a result, around 25 % of the original data points are excluded from the analysis during the data preprocessing step.

After eliminating these outliers, the signals are resampled into 6 h time intervals. Increasing the sampling period reduces the number of data points, leading to a lower computational burden. It also contributes to a higher signal continuity by reducing the percentage of missing data. However, this comes at the cost of decreased fault detection and RUL prediction accuracy. A 6 h resampling period has been identified as a good trade-off between these factors. As a result, the number of data points in a year drops from around 52 000 to around 1400, and the percentage of missing data decreases from around 25 % to around 3 %.

**Table 1.** Selected SCADA signals.

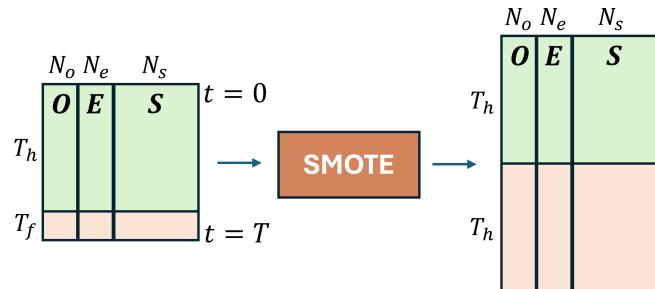
Gearbox-related signals ( $S$ )	Environmental condition signals ( $E$ )	Operational condition signals ( $O$ )
Gearbox bearing A temperature	Ambient wind speed	Rotor speed
Gearbox bearing B temperature	Ambient temperature	
Gearbox bearing C temperature		
Gearbox oil temperature		
Gearbox oil temperature at the inlet		
Gearbox oil pressure before filter		
Gearbox oil pressure after filter		

**Figure 5.** Applying a rolling time window of length  $w$  to the condition time series.

#### 4 Fault detection case study

In this section, an experiment is performed to assess the effectiveness of the developed synthetic data generation method in fault detection by comparing the detection performance with and without synthetic data. A classifier is trained with the training (WT8) signals spanning 23 February 2021 to 23 February 2022 (the gearbox failure time) to identify faulty gearbox operation. The timestamps during the last month before failure are labelled as faulty (1), while the remaining 11 months is labelled as healthy (0). Selecting a smaller portion of data as the faulty class can reduce false positives. However, it leads to a higher data imbalance and can reduce model performance. In this work, the length of 1 month constitutes the minimum length of the faulty training data that maintain an acceptable level of data imbalance, allowing for successful training of the fault detection model.

The classifier used is an artificial neural network (ANN) with three hidden layers consisting of 16, 8, and 4 neurons, each with a ReLU activation function. The output layer uses a sigmoid activation function. The model is trained using the Adam optimiser (Kingma and Ba, 2014) with its default parameters and the binary cross-entropy loss function (Goodfellow et al., 2016). During training, a training-validation split is performed, where 20 % of the training data are ran-

**Figure 6.** Flowchart of the SMOTE oversampling.  $T_h$  and  $T_f$  refer to WT8 time frames labelled as healthy and faulty. They span 11 months and 1 month, respectively.

domly set aside for validation, and the training is stopped when the validation loss stops decreasing for 20 consecutive epochs. The model architecture is set using trial and error, where the model complexity is gradually increased in terms of the number of hidden layers and neurons per layer, until a significant performance improvement is not observed in the validation set. Because of the highly imbalanced nature of the number of healthy and faulty data points in the training set with the minority (faulty) class encompassing only 8.3 % of the data points, the SMOTE method (Chawla et al., 2002) has been used to oversample the faulty data points and balance the two classes. The shape of the training data before and after SMOTE oversampling is shown in Fig. 6.

Once trained, the ANN is used to predict fault state labels (0 or 1) for each timestamp in the SCADA signals of WT1–7 and WT9. State change from healthy to faulty is generally a gradual rather than instantaneous process. Therefore, an individual timestamp predicted as faulty might be due to noise rather than an actual fault (Zhao et al., 2017). For this reason, in Zhao et al. (2017) and Peter et al. (2022), the ratio of the detected anomalous data points to the total number of data points within a fixed time window is used as a fault index. In this work, a similar approach is adopted. A fault index is defined as the weekly moving average (MA) of the predicted labels to improve detection robustness. The flowchart of the fault detection methodology is shown in Fig. 7. The results for the training wind turbine and three test wind turbines are shown in Fig. 8a–d. Numerous misclassified labels,

i.e., a predicted label of 1 when the gearbox is healthy, are observed mainly for WT2 in Fig. 8b and WT7 in Fig. 8d, particularly clustered around February each year, which coincides with the failure month of WT8, used to train the ANN. This suggests that, due to the availability of only one failure event for training, the ANN has learnt the seasonal features in the signals around the failure time of WT8, associating these features with class 1 (faulty).

To address this problem, the developed cGAN-based framework is used to generate synthetic signals that simulate the degradation and failure of the WT8 gearbox across various time frames. To create the condition time series  $\mathbf{C}$ , an HI is built using the proposed HI construction method, representing the degradation of the WT8 gearbox during the year leading to its failure. The obtained HI is shown in Fig. 9. This HI and the three environmental and operational condition signals of WT8 form the condition time series ( $N_c = 4$ ). These, along with the seven gearbox-related signals of WT8 ( $N_s = 7$ ), are used to train the cGAN. The Adam optimiser is used with a learning rate of 0.0005 to train both the generator and the discriminator. This learning rate is lower than the default 0.001 value. Due to the complexities arising from coupling two neural networks within a single training process, this lower learning rate is required to ensure training stability.

Once the cGAN is trained, the environmental (**E**) and operational (**O**) condition signals are shifted from the February 2021–February 2022 time frame to new time frames, April 2020–April 2021, July 2019–July 2020, September 2018–September 2019, and December 2017–December 2018, while keeping the degradation signal ( $D$ ) unchanged. These time frames are selected to simulate the gearbox failure in a variety of seasonal conditions. This results in four new condition time series  $\mathbf{C}_i = \{\mathbf{O}_i, \mathbf{E}_i, D\}$ , where  $i = 1, \dots, 4$ . These condition time series are then input to the trained generator, which generates the corresponding synthetic signal time series  $\mathbf{S}_i^s$ , where  $i = 1, \dots, 4$ . Together with their corresponding environmental and operational signals (**E** and **O**), these synthetic signal time series form four synthetic SCADA datasets. Similar to the original dataset, the initial 11 months in the synthetic datasets is labelled as healthy, while the last month leading to failure is labelled as the faulty class and is oversampled using SMOTE to balance the two classes. The test set predictions using the ANN trained with both the original and the synthetic datasets are shown in Fig. 8e–h for the training wind turbine and three test wind turbines. A noticeable decrease in the number of misclassified labels can be observed when compared to the predictions made using only the original dataset. Across the wind turbines in the healthy test and validation sets, the number of misclassified labels, reported in Table 2, has decreased significantly, by around 84 % on average. A detection threshold of 0.107 is selected for the fault index based on the maximum fault index value observed in the validation wind turbine, the crossing of which indicates a fault. False positives, i.e., the timestamps

when the fault index is above the detection threshold while the gearbox is healthy, are reported in Table 3 and have been almost completely resolved. It is important to note that the number of false positives depends on the detection threshold, which is set based on the randomly selected validation wind turbine (WT9). The component temperatures in different wind turbines can have slight differences in the healthy state. This results in a variability in the number of misclassified labels in different healthy wind turbines reported in Table 2. Therefore, selecting a different wind turbine for validation can change the number of false positives slightly by lowering or raising the detection threshold. However, this affects the number of false positives similarly in the two cases with and without synthetic datasets. Therefore, this experiment is a valid tool for performance comparison. The flowchart of the fault detection methodology with the synthetic datasets is shown in Fig. 10.

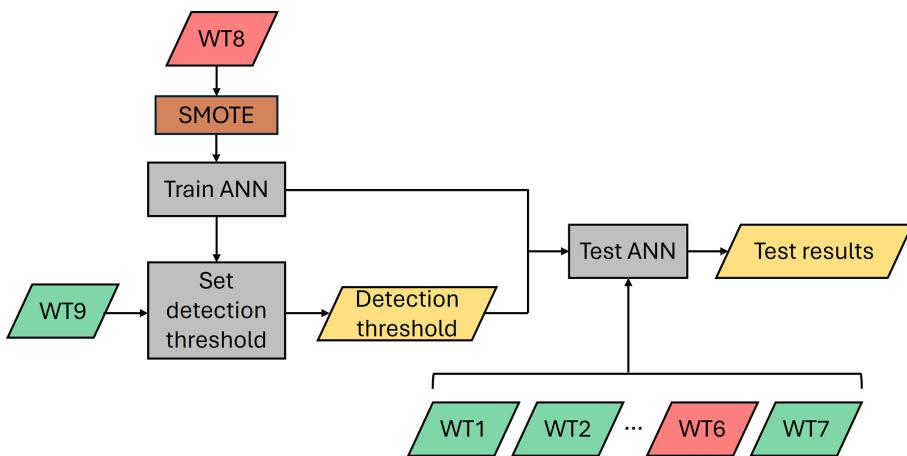
The results show a clear detection of a potential anomaly and fault in WT6. The fault index increases sharply around the end of October 2018 and remains at a lower level until around the end of May 2019. Further investigation into the farm maintenance logs and monthly reports revealed that a moderate gearbox-related fault was discovered and addressed during a 12 h maintenance intervention on 30 May 2019. Coincidentally, this anomaly occurred during the seasonal period where false positives were previously observed in Fig. 8a–d. Therefore, without the synthetic data, detecting this fault with high certainty would have been challenging.

Considering the detection threshold of 0.107, the fault in WT6 is detected on 19 October 2018 at 06:00:00, more than 7 months before the maintenance. Furthermore, the fault in WT8 is detected on 28 August 2021 at 06:00:00, as shown in Fig. 8e, while the actual failure occurred on 23 February 2022.

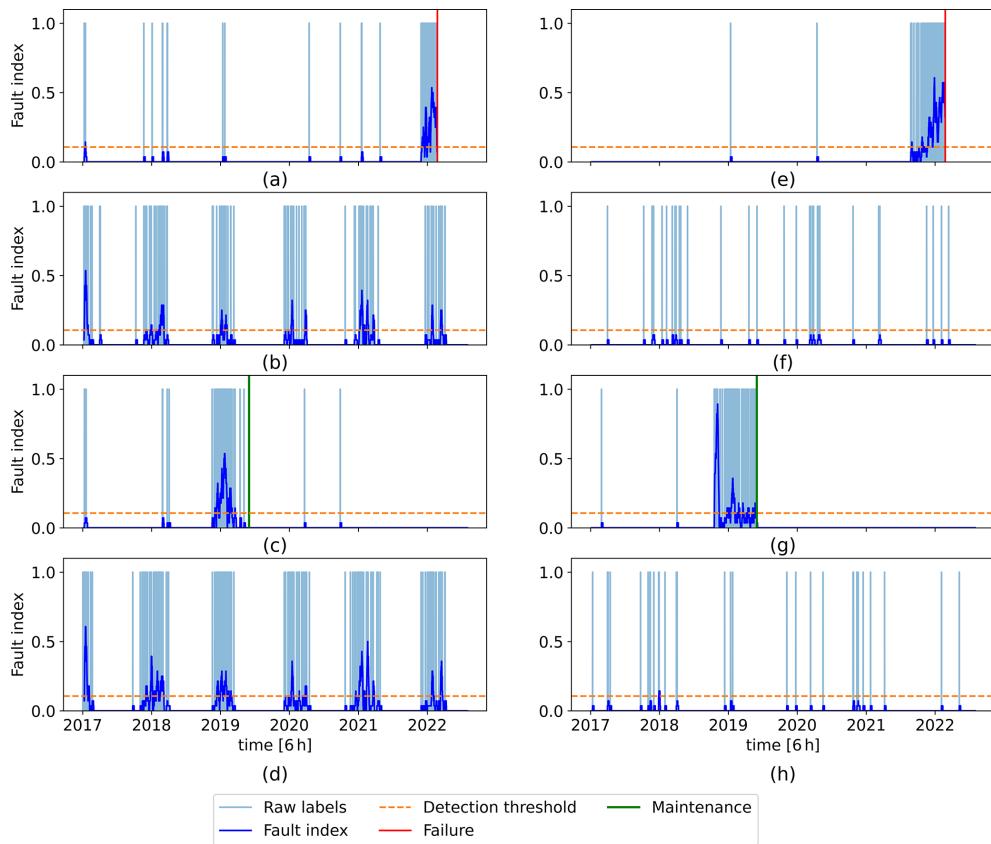
## 5 RUL prediction case study

This section assesses the effectiveness of the developed method in fault prognosis. WT8, used for training, experienced a gearbox failure, while WT6's gearbox underwent maintenance while the fault was at a moderate level, and failure did not occur. As a result, a ground truth failure time is not available for the gearbox of WT6. For this reason, the fault prognosis in this section aims to predict the time when the degradation reaches a level that can be detected by the fault detection method discussed in the previous section.

The CAE used for HI construction is trained with WT8 signals spanning the year leading to the gearbox failure. A threshold of 1 is assigned for gearbox failure, as indicated by the  $f_1$  term in the CAE training fitness function in Eq. (3). Additionally, the mean value of the HI during the week leading to the detection in WT8, which is 0.84, is set as the HI threshold corresponding to the degradation level associated



**Figure 7.** Flowchart of the fault detection without synthetic datasets. The red blocks represent faulty wind turbine data, and the green blocks represent healthy wind turbine data.



**Figure 8.** Predicted raw labels and their weekly moving average (fault index): (a-d) obtained with the original signals and (e-h) with both original and synthetic signals. Panels (a) and (e) refer to the training wind turbine (WT8), (b) and (f) to WT2 (healthy), (c) and (g) to WT6 (faulty), and (d) and (h) to WT7 (healthy).

with the fault detection. Figure 11 shows WT8 HI along with these thresholds. Using the trained CAE, the WT6 HI is then constructed and is shown in Fig. 12a.

To properly train the CAE to extract the true degradation trend, a training set comprising multiple run-to-failure

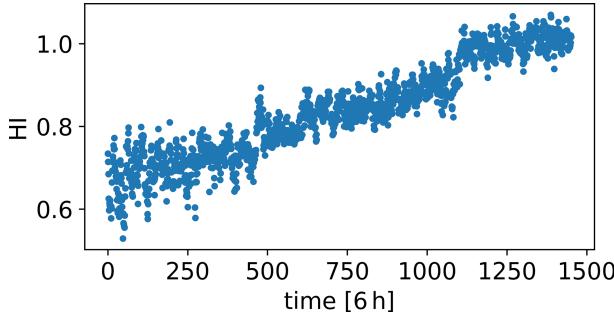
datasets with diverse degradation trajectories is needed. To achieve this, synthetic signals are generated by modifying the degradation trend  $D$  in the condition time series **C** and shifting the environmental **E** and operational **O** signals. The resulting four new condition time series, then, become

**Table 2.** Number of misclassified labels in the healthy test and validation set wind turbines.

	WT1	WT2	WT3	WT4	WT5	WT7	WT9
Without synthetic datasets	21	193	319	125	14	281	356
With synthetic datasets	4	33	80	13	2	33	44

**Table 3.** Number of false positives in the healthy test and validation set wind turbines.

	WT1	WT2	WT3	WT4	WT5	WT7	WT9
Without synthetic datasets	24	508	928	298	22	802	1045
With synthetic datasets	0	0	0	0	0	18	0

**Figure 9.** HI of WT8 during the year leading to the gearbox failure.

$\mathbf{C}'_i = \{\mathbf{O}_i, \mathbf{E}_i, D_i\}$ , where  $i = 1, \dots, 4$ , with  $D_1, \dots, D_4$  representing four different degradation trajectories. To derive  $D_1, \dots, D_4$ , the Complete Ensemble Empirical Mode Decomposition with Adaptive Noise (CEEMDAN) algorithm (Colominas et al., 2014) is applied to decompose the training HI in Fig. 9 into its trend component  $D_{\text{trend}}$  and several other intrinsic mode functions (IMFs),  $D_{\text{IMF},i}$ , which collectively make up the HI noise.

$$D = D_{\text{trend}} + D_{\text{noise}} \quad (12)$$

$$D_{\text{noise}} = \sum_{i=1}^{N_{\text{IMF}}} D_{\text{IMF},i} \quad (13)$$

Here  $N_{\text{IMF}}$  is the number of IMFs other than the trend. The decomposed trend and noise components of the training HI are shown in Fig. 13.

Then, four synthetic trends are created using the equation below:

$$D_{\text{trend},i}(t) = \min(D_{\text{trend}}) + \left[ \max(D_{\text{trend}}) - \min(D_{\text{trend}}) \right] \left( \frac{t}{T} \right)^{p_i}, \quad i = 1, \dots, 4, \quad (14)$$

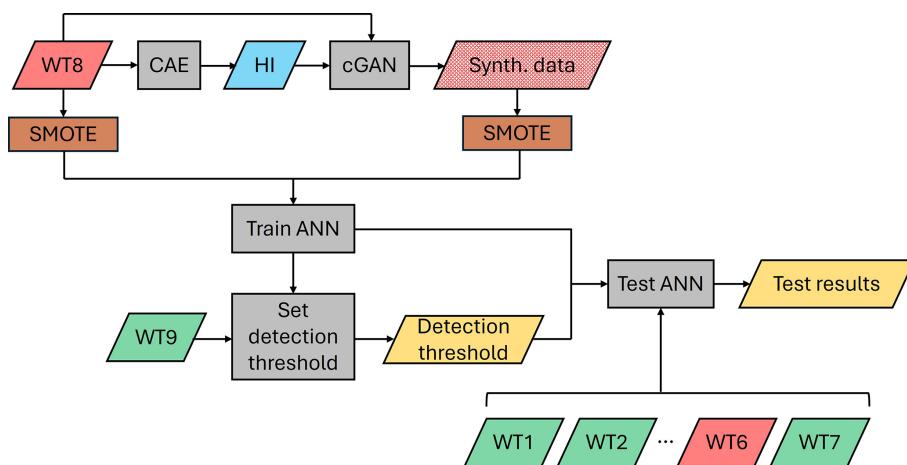
with  $p_{1-4}$  equal to 1, 3, 10, and 50, respectively, and are plotted in Fig. 14. This equation models an HI trend starting at the minimum value of the WT8 HI trend and ending at its

maximum value. It is important to note that assigning a fixed HI value at failure time for all synthetic HIs is valid in the proposed framework, as the model used for HI construction is explicitly trained to associate failure with an HI value of 1. This enforces consistency by design and minimises variability in the HI at failure across different realisations. The exponent  $p_i$  determines the trend regime. A value of 1 produces a linear trend, while higher values indicate more pronounced non-linearity, representing increasingly abrupt degradation patterns. The  $p_{1-4}$  values are chosen to represent meaningfully distinct degradation behaviours.

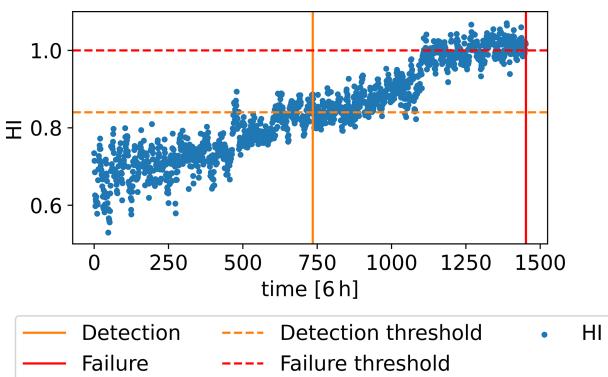
Next, the corresponding synthetic HIs are obtained by adding the WT8 HI noise to the synthetic trends and are shown in Fig. 15. These, combined with the shifted operational and environmental signals, form the four new condition time series  $\mathbf{C}'_1, \dots, \mathbf{C}'_4$ , which are fed into the trained generator to obtain their respective synthetic signal time series  $\mathbf{S}'_1, \dots, \mathbf{S}'_4$ . These synthetic signals, along with the corresponding  $\mathbf{E}$  and  $\mathbf{O}$  signals, create four additional synthetic datasets that are added to the CAE's training set. Subsequently, an HI is constructed for WT6, which is shown in Fig. 12b.

Regardless of the inclusion of synthetic datasets, the HI built for WT6 crosses the detection threshold around the time the gearbox fault is detected and briefly overshoots the failure threshold. However, this exceedance is not sustained, and the HI stabilises between the detection and failure thresholds until the fault is identified and addressed through maintenance, preventing complete failure. These results align well with the behaviour of the fault index in Fig. 8g and the WT6 maintenance report, which confirms a moderate gearbox-related fault. The sudden jump observed both in the fault index and the HI might be due to a sudden fault, such as a crack. However, this hypothesis cannot be asserted with confidence, since no in-depth details are available about this fault case.

Notably, the HI built using synthetic datasets exhibits a clearer degradation trend, achieving a monotonicity of 0.61 (measured by the MK metric) up to the detection point compared to 0.53 when synthetic datasets are not used.



**Figure 10.** Flowchart of the fault detection with synthetic datasets. The red blocks represent faulty wind turbine data, and the green blocks represent healthy wind turbine data.



**Figure 11.** HI of WT8 and the selected thresholds for detection and failure.

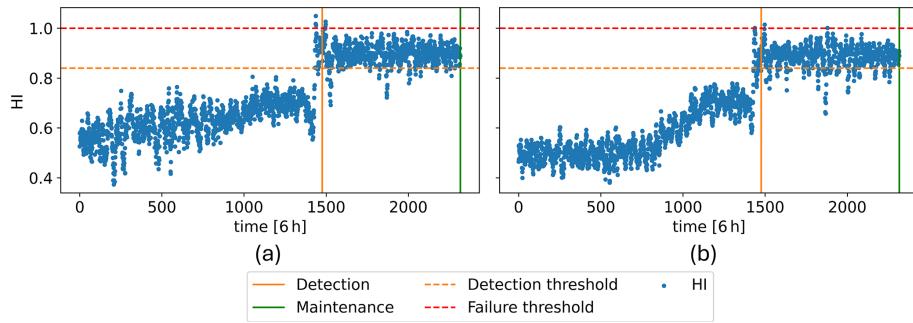
To assess the performance of the two HIs in Fig. 12a and b in predicting the RUL up to the detection point, a second-order polynomial function  $at^2 + bt + c$  is fitted to the HI up to timestamp  $t$  with  $t = 700, \dots, 1470$  (detection time) in 6 h units. The projected crossing point of the fitted function with the detection threshold of 0.84 is then used to calculate the predicted RUL at each  $t$ . To ensure convexity and guarantee that the projected trend crosses the threshold, the constraint  $a \geq 0$  is enforced during curve fitting. This approach can model both linear and curved trends, using the fewest parameters possible, minimising the risk of overfitting. The true and predicted RULs are shown in Fig. 16. At each timestamp, the true RUL corresponds to the number of 6 h timestamps remaining until the detection time. It can be seen that the HI built with the synthetic datasets considerably outperforms the one built without them in correctly predicting the RUL. Considering an error bound of 1 month, this HI achieves a prognosis horizon of around 4.5 months. This implies that using this HI, along with the adopted RUL prediction approach,

enables consistent fault time prediction with the assigned accuracy up to 4.5 months in advance. It is worth noting that a simple RUL prediction approach is used in this work, as developing more complex forecasting approaches is beyond the scope of this work. Adopting more sophisticated approaches could lead to even better RUL prediction performance.

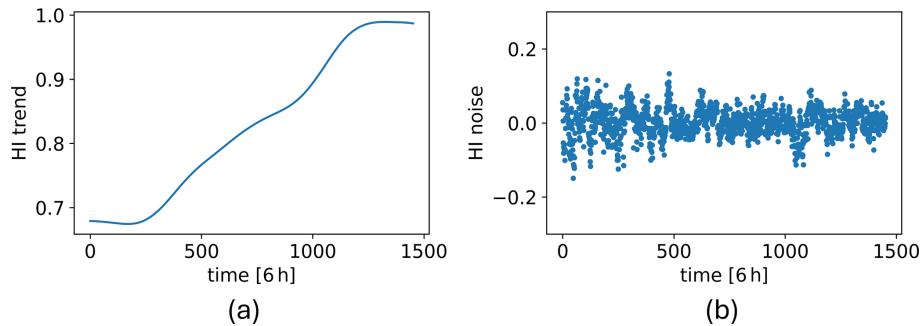
## 6 Conclusions

In this work, a method based on cGANs is developed to generate synthetic SCADA signals. This approach enables the generation of sensor time series with controllable degradation trends, as well as operational and environmental conditions. As a result, it can simulate new failure events or recreate a given failure under different conditions. A fault detection case study demonstrates that this method almost completely resolves false positives. This allows for the blind detection of a fault in one of the test wind turbines more than 7 months before it was discovered and maintained by the wind farm operators. The false positives are due to the availability of only one failure event in the training data, leading the model to associate the seasonal characteristics of signals at the time of failure with fault features. By augmenting the training data with synthetic signals that simulate the same failure under different seasonal conditions, this problem is mitigated. Furthermore, the study shows that training an HI construction method with synthetic signals simulating diverse degradation scenarios leads to a considerably more accurate RUL prediction. This approach enables the prediction of the RUL up to 4.5 months before fault detection in a test wind turbine. In contrast, the HI built without synthetic signals fails to provide accurate RUL estimates.

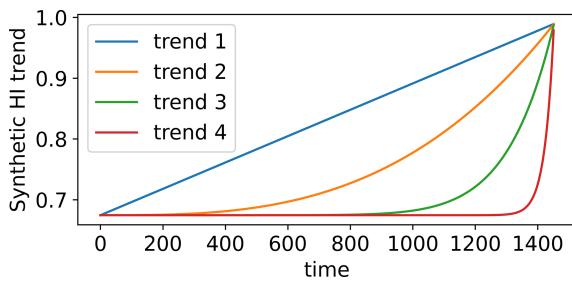
It is important to note that the training and test fault cases in this work are similar, both involving a fault in the gearbox that led to elevated temperatures in the gearbox-related



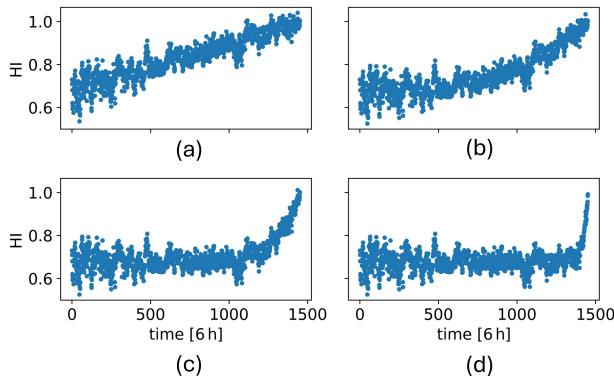
**Figure 12.** WT6 HI: (a) with the original signals and (b) with both original and synthetic signals (b).



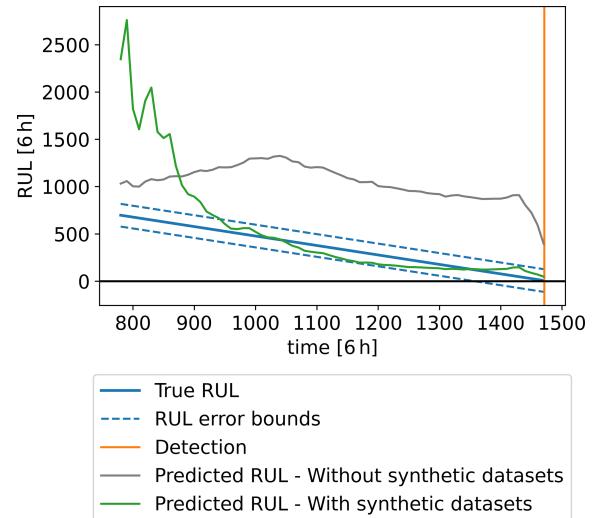
**Figure 13.** Decomposition of the WT8 HI into the (a) trend and (b) noise component.



**Figure 14.** Four degradation trends used to obtain the synthetic HIs.



**Figure 15.** Synthetic HIs used to generate the synthetic signals: (a-d) refer to HI1-4 built with trends 1-4.



**Figure 16.** Comparison between the WT6 true and predicted RULs during the year leading to the fault detection.

SCADA signals. Therefore, the method's performance might deteriorate if the test failure mode is significantly different from the training case. This work serves as a feasibility analysis that proves the proposed approach can generate entire sets of time series simulating new failure events that are able to mitigate the overfitting problem. To identify the extent of diversity in the failure patterns that can be simulated using

this method and assess its performance in the presence of various failure modes, further studies and experiments with multiple failure cases are required.

Accurately forecasting the future trajectory of an HI for RUL prediction remains an important topic that is out of the scope of this paper. Therefore, a simple prediction approach is employed. This approach can perform well when the degradation trend remains consistent throughout the lifetime of the component. However, it can fail in the presence of inconsistent and complex trends. Future research will focus on developing probabilistic approaches for more precise RUL prediction using component HIs.

**Code availability.** The codes used for the analyses and experiments presented in this study are available upon request.

**Data availability.** The dataset used in this research is not publicly accessible due to proprietary restrictions and confidentiality agreements.

**Author contributions.** AEM: conceptualisation, data curation, formal analysis, investigation, methodology, software, validation, visualisation, original draft preparation; DZ: funding acquisition, project administration, resources, supervision, validation, review and editing; FC: data curation, validation, review and editing; SW: funding acquisition, project administration, resources, supervision, review and editing.

**Competing interests.** The contact author has declared that none of the authors has any competing interests.

**Disclaimer.** Publisher's note: Copernicus Publications remains neutral with regard to jurisdictional claims made in the text, published maps, institutional affiliations, or any other geographical representation in this paper. While Copernicus Publications makes every effort to include appropriate place names, the final responsibility lies with the authors. Views expressed in the text are those of the authors and do not necessarily reflect the views of the publisher.

**Acknowledgements.** We thank Lucky Wind SpA for providing the SCADA datasets used in this paper.

**Review statement.** This paper was edited by Nikolay Dimitrov and reviewed by two anonymous referees.

## References

Antoniou, A., Storkey, A., and Edwards, H.: Data Augmentation Generative Adversarial Networks, arXiv, <https://doi.org/10.48550/ARXIV.1711.04340>, 2017.

Chatterjee, J. and Dethlefs, N.: Scientometric review of artificial intelligence for operations & maintenance of wind turbines: The past, present and future, *Renewable and Sustainable Energy Reviews*, 144, <https://doi.org/10.1016/j.rser.2021.111051>, 2021.

Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P.: SMOTE: Synthetic Minority Over-sampling Technique, *Journal of Artificial Intelligence Research*, 16, 321–357, <https://doi.org/10.1613/jair.953>, 2002.

Chesterman, X., Verstraeten, T., Daems, P.-J., Nowé, A., and Helsen, J.: Overview of normal behavior modeling approaches for SCADA-based wind turbine condition monitoring demonstrated on data from operational wind farms, *Wind Energ. Sci.*, 8, 893–924, <https://doi.org/10.5194/wes-8-893-2023>, 2023.

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y.: Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation, arXiv [preprint], <https://doi.org/10.48550/ARXIV.1406.1078>, 2014.

Chu, M., Xie, Y., Mayer, J., Leal-Taixé, L., and Thuerey, N.: Learning temporal coherence via self-supervision for GAN-based video generation, *ACM Transactions on Graphics*, 39, <https://doi.org/10.1145/3386569.3392457>, 2020.

Chung, J., Gulcehre, C., Cho, K., and Bengio, Y.: Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling, arXiv [preprint], <https://doi.org/10.48550/ARXIV.1412.3555>, 2014.

Cleveland, W. S. and Devlin, S. J.: Locally Weighted Regression: An Approach to Regression Analysis by Local Fitting, *Journal of the American Statistical Association*, 83, 596–610, <https://doi.org/10.1080/01621459.1988.10478639>, 1988.

Colominas, M. A., Schlotthauer, G., and Torres, M. E.: Improved complete ensemble EMD: A suitable tool for biomedical signal processing, *Biomedical Signal Processing and Control*, 14, 19–29, <https://doi.org/10.1016/j.bspc.2014.06.009>, 2014.

Eftekhari Milani, A., Zappalá, D., Castellani, F., and Watson, S.: Boosting field data using synthetic SCADA datasets for wind turbine condition monitoring, *Journal of Physics: Conference Series*, <https://doi.org/10.1088/1742-6596/2767/3/032033>, 2024a.

Eftekhari Milani, A., Zappalá, D., and Watson, S.: A hybrid Convolutional Autoencoder training algorithm for unsupervised bearing health indicator construction, *Engineering Applications of Artificial Intelligence*, <https://doi.org/10.1016/j.engappai.2024.109477>, 2024b.

Goodfellow, I., Bengio, Y., and Courville, A.: Deep Learning, MIT Press, <http://www.deeplearningbook.org> (last access: 6 November 2025), 2016.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y.: Generative adversarial networks, *Communications of the ACM*, 63, 139–144, <https://doi.org/10.1145/3422622>, 2020.

Jonkman, B., Mudafort, R. M., Platt, A., Branlard, E., Sprague, M., jjonkman, HaymanConsulting, Vijayakumar, G., Buhl, M., Ross, H., Bortolotti, P., Masciola, M., Ananthan, S., Schmidt, M. J., Rood, J., rdamiani, nrmendoza, sinolong-

hai, Hall, M., ashesh2512, kshaler, Bendl, K., pschuenemann, psakievich, ewquon, mattrphillips, Kusuno, N., alvarogonzalezsalcedo, Martinez, T., and rcorniglion: OpenFAST/openfast: OpenFAST v3.1.0 (v3.1.0), Zenodo, <https://doi.org/10.5281/zenodo.6324288>, 2022.

Kingma, D. P. and Ba, J.: Adam: A Method for Stochastic Optimization, arXiv, <https://doi.org/10.48550/ARXIV.1412.6980>, 2014.

Li, S., Peng, Y., and Bin, G.: Prediction of Wind Turbine Blades Icing Based on CJBM With Imbalanced Data, *IEEE Sensors Journal*, 23, 19726–19736, <https://doi.org/10.1109/jsen.2023.3296086>, 2023.

Li, Y., Pan, Q., Wang, S., Yang, T., and Cambria, E.: A Generative Model for category text generation, *Information Sciences*, 450, 301–315, <https://doi.org/10.1016/j.ins.2018.03.050>, 2018.

Liu, J., Qu, F., Hong, X., and Zhang, H.: A Small-Sample Wind Turbine Fault Detection Method With Synthetic Fault Data Using Generative Adversarial Nets, *IEEE Transactions on Industrial Informatics*, 15, 3877–3888, <https://doi.org/10.1109/tni.2018.2885365>, 2019.

Liu, J., Yang, G., Li, X., Wang, Q., He, Y., and Yang, X.: Wind turbine anomaly detection based on SCADA: A deep autoencoder enhanced by fault instances, *ISA Transactions*, 139, 586–605, <https://doi.org/10.1016/j.isatra.2023.03.045>, 2023.

Liu, S., Li, S., and Cheng, H.: Towards an End-to-End Visual-to-Raw-Audio Generation With GAN, *IEEE Transactions on Circuits and Systems for Video Technology*, 32, 1299–1312, <https://doi.org/10.1109/tcsvt.2021.3079897>, 2022.

Mello, E. d., Kampolis, G., Hart, E., Hickey, D., Dinwoodie, I., Carroll, J., Dwyer-Joyce, R., and Boateng, A.: Data driven case study of a wind turbine main-bearing failure, *Journal of Physics: Conference Series*, 2018, 012011, <https://doi.org/10.1088/1742-6596/2018/1/012011>, 2021.

Mirza, M. and Osindero, S.: Conditional Generative Adversarial Nets, arXiv, <https://doi.org/10.48550/ARXIV.1411.1784>, 2014.

Peng, C., Chen, Q., Zhang, L., Wan, L., and Yuan, X.: Research on fault diagnosis of wind power generator blade based on SC-SMOTE and kNN, *Journal of Information Processing Systems*, 16, 870–881, <https://doi.org/10.3745/JIPS.04.0183>, 2020.

Peter, R., Zappalá, D., Schamboeck, V., and Watson, S. J.: Wind turbine generator prognostics using field SCADA data, *Journal of Physics: Conference Series*, 2265, 032111, <https://doi.org/10.1088/1742-6596/2265/3/032111>, 2022.

Pohlert, T.: Non-Parametric Trend Tests and Change-Point Detection, <https://doi.org/10.13140/RG.2.1.2633.4243>, 2015.

Pujana, A., Esteras, M., Perea, E., Maqueda, E., and Calvez, P.: Hybrid-Model-Based Digital Twin of the Drivetrain of a Wind Turbine and Its Application for Failure Synthetic Data Generation, *Energies*, 16, 861, <https://doi.org/10.3390/en16020861>, 2023.

She, D. and Jia, M.: Wear indicator construction of rolling bearings based on multi-channel deep convolutional neural network with exponentially decaying learning rate, *J. Measurement*, 135, 368–375, <https://doi.org/10.1016/j.measurement.2018.11.040>, 2019.

Shorten, C. and Khoshgoftaar, T. M.: A survey on Image Data Augmentation for Deep Learning, *Journal of Big Data*, 6, <https://doi.org/10.1186/s40537-019-0197-0>, 2019.

Song, F., Han, Y., William Heath, A., and Hou, M.: Structural damage detection of floating offshore wind turbine blades based on Conv1d-GRU-MHA network, *Engineering Failure Analysis*, 166, 108896, <https://doi.org/10.1016/j.engfailanal.2024.108896>, 2024.

Spinato, F., Tavner, P., van Bussel, G., and Koutoulakos, E.: Reliability of wind turbine subassemblies, *IET Renewable Power Generation*, 3, 387, <https://doi.org/10.1049/iet-rpg.2008.0060>, 2009.

Tao, C., Tao, T., He, S., Bai, X., and Liu, Y.: Wind turbine blade icing diagnosis using B-SMOTE-Bi-GRU and RFE combined with icing mechanism, *Renewable Energy*, 221, 119741, <https://doi.org/10.1016/j.renene.2023.119741>, 2024.

Tatsis, K., Dertimanis, V., Abdallah, I., and Chatzi, E.: A substructure approach for fatigue assessment on wind turbine support structures using output-only measurements, *Procedia Engineering*, 199, 1044–1049, <https://doi.org/10.1016/j.proeng.2017.09.285>, 2017.

Tatsis, K., Ou, Y., Dertimanis, V. K., Spiridonakos, M. D., and Chatzi, E. N.: Vibration-based monitoring of a small-scale wind turbine blade under varying climate and operational conditions. Part II: A numerical benchmark, *Structural Control and Health Monitoring*, 28, <https://doi.org/10.1002/stc.2734>, 2021.

Wang, A., Qian, Z., Pei, Y., and Jing, B.: A de-ambiguous condition monitoring scheme for wind turbines using least squares generative adversarial networks, *Renewable Energy*, 185, 267–279, <https://doi.org/10.1016/j.renene.2021.12.049>, 2022.

Yang, N., Liu, S., Liu, J., and Li, C.: Assessment of Equipment Operation State with Improved Random Forest, *International Journal of Rotating Machinery*, 2021, 1–10, <https://doi.org/10.1155/2021/8813443>, 2021.

Yang, Z., Baraldi, P., and Zio, E.: A method for fault detection in multi-component systems based on sparse autoencoder-based deep neural networks, *Reliability Engineering and System Safety*, 220, 108278, <https://doi.org/10.1016/j.ress.2021.108278>, 2022.

Yoon, J., Jarrett, D., and van der Schaar, M.: Time-series Generative Adversarial Networks, in: *Advances in Neural Information Processing Systems*, edited by Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., 5508–5518, <https://proceedings.neurips.cc/paper/2019/file/c9efe5f26cd17ba6216bbe2a7d26d490-Paper.pdf> (last access: 6 November 2025), 2019.

Zhao, Y., Li, D., Dong, A., Kang, D., Lv, Q., and Shang, L.: Fault Prediction and Diagnosis of Wind Turbine Generators Using SCADA Data, *Energies*, 10, 1210, <https://doi.org/10.3390/en10081210>, 2017.