1D-DGAN-PHM

A 1-D denoising GAN for Prognostics and Health Management with an application to turbofan

Baptista, Marcia L.; Henriques, Elsa M.P.

DOI
10.1016/j.asoc.2022.109785

Publication date
2022

Document Version
Final published version

Published in
Applied Soft Computing

Citation (APA)
Baptista, M. L., & Henriques, E. M. P. (2022). 1D-DGAN-PHM: A 1-D denoising GAN for Prognostics and Health Management with an application to turbofan. *Applied Soft Computing*, *131*, Article 109785. https://doi.org/10.1016/j.asoc.2022.109785

Applied Soft Computing 131 (2022) 109785

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

1D-DGAN-PHM: A 1-D denoising GAN for Prognostics and Health Management with an application to turbofan

Marcia L. Baptista [a,*], Elsa M.P. Henriques [b]

[a] Section of Air Transport and Operations, Delft University of Technology, Aerospace Engineering Faculty, Delft, The Netherlands
[b] Instituto Superior Tecnico, Universidade de Lisboa, Lisbon, 1049-001, Portugal

ARTICLE INFO

Article history:
Received 16 May 2021
Received in revised form 5 October 2022
Accepted 2 November 2022
Available online 8 November 2022

Keywords:
Prognostics and Health Management
Failure prognostics
Generative Adversarial Network
Signal processing
Denoising
Edge-preserving method

ABSTRACT

The performance of prognostics is closely related to the quality of condition monitoring signals (e.g., temperature, pressure, or vibration signals), which reveal the degradation of the system of interest. However, typical condition monitoring signals include noise and outliers. Disentangling noise from these signals is essential to obtain the actual degradation trajectories. Different denoising methods have been proposed in prognostics. Conventional denoising methods have low complexity but usually do not preserve edge information and do not involve physical considerations. A promising deep learning approach is denoising generative models. This approach is popular in Computer Vision, which has been shown to outperform other classical techniques but has seldom been used in prognostics on 1-D signals. In this paper, we propose the 1-D Denoising Generative Adversarial Network for Prognostics and Health Management (1D-DGAN-PHM). The 1D-DGAN-PHM is trained on synthetic data generated by a custom data generator that infuses physics-of-failure knowledge in paired samples of noisy and noise-free trajectories. The network consists of two components, a denoising generator and a discriminator. The denoising generator aims to learn to denoise a 1-D input signal. The discriminator guides the learning by comparing noise-free signals with signals from the denoising generator. Advantages of the 1D-DGAN-PHM include the physics-of-failure information in the synthetic data generator and the model sophistication. In this work, we apply the 1D-DGAN-PHM to denoise the raw signals derived from NASA's C-MAPSS simulator of an aircraft turbofan engine. Baseline methods are Moving Average, Median filter, Savitzky–Golay filter, and a denoising autoencoder. The 1D-DGAN-PHM produces smooth trajectories and preserves the initial linear degradation of the signals. The 1D-DGAN-PHM has the most significant improvement in prognosability (on average, 0.73 to 0.81). Data from the 1D-DGAN-PHM resulted in the best MAE (29 to 25 cycles) and RMSE (score of 39 to 36) for a Random Forest.

The code is publicly available at 1D-DGAN-PHM.

1. Introduction

Fault diagnostics and failure prognostics can be understood as the science of making predictions about engineering systems [1]. The fundamental questions of Prognostics and Health Management (PHM) are "When will it break?", "What is the level of degradation so far?" or "When or why did the fault initiate?". These questions can be answered in PHM by applying algorithms to condition monitoring data based on physics-based principles, data-driven techniques, or hybrid approaches [2].

An issue frequently encountered in diagnostics and prognostics is how to account for and reduce measurement uncertainty. This kind of uncertainty relates to condition monitoring data being contaminated with noise. Noise is unavoidable in sensor systems, where there is always a region of uncertainty around the actual signal. Even though it may not always be possible to eliminate all the noise, especially for low-level sensors, this uncertainty element can considerably degrade the performance of prognostics (and diagnostics) activities. Therefore, developing and using signal processing methods to address this issue is critical.

The search for efficient denoising (or smoothing) methods is still a demanding problem in prognostics with no unique solution [3–7]. It remains a challenge to remove the noise of a sensor signal while preserving the underlying damage trajectory both in system condition monitoring and in structural condition monitoring [8,9]. Part of the problem concerns balancing noise reduction and loss of information. This trade-off is at the core of signal denoising in PHM: reaching a compromise between removing noise and preserving details of the original health indicator. The

* Corresponding author.
E-mail address: m.lbaptista@tudelft.nl (M.L. Baptista).

https://doi.org/10.1016/j.asoc.2022.109785

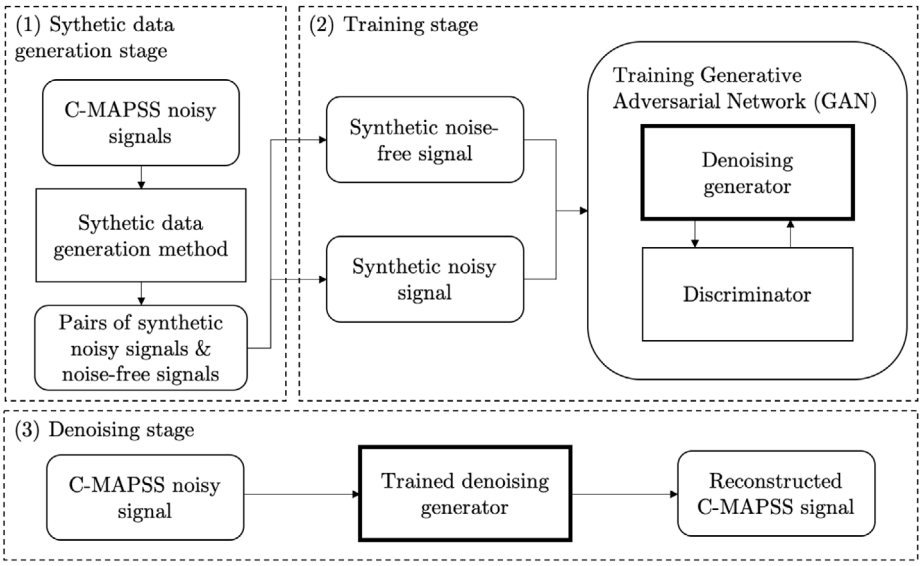


Fig. 1. Denoising stages. The Generative Adversarial Network (GAN) consists of a denoising generator and a discriminator. The generator tries to produce signals close to the noise-free signals. The objective of the discriminator is to identify which of the received signals are noise-free signals or mere artificial reconstructions from the generator. After several rounds of adversarial training, the quality of the reconstructed signal gradually increases, and the signals become more smooth.

goal is to balance several factors simultaneously, such as the noise level, the signal, and the calculation time.

There are several conventional methods to denoise 1-Dimensional (1-D) signals [10]. Techniques such as Moving Average or Median Filter have low time complexity and require little effort to implement. However, these methods typically do not preserve edge information and do not make use of domain information. Deep learning techniques [11] hold promise for denoising applications and one such technique is Generative Adversarial Networks (GANs) [12].

We propose a GAN model to denoise 1-D signals. Our network, the 1D-DGAN-PHM (1-D Denoising Generative Adversarial Network for Prognostics and Health Management), consists of a denoising generator and a discriminator. The denoising generator aims to learn how to denoise the input signal. The discriminator guides the learning process by comparing the ground truth signal, without noise, and the (denoised) signal that the generator reconstructs. The network is trained with synthetic data, consisting of pairs of noise-free and noisy signals developed for the specific case study. The network is general enough to be adapted to different case studies by adjusting the synthetic data generator to the characteristics of the condition monitoring data. The flowchart of our proposed approach is shown in Fig. 1.

In this paper, we show that the proposed 1D-DGAN-PHM removes noise from condition monitoring signals while preserving the degradation patterns in the data. Also, the algorithm is an edge-preserving smoothing technique. Other smoothing techniques such as Median Filtering or Moving Average Filtering can effectively remove noise in some areas of the signal but do not preserve edges. Edges are of critical importance to prognostics. For example, the first points of a health monitoring trajectory are essential to classify and discover the inflection point (i.e., the start of the functional failure) of the degradation trajectory. In contrast, the last points of a signal are essential to produce correct Remaining Useful Life (RUL) estimations.

This research aims to compare the performance of the 1D-DGAN-PHM network with other methodologies from the PHM community. We apply several denoising techniques to C-MAPSS data (FD001 dataset). These data was generated from the Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) [13], and was publicly released by NASA [14]. It is currently available at NASA's Prognostic Data Repository.

The contribution of this paper is a denoising algorithm based on a generative modeling approach. We explain how we infuse physics-of-failure knowledge into the training data using a custom synthetic data generator. We apply different metrics from prognostics to assess the quality of the denoising. We also evaluate the influence of the denoising on prognostics. The RUL estimation is tested on different Machine Learning (ML) models. The goal here was to test a simple prognostics approach.

The remainder of this article is organized as follows. Section 2 reviews denoising techniques in signal processing and in prognostics. Section 3 describes the case study and methodology. Results are presented and discussed in Section 4. Section 5 concludes the article where recommendations and future work are discussed.

2. Background

Prognostics relies on the efficacy of sensors placed at critical location points to record different kinds of data, such as temperature, pressure, oil debris, acoustic signals, and other variables. Since sensors are not always reliable, and there is always a degree of uncertainty in any measurement, operational data can be of low quality. Two main kinds of errors can occur [15]: systematic errors and random errors. The main difference between these two kinds of error is that random errors result in unpredictable fluctuations around the actual signal values, whereas systematic errors lead to consistent deviations.

Denoising consists of suppressing the signal's error part to approximate the original signal. Despite the apparent simplicity of the denoising task, it is usually non-trivial to reconstruct data obtained from harsh environments (e.g., engine data). This difficulty follows mainly from the different sources of noise in the data. Noise sources can have a different impact along the measurement process resulting in complex compound effects on the sensor signals.

Different approaches have been proposed in the literature to perform the denoising of one-dimensional signals. In prognostics, Blind Source Separation (BSP) is a popular approach. It separates the original signal into a set of factors. Independent Component Analysis (ICA) is one technique to perform blind source separation. Several authors [16–22] have used ICA in PHM. However,

despite its effectiveness, ICA is mostly suited for signals subject to systematic errors. Separation techniques such as ICA do not deal well with random errors.

Adaptive Noise Cancellation (ANC) is another form of denoising [23]. The idea is to reduce noise by adding a second signal that cancels the noise component. ANC techniques necessitate little to no knowledge about the signal and its noise properties.

Even though the ANC was originally based on two sensors, the Self-Adaptive Noise Cancellation (SANC) approach extended the methodology for a one-sensor case by replacing the noise reference with a lagged version of the original noisy signal. Antoni and Randall [24] provided a brief review on SANC applied to vibration signals. It is not convenient to use SANC when the signal-to-noise ratio is poor and non-linear effects are not negligible. These and other issues have been described by Antoni and Randall [24], who proposed solutions to address some of these problems.

Another denoising technique is Fourier filtering. The noisy signal is decomposed based on sine and cosine functions using the Fourier transform. This kind of filtering aims to attenuate or amplify specific frequencies. The idea is that the Fourier spectrum of the signal exhibits some peaks that correspond to the noise frequencies and that it is possible to attenuate them. A final inverse transformation is necessary to obtain the desired result. The removal of the Fourier spectrum peaks has a significant drawback: the abrupt removal of all the Fourier coefficients can introduce artifacts and remove critical frequencies from the original noiseless signal. Notwithstanding, the method is one of the most popular techniques to deal with data corrupted by periodic noise, namely vibration signals. Contributions of note to PHM are the works in Ref. [25–29]. A review of these algorithms on bearing vibration data is provided by Lin and Ye [30].

Denoising techniques based on wavelets [31] have been studied for quite some time in signal processing. The basic idea is similar to that of Fourier filtering. The goal is to represent the noisy signal as a composition of functions, in this case, wavelet functions, generated as scaled and translated versions of a particular mother wavelet. Antoni and Randall [32] provides a review of the different wavelet-based denoising methods.

Wavelet denoising methods are appropriate for non-stationary noise, and several works in PHM have applied them to vibration data [27,33–36]. The major limitation of wavelet-based methods is the difficulty of selecting the most appropriate wavelet function and determining the number of decomposition levels.

The previously explained methods are tailored for periodic noise, but other alternative approaches can deal well with random noise. Probably the oldest (see Ref. [37]) and simplest denoising method is the Simple Moving Average (SMA) filter. This filter is a Finite Impulse Response (FIR) filter where each denoised value is the mean of the previous measurements. The Exponential Moving Average (EMA) is another popular FIR filter that can be calculated recursively.

Despite the simplicity of SMA and EMA, these methods tend to work well on different kinds of data as they discover both linear and non-linear trends. The EMA gives greater weight to the most recent measurements, whereas the SMA gives equal importance to all values. Therefore, the EMA filter responds to signal changes faster and captures slight signal fluctuations. Trinh and Kwon [38] compared the performance of different filtering approaches, including the SMA and EMA, on two benchmark datasets of RUL estimation. Among their findings, the authors recognized the difficulty of selecting a single filtering approach for an application due to the overfitting effects that can occur when adjusting the filters to the training data.

Another interesting study is by Kohler and Lorenz [39] who compared a wide range of techniques and evaluation metrics on different noisy signals. The authors found no clear winner and argued that end performance depends on the error measure. Also, the authors concluded that the EMA filter was unsuitable for all scenarios/error measures.

The most negative aspect of FIR filters, such as SMA of EMA, are limitations when dealing with signals of finite duration. For example, the SMA and EMA do not have data to calculate the initial denoised points. This issue can be critical in prognostics, where the initial and last points of the degradation trajectories are of central importance for RUL estimation. For example, the initial measurements can signal the existence of an inflection point and the start of observable degradation. In turn, the final measurements are essential to diagnose the failure. No optimal solution has been found, from a practical viewpoint, to deal with these edge distortions. Some of these techniques, such as zero-padding, symmetrization, and smooth padding, are reviewed by Gopinath and Burrus [40].

Recent studies in deep learning have shown that neural models can deal with complex multivariate noisy data and successfully reconstruct the original noiseless signals. One such approach is the Denoising Auto-Encoder (DAE) proposed by Vincent et al. [41]. This techniques is for robust representation learning given corrupted data as input (see Ref. [42] for detailed explanation).

Motivated by the successful application of DAE in diverse areas such as speech recognition [43–46], human activity recognition [47], analysis of biomedical data such as ECGs [48–50], music source separation [51], energy load forecasting [52] and computer vision [53–56], a number of scholars in the PHM community started to adopt DAE to denoise sensor data. For example, Meng et al. [57] proposed a modified version of the classical DAE for rolling bearing fault diagnosis. A comparison with a standard DAE validated the model outcome. Another example of the use of DAE in fault diagnostics is the work of Liu et al. [58]. The authors combined a non-linear DAE with a Recurrent Neural Network (RNN) which improved the data reconstruction.

Even though some authors have used DAE in PHM, the use of the Stacked Denoising Auto-Encoder (SDAE) has become prevalent. The SDAE was originally proposed by Vincent et al. [59] as a deep network created by stacking several layers of DAEs. The main advantage of an SDAE over a DAE is its generalization power, which allows the network to learn compact input representations more effectively.

In fault diagnostics, Guo et al. [60] used a Stacked Denoising Auto-Encoder (SDAE) to denoise and extract relevant features from raw vibration signals. The authors showed that this kind of network successfully reduces aleatory uncertainty (random noise) for bearing rolling and gearbox fault applications. Several authors [61–63] have combined the SDAE technique with other methods. For example, Wang et al. [61] combined an SDAE with a Deep Belief Network (DBN) to learn more complex feature representation. The drawback of this approach is a large number of parameters of the model.

Hu et al. [62] used Kernel Principal Component Analysis (KPCA) as a preprocessing step for the SDAE algorithm. The KPCA is used to reduce the dimensionality of the data and remove non-relevant information. Liu et al. [63] combined SDAE with hierarchical Bayesian modeling to diagnose and forecast fault events. The main disadvantage of SDAE is that to perform denoising, the model needs to successfully capture the major structural patterns of the input in the hidden layers, which may not always happen.

A model that has shown quite promise in a wide range of tasks is the Generative Adversarial Networks (GANs). Originally proposed by Goodfellow et al. [12], who outlined the learning theory of GANs on a game theoretic scenario, these networks have since shown remarkable capability in diverse domains, such as image generation [64–67], text to image translation [68,69], and image to image translation [70–72]. In simple terms, a GAN

consists of two networks: a generator and a discriminator. During the adversarial training process between the two models the generator learns how to produce data based on some input. The discriminator attempts to distinguish between samples from the synthetic database of noiseless images and images generated by the GAN generator.

The GAN technique is not so explored in PHM. However, a work of note is by Wang et al. [73] who addressed the problem of denoising vibration signals in planetary gearboxes by combining GAN and SDAE. In their work, the GAN is used to capture the distribution of the original signals and generate new signals with a similar distribution in order to augment the number of training fault samples. The denoising itself is performed by the SDAE. Another similar work to ours is by [74] who used a GAN to expand the set of diagnostics samples for rotating machinery. A work of note is that by Lyu et al. [75] who performed restoration of images corrupted by mixed noise using a GAN.

We note the singularity of our contribution: in this work, we use the GAN, not for data generation (data augmentation), but for 1-D signal denoising. This contribution is motivated by the promising results of denoising GANs in the computer vision domain. To our best knowledge, this kind of utilization of GANs has not been sufficiently explored in the PHM domain.

To the best of our knowledge, no other paper has extensively studied the technique of denoising GAN for prognostics. The basic idea of the 1D-DGAN-PHM approach is to build a synthetic training dataset composed of pairs of noisy/clean signals and then to train a deep denoising Generative Adversarial Network (GAN) to remove noise from unseen 1-dimensional signals. This approach is tailored to PHM solutions.

3. Methodology

This section starts by describing the research question and the study's goal. We then describe the data and case study. We describe the methods, metrics, and methods used to achieve the research objectives. In this section, we first present the problem (Section 3.1) and then the research question (Section 3.2). We describe the case study in Section 3.3. The measures of feature importance are described in Section 3.4. We conclude this section by describing the main properties of the synthetic data generator (Section 3.5) and the 1D-DGAN-PHM network (Section 3.6). Section 3.7 describes the prognostics evaluation.

3.1. Problem

In prognostics, we consider the evolution of N engineering assets (or units) designated as $u = \{0, \ldots, N\}$. Time is designated as $n = \{1, \ldots, T\}$. A set of features characterizes the behavior of each asset. We define the problem of denoising as the transformation of the mth feature from unit u to a cleaner signal designated as $g_m^u(n)$ after observing the feature evolution over a time interval of duration $n \in [0, t_{obs}]$. The denoised signal $g_m^u(n)$ approximates the original clean signal $x_m^u(n)$:

$$g_m^u(n) \approx x_m^u(n) \tag{1}$$

To investigate this study's research question, we follow the methodology described in Section 3.2.

3.2. Research question

The research problem relates to developing more accurate methods to denoise 1-D trajectory data in PHM. The research question is the following:

RQ: How can we improve the denoising of 1-D degradation trajectories in PHM?

For this research question, we have the hypothesis:

H: Generative adversarial networks (GAN) can outperform the performance of baseline methods for denoising PHM data.

Vibration data and data under different operational conditions are not in the scope of this study. However, in theory, applying the studied denoising methods to these kinds of data should be possible. To investigate the research question of this study, we follow the steps:

- *Generate Synthetic Data*: A dataset of synthetic trajectories is created for training the 1D-DGAN-PHM network and the baseline DAE.
- *Train Model*: The synthetic data is used to train the 1D-DGAN-PHM network and the baseline DAE.
- *Denoise Data*: New data is denoised using the 1D-DGAN-PHM network or other baseline denoising method.
- *Evaluate Features*: The indicators of monotonicity, trendability and prognosability are used to classify the features **after** the denoising.
- *Evaluate Prognostics Performance*: Different ML models are used to assess the prognostics performance of the features **after** the denoising.

In addition to the Generative Adversarial Network (GAN), we investigate the performance of the classical denoising approaches of the Median (Med) filter and Moving Average (MA) filter. We also compare the filter proposed by Savitzky and Golay [76] (SG). In addition to the proposed GAN, we implement a Denoising Auto-Encoder (DAE).

3.3. Data

The data used in this case study is from the Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) developed by NASA [13]. The simulator emulates the operation of a turbofan engine similar to GE90 [13]. The Prognostics Center of Excellence at NASA Research Ames made public four C-MAPSS datasets [77] consisting of several run-to-failure trajectories. The data are currently available at NASA's Prognostic Data Repository.

In C-MAPSS data, an engine asset (or unit) is characterized by 21 prognostics sensors and three additional indicators (Altitude, Mach Number, and Throttle Resolver Angle). This paper studies the first C-MAPSS training dataset (FD001), which comprises data from 100 engines. We selected this dataset as it is the least complex, with only one operating regime and fault mode. Our focus is signal denoising and its effects on Remaining Useful Life (RUL) estimation. Handling other datasets would require different methodologies to baseline the operating conditions and fault modes. In this study, we selected for denoising 14 non-steady-state signals. Table 1 presents the selected features.

3.4. Measures of feature importance

The dependent variable of this study is the quality of the denoising process, which we measure using prognostics indicators. We evaluate the quality of the trajectories after denoising using the popular measures of monotonicity, trendability, and prognosability proposed by Coble and Hines [78]. These metrics have been shown [78–83] to characterize the importance of a feature (or predictor) for RUL estimation. We hereafter explain each metric in detail.

Monotonicity characterizes a predictor's increasing or decreasing trend. Formally, the monotonicity of a feature is

$$\text{monotonicity} = \frac{1}{M}\sum_{j=1}^{M}\left|\sum_{k=1}^{N_j-1}\text{sgn}\left(\frac{x_j(k+1)-x_j(k)}{N_j-1}\right)\right| \tag{2}$$

Table 1
Non steady state condition monitoring features of C-MAPSS data.

Predictor	Description	Units
T24	Total temperature at LPC outlet	°R
T30	Total temperature at HPC outlet	°R
T50	Total temperature at LPT outlet	°R
P30	Total pressure at HPC outlet	psia
Nf	Physical fan speed	rpm
Nc	Physical core speed	rpm
Ps30	Static pressure at HPC outlet	psia
phi	Ratio of fuel flow to Ps30	pps/psi
NRf	Corrected fan speed	rpm
NRc	Corrected core speed	rpm
BPR	Bypass Ratio	–
htBleed	Bleed Enthalpy	–
W31	HPT coolant bleed	lbm/s
W32	LPT coolant bleed	lbm/s

where:

M = number of units
N_j = number of measurements of a feature on unit j
x_j = vector of measurements of a feature on unit j
$x_j(k+1)$ = a measurement of a feature of unit j at time $k+1$
$x_j(k)$ = a measurement of a feature of unit j at time k
sgn = signal function

A more accurate RUL estimation is expected if a feature shows a noticeable increasing or decreasing trend over time. Monotonicity is in the range of [0, 1].

Trendability measures the degree to which the predictor displays the same shape across a group of units. It is a measure of similarity among the trajectories of the population of units. Formally, the trendability of a feature is

$$\text{trendability} = \min_{j,k} \left|\text{corr}(x_j, x_k)\right|, \quad j,k = 1,\ldots,M \tag{3}$$

where:

M = number of units
N_j = number of measurements of a feature on unit j
x_j = vector of measurements of a feature on unit j
x_k = vector of measurements of a feature on unit k
corr = Pearson correlation function

When x_j and x_j have different lengths, we (linearly) interpolate the shortest vector to match the size of the longer vector. The trendability metric is in the range [0, 1] and positively correlates with the importance of the predictor.

Prognosability measures the variance of a predictor at the End of Life (EoL) for the set of units. Formally, the prognosability of a feature is

$$\text{prognosability} = -\exp\left(\frac{\text{std}_j(x_j(N_j))}{\text{mean}_j|x_j(1) - x_j(N_j)|}\right) \tag{4}$$

where:

M = number of units
j = index of unit ($j = 1,\ldots,M$)
N_j = number of measurements of a feature on unit j
x_j = vector of measurements of a feature on unit j
x_k = vector of measurements of a feature on unit k
mean_j = average function of all units

Prognosability is within the range of [0, 1]. The prognostics' performance is expected to be higher when the prognosability is close to 1.

3.5. Synthetic data generator

An essential problem of 1D-DGAN-PHM is how to construct the synthetic training set. A generative model such as 1D-DGAN-PHM needs to learn the mapping from a signal corrupted by noise to a clean signal based on paired synthetic data since we do not have C-MAPSS clean trajectories. Because physics-of-failure considerations are involved in synthesizing these data, we developed a custom synthetic data generator to model the C-MAPSS data (see Section 3.3). The generator produces a synthetic dataset with pairs of trajectories, each with one clean synthetic trajectory and the same trajectory with noise. We followed three steps to generate these data: (a) elbow point location, (b) nominal stage, and (c) faulty stage computation.

Aero-engines (as well as most engineering systems) typically exhibit two distinct degradation stages over their life course, the nominal stage and the faulty stage, with an inflection point dividing the two regions. Baptista et al. [84] have shown that taking into account this behavior change can significantly impact prognostics performance. When generating the synthetic data of the C-MAPSS turbofan, it is important to simulate the engine's two stages of degradation. In the following subsections, we describe each step of the synthetic data generator.

3.5.1. Data fitting for elbow point

Sensor trajectories of the same unit should display the exact inflection point location since the nominal-to-faulty transition is a property of the component or system. The first step of the synthetic generator is to compute a random elbow point for a given synthetic unit. The elbow or inflection point located on the x-axis is randomly generated according to a density function. The distribution function of the elbow point location is obtained by computing all the inflection points found in the original trajectories (see Fig. 2).

To find the inflection point of a specific unit, we utilize the Kneedle algorithm, proposed by Satopaa et al. [85]. Before applying the Kneedle algorithm, it is necessary to construct an average signal by adding all the features of the unit that exhibit an increasing trend. Before the averaging process, the original signals are normalized. The resulting signal is then smoothed using a Kalman filtering scheme. The Kneedle algorithm [85] is applied to the filtered signal to estimate the knee point of the unit, i.e., the location of maximum curvature in the average projection. This procedure is done for all units.

After obtaining an array of elbow point locations, we fit several distributions to the data with Maximum Likelihood Estimation (MLE), compare the Chi-Squared test of independence, and test for significant difference between observed and fitted distributions with a Kolmogorov–Smirnov test. More details of the fitting are provided in Appendix A.1. Fig. 12 summarizes this entire step.

3.5.2. Data fitting for nominal stage slope

After determining the inflection point location, it is important to define the asset behavior before the elbow point during the nominal stage. The nominal performance consists of the system deteriorating gradually at a slow pace before actual fault initiation. To synthesize the first nominal measurements of a new unit, we synthesize the quasi-linear degradation rate and the noise level. These variables are selected based on density functions generated according to the values observed in the original data.

To obtain the distribution function of the slope, we compute all the degradation rates found in the original trajectories. For this, we apply Least-Squares Regression (LSR) to the first points of each trajectory to fit a straight line to the data.

The linear fit method is illustrated in two examples in Fig. 4. The bold red lines on top of the original signals T24 (Fig. 4(a)) and

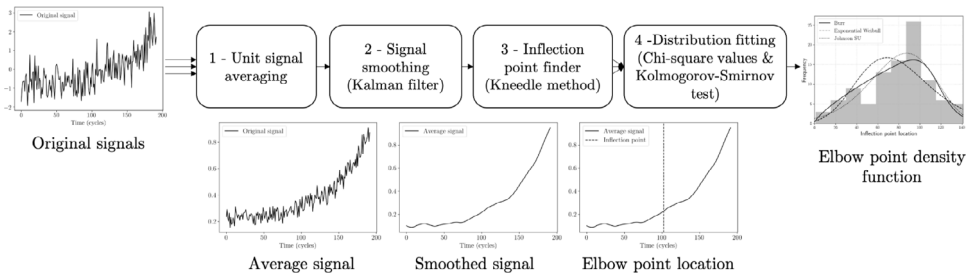


Fig. 2. Generation of the density plot of inflection point location. The process for generating the density function of the inflection point consists of four steps: 1 — Averaging the trajectories of each unit, 2 — Kalman smoothing the average signals, 3 — Kneedle method to estimate the inflection point of each unit and 4 — Fitting distribution functions to the array of inflection points.

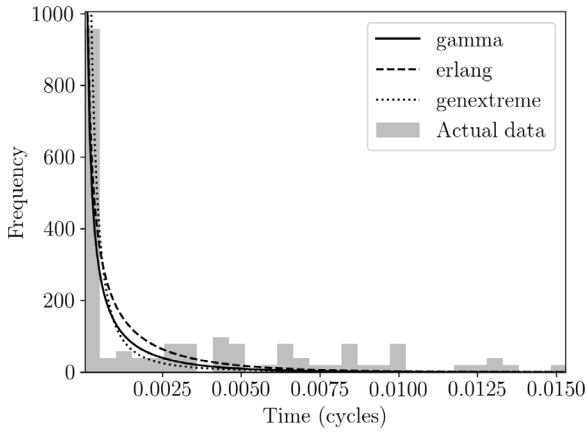


Fig. 3. Histogram and best fitting density functions of slope for signal T24. It is necessary fit a density function to the slope to obtain the underlying quasi-linear degradation, and to capture the nominal behavior of each feature.

W31 (Fig. 4(b)) represent the estimated quasi-linear degradation. As shown, the method works for both increasing and decreasing signals. The slope of the estimated straight line is an important variable to characterize nominal behavior in the synthetic dataset. More details of the fitting are provided in Appendix A.2 (see Fig. 3).

3.5.3. Data fitting for noise level

It is also necessary to compute the noise levels to produce noisy synthetic signals paired with noiseless ones. We assume the noise level is the same throughout the equipment's lifecycle. To calculate the noise level of each unit for a given feature, we apply the $\beta\sigma$ procedure proposed by Czesla et al. [86]. Before calculating the noise level, a translation is performed according to the estimated linear degradation rate to remove undesired effects from the ongoing degradation. Fig. 5 illustrates the translation process with a single unit signal. The translation follows the equation:

$$\bar{y}(n) = y(n) - \text{slope} \times n, \forall n \in \{0, p\} \tag{5}$$

where:

p = elbow point location
n = discrete time indicator
$\bar{y}(n)$ = signal without linear degradation rate
$y(n)$ = noisy signal
p = estimated inflection point
slope = estimated slope of linear degradation rate

As before, we only use the first $p \times \frac{2}{3}$ points of the original trajectories to estimate the noise levels. Note that to estimate the level of noise we apply the noise calculation method to the original noisy signal and not to a smoothed version. The density functions for the slope and noise level are sufficient to simulate signal behavior before inflection point p. More details of the fitting are provided in Appendix A.3.

3.5.4. Data fitting for faulty stage duration

In the fifth step of the synthetic data generator, it is necessary to prescribe the behavior of the asset after the elbow point during the faulty stage. During this stage, the system is degrading at a more accelerated pace. This stage's duration is estimated similarly to the inflection point location with density function sampling. We obtained the distribution of the faulty stage duration using a procedure similar to the one described in Section 3.5.1. More details of the fitting are provided in Appendix A.4.

3.5.5. Trajectory data generation

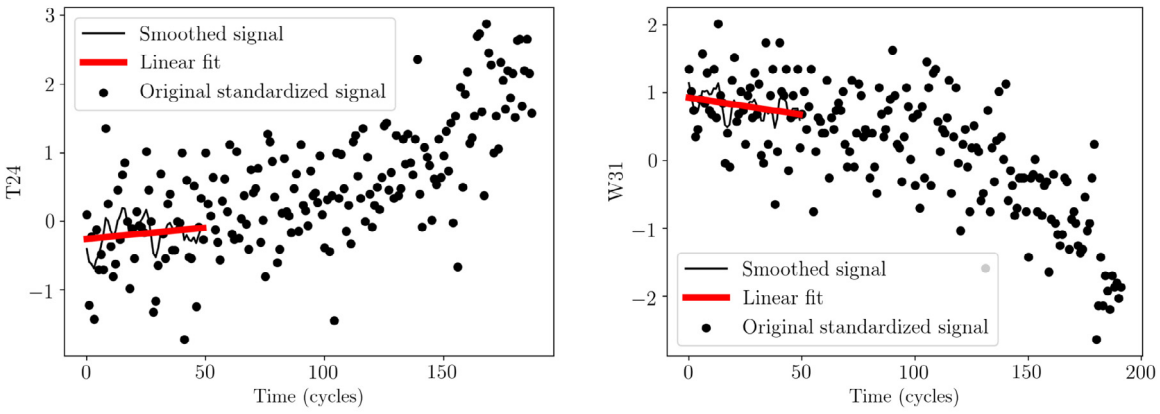
The faulty stage is generated using curve fitting. We generate a new final measurement point for each new trajectory according to the faulty stage duration distribution. Then we fit the following equation with the EoL point and with the final point of quasi-linear degradation:

$$f(x) = a(bx)^c \quad a, b \in \mathbb{R}, \; c \sim U(0, 10) \tag{6}$$

We use a random number (c) to characterize the exponential behavior. The variable c is obtained from an intrinsic Python subroutine that generates a real random number uniformly distributed in [0, 10[. The noise level is used to corrupt the data of the faulty stage signal. At the end of the entire procedure, we have several hundreds of pairs of noiseless/noisy synthetic signals. In Fig. 6 we show some final results of our generation procedure. Figs. 6(a) and 6(b) plot two distinct trajectories of feature T24 synthesized by our generator. The dashed line shows the location of the inflection point that marks the beginning of the faulty stage and the end of the nominal stage. The pure signal curves are generated according to the density functions previously described. The dotted points represent the signal corrupted with noise. Fig. 6(c) shows an example of original T24 data. As can be observed from the three plots of Fig. 6, the synthetic signals closely resemble the original signals. These synthetic data are used to train the GAN. After training the 1D-GAN-PHM network, the network can be used to denoise the (unseen) C-MAPSS data.

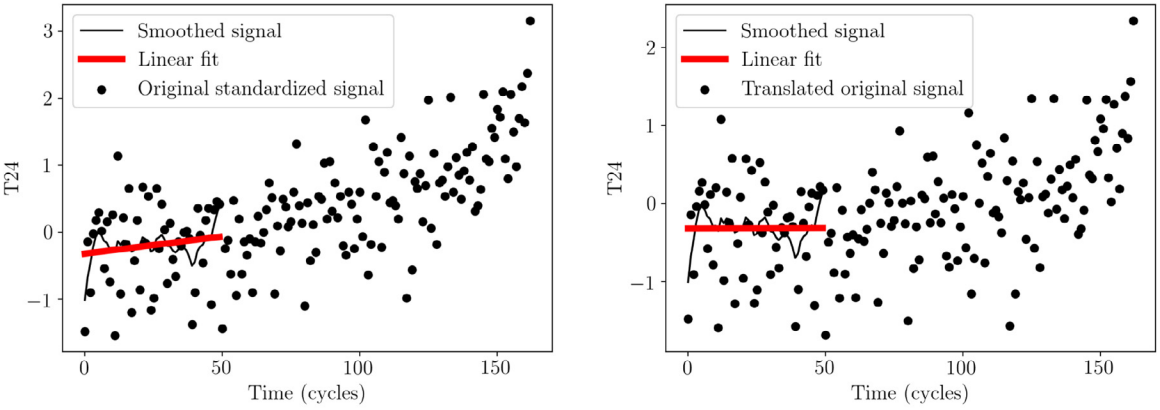
3.6. Denoising Generative Adversarial Network

In this section, we describe the architecture of the 1D-DGAN-PHM. A Generative Adversarial Network (GAN) [12] consists of a generator (or generative) network and a discriminator (discriminative) network. The discriminative network is trained to determine whether a sample is from "real" data or "fake" data generated by the generative network. The generative network is



(a) Example of signal T24 of one unit (b) Example of signal W31 of one unit

Fig. 4. Calculation of quasi-linear trajectories during the nominal degradation stage. To estimate the quasi-linear trajectory of a signal during the nominal stage we apply Linear Squares Regression (LSR) to the first points of a smoothed version of the original signal.



(a) Original signal (b) Translated signal

Fig. 5. Example of noise calculation procedure. It is necessary perform a translation on the data of each unit before applying the $\beta\sigma$ procedure from Czesla et al. [86] onto the first $p \times \frac{2}{3}$ points (p = estimated inflection point) to estimate the noise level of the feature for that unit.

trained to produce realistic examples to deceive the discriminator. During the adversarial training process, the two networks compete so that the generator learns to produce the desired output. The general architecture of the GAN is illustrated in Fig. 7(a).

In this work, we use a deep convolutional GAN similar to the one proposed by Radford et al. [87] (see Fig. 7). Mostly motivated by the work of Radford et al. [87], we made the following changes to the original formulation of Goodfellow et al. [12]:

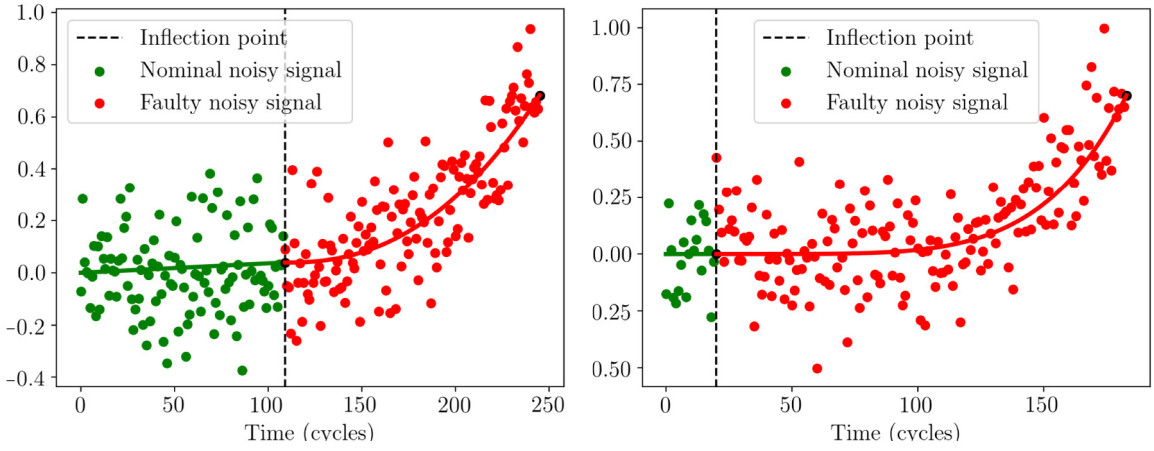
- Pooling layers were replaced by strided convolutions in the discriminator and generator. It has been shown [88] that by replacing max-pooling layers with strided convolution layers, we can enhance network accuracy and reduce model size.
- Batch normalization is used in the generator and discriminator. We use Batch normalization layering to allow every layer in the network to learn more independently and effectively.
- LeakyReLU activation is used for all layers in the discriminator and generator. The advantage of using Leaky ReLU instead of ReLU is that we avoid the vanishing gradient. Both ReLU and LeakyReLU implement nonlinear functions in the network.
- Only one fully connected layer is used as the last layer of the discriminator. A fully connected layer multiplies the input by a weight matrix and then adds a bias vector. This layer typically forms the last layers of convolutional neural networks. We only use one layer for robustness and simplicity.
- In addition to these changes, the proposed generator function has the form of an autoencoder to promote a better denoising function. This architecture is motivated by the work of Makhzani et al. [89].

The two GAN networks consist of strided convolution layers, batch normalization layers, and LeakyReLU activations without max-pooling layers. Fig. 7(b) illustrates the architecture of the generative network. Formally, the objective of the auto-encoding generator is to learn an encoding function f_E and a decoding function f_D such that:

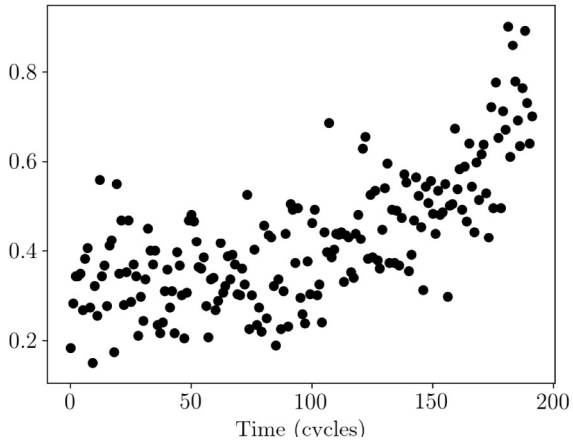
$$\arg\min_{\theta_E,\theta_D}\left(\mathbb{E}_{x\sim p_d}\left(\mathcal{L}(f_D(f_E(x)),x)\right)\right) \tag{7}$$

where:

θ_E = weights and bias of the encoder
θ_D = weights and bias of the decoder



(a) Example #1 of synthetic T24 signal (b) Example #2 of synthetic T24 signal



(c) Example of C-MAPSS T24 original signal

Fig. 6. Example of synthetic trajectories generated with the goal of training the Generative Adversarial Network (GAN). The first two plots are two synthetic trajectories of feature T24 and the last plot is an original randomly selected trajectory of T24.

x = synthetic noisy data sample from distribution p_d
$\mathcal{L}$ = loss function

Fig. 7(c) illustrates the architecture of the discriminator. The discriminator attempts to distinguish between samples from the synthetic database of noiseless images and images generated by the GAN generator. Formally, the discriminator attempts to learn a classifier function f_l that satisfies:

$$\arg\min_{\theta_l}\left(\mathbb{E}_{x\sim p_d}\left(f_l(f_D(f_E(x)))\right)-\mathbb{E}_{z\sim p_z}\left(f_l(z)\right)\right) \tag{8}$$

Where:

θ_l = weights and bias of the discriminator net
x = synthetic noisy data sample from distribution p_d
z = synthetic noiseless data sample from distribution p_z

The functions f_D, f_E, and f_l are learned adversarially. The discriminator learns to distinguish between samples from the two data distributions while the generator attempts to satisfy the following (in addition to Eq. (7)):

$$\arg\min_{\theta_E,\theta_D}-\left(\mathbb{E}_{x\sim p_d}\left(f_l(f_D(f_E(x)))\right)\right) \tag{9}$$

Where:

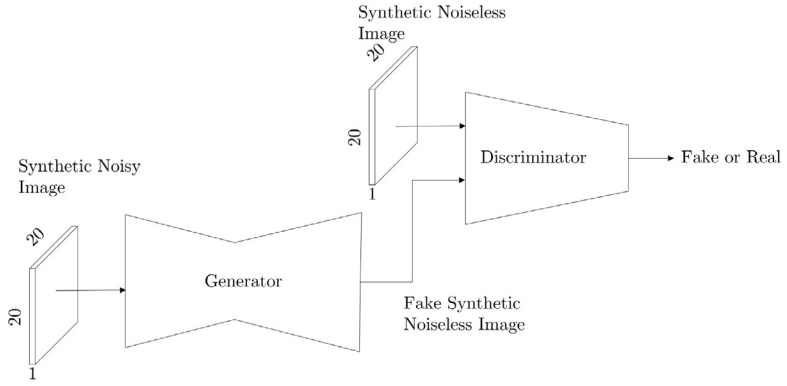
θ_E = weights and bias of the encoder
θ_D = weights and bias of the decoder
x = synthetic noisy data sample from distribution p_d

In our proposed GAN, the 1D-GAN-PHM, the generator is trained with the dual objective of minimizing the standard reconstruction error and the adversarial training loss. At the end of the adversarial learning process, the generator can make data indistinguishable from actual noiseless data to the discriminator. The generator can then be used to denoise unseen noisy data.

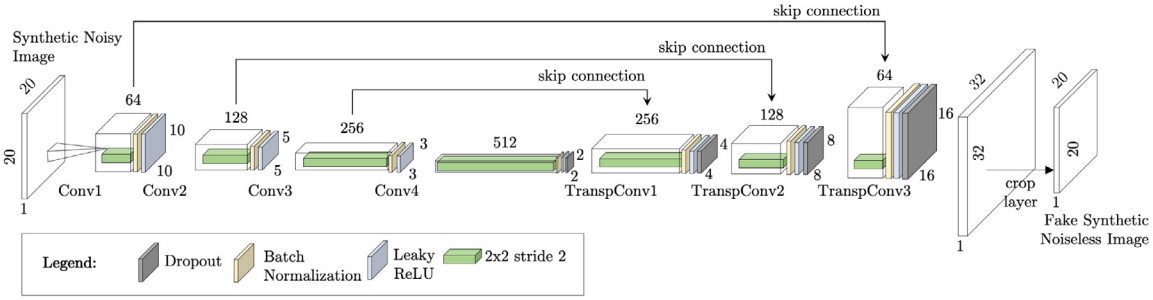
The 1D-GAN-PHM is trained using two different approaches. In the first approach, the Specialized GAN (1D-DSGAN-PHM), we apply a different 1D-DGAN-PHM model to each feature. The Global 1D-DGAN-PHM (1D-DGGAN-PHM) is trained on all data. Fig. 8 shows the two approaches.

3.7. Prognostics evaluation

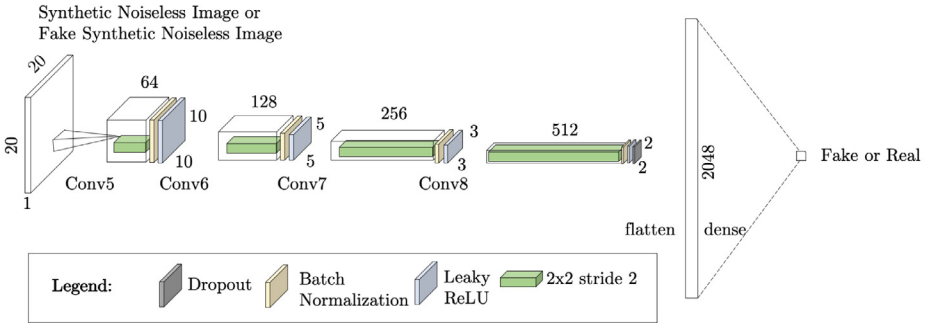
It is important to evaluate the quality of the denoised features but it is also relevant to assess prognostics performance.



(a) Generative adversarial network



(b) Generator network used for denoising. A 20x20 image is projected to a convolutional representation with many feature maps. A series of seven convolution and transpose-convolution layers perform transformations on the input image. No fully connected or pooling layers are used.



(c) Discriminator network used for guiding the training process of the generator network. A 20x20 image is projected to a convolutional representation with several feature maps. A series of four convolution layers perform transformations on the input image. No fully connected or pooling layers are used. A flatten and a dense layer transform the data into the final binary output.

Fig. 7. The architecture of the Generative Adversarial Network (GAN).

The RUL estimation is performed using three different ML models: Multi-Layer Perceptron (MLP), Random Forest (RF), and Support Vector Regressor (SVR). We perform 10-fold cross-validation on the C-MAPSS data denoised by the four selected denoising techniques.

In prognostics, model evaluation is an essential step to establish trust [90] and to better interpret [91] the prognostics. We have implemented five measures of evaluation: Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), PHM'08 Score, True Positive Rate (TNR), and True Negative Rate (TNR). These metrics are described in detail in the work of Saxena et al. [92].

4. Results and discussion

This section presents the experimental results of the paper. We compare denoising methods to investigate the effectiveness of the 1D-DGAN-PHM approach. The comparison is made using the conventional prognostics metrics and the prognostics evaluation measures described in Section 3.7. We test the denoising methods of the Median (Med) filter, the Moving Average (MA), and the Savitzky–Golay (SG) filter. The deep learning denoising approaches include two versions of the Denoising Auto-Encoder (DAE) and two versions of the 1D-DGAN-PHM. We apply a dif-

Table 2
Performance of denoising methods measured by the monotonicity metric. The closer to one is the best monotonicity score. Best performing models are shown in bold and gray. The second-best performing models are shown in light gray.

Sensor	Raw	Classical Denoising			Deep Learning			
		Med	MA	SG	SDAE	GDAE	SDGAN	GDGAN
T24	0.14	0.21	0.30	0.24	0.26	0.56	**0.60**	0.56
T30	0.11	0.20	0.28	0.23	0.25	**0.54**	**0.54**	0.51
T50	0.17	0.29	0.40	0.33	0.36	**0.68**	0.66	0.64
P30	0.17	0.28	0.37	0.32	0.22	**0.65**	0.52	0.62
Nf	0.23	0.26	0.35	0.30	0.32	**0.58**	0.56	0.54
Nc	0.22	0.32	0.39	0.35	0.36	**0.56**	0.46	0.33
Ps30	0.23	0.34	0.45	0.38	0.40	**0.71**	0.67	0.62
phi	0.19	0.31	0.41	0.34	0.37	**0.69**	0.46	0.62
NRf	0.24	0.25	0.35	0.28	0.26	**0.60**	0.53	0.54
NRc	0.25	0.33	0.42	0.37	0.38	**0.59**	0.40	0.35
BPR	0.14	0.25	0.35	0.29	0.30	**0.61**	0.55	0.59
htBleed	0.40	0.21	0.33	0.24	0.25	**0.57**	0.42	0.56
W31	0.16	0.23	0.33	0.26	0.27	**0.61**	0.43	0.60
W32	0.13	0.25	0.34	0.27	0.29	**0.61**	0.49	0.59
Average:	0.20	0.27	0.36	0.30	0.31	**0.61**	0.52	0.55

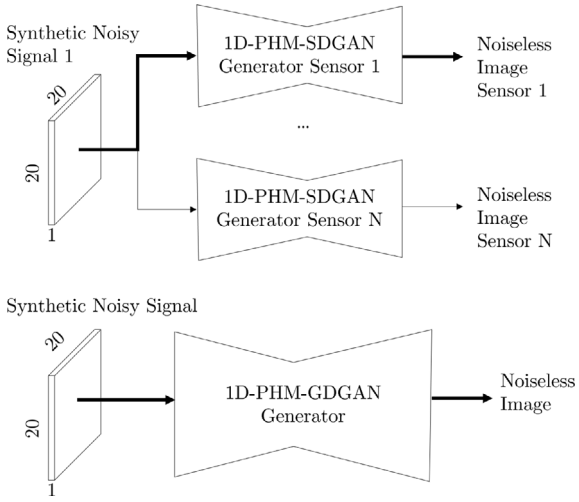


Fig. 8. Data Training Approaches. The 1D-DGAN-PHM model is trained using two approaches: with all the data and one pair of networks to model each type of signal.

ferent network to each feature in the first version of the Specialized GAN (1D-SDGAN-PHM) and Specialized DAE (SDAE). The Global 1D-DGAN-PHM (1D-GDGAN-PHM) is trained on all data. The same for the Global DAE (GDAE).

4.1. Monotonicity

4.1.1. Results

In this section, we evaluate the data before and after denoising. Evaluation is performed using the metrics referred to in Section 3.4. One of these metrics is monotonicity, a fundamental property in prognostics [78–83]. Monotonicity measures the degree to which a population of signals has a consistently increasing or decreasing trend. The closer to one the monotonicity is, the more noticeable the trend is and the more suitable the data is for prognostics.

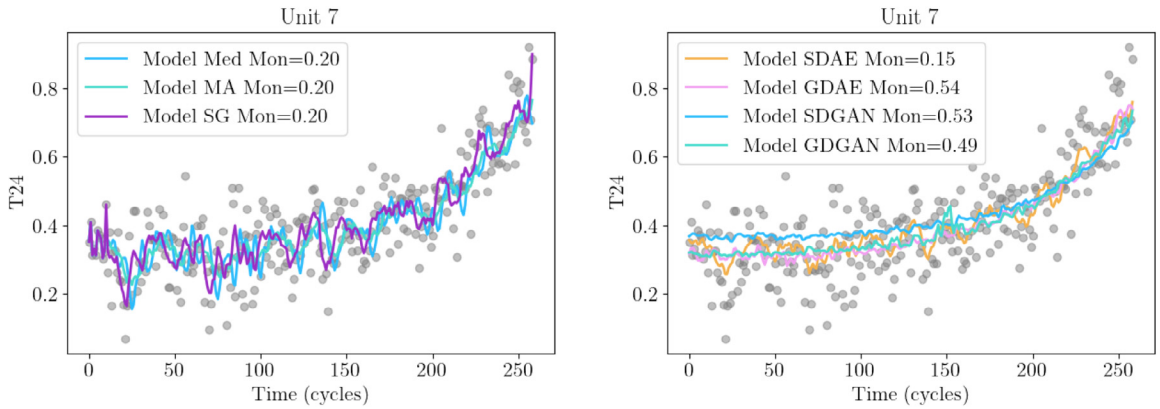
Table 2 shows the significance that denoising can have on the monotonicity of the prognostics signals. The raw signals score 0.20, but the denoising methods can increase the monotonicity to 0.61 (GDAE) on average. The procedures that perform better in monotonicity are the GDAE and the GDGAN models, both being models that explore the entire set of training data (0.61 for GDAE and 0.55 for GDGAN). The SDGAN and the SDAE have acceptable monotonicity scores but are lower than their global versions (0.31 for SDAE and 0.52 for SDGAN). The difference is less noticeable between the DGANs. These results suggest the superiority of the deep learning methods to capture monotonic trends based on the entire dataset. These global methods are exposed to a broader range of patterns during training. The specific networks are tailored to a particular type of sensor. These results seem to point that the diversity of the training data in the global methods is an enabler of monotonicity constraints.

The results also suggest the superiority of the deep learning methods in capturing monotonic trends compared to classical denoising methodologies. Only the MA method has comparable results in monotonicity (average score of 0.36) to deep learning. The MA (score of 0.36) is superior on average to the deep learning SDAE (score of 0.31). The MA is better than the SDAE for all sensors, from T24 to W32. The remaining classical methods have lower average monotonicity scores than the deep learning methods, including the SDAE, the least performing deep learning method.

4.1.2. Discussion

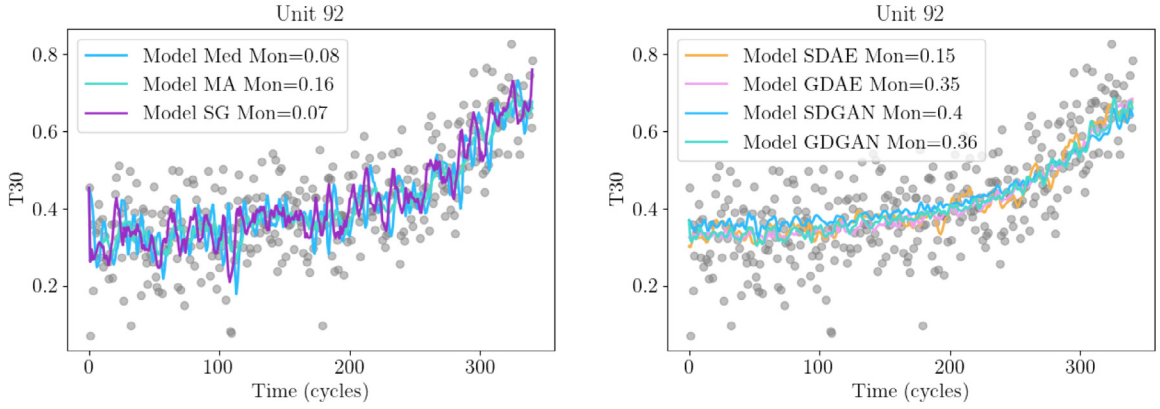
Fig. 9 shows three examples of denoising. On the left side, we show the classical methods, and on the right side, we show the deep learning approaches. As can be seen, classical denoising tends to produce noisier signals that are less monotonic. Also, the classical methods are not edge-preserving and show difficulties in capturing the quasi-linear degradation of the first data points. The signals are more oscillatory and noisy at the beginning than at the remaining trajectory. This can be observed in 9(a), 9(c), and 9(e). The GDGAN shows superior performance in Figs. 9(b) and 9(d). The method, however, performs less well in Fig. 9(f). Only the SDAE has a worse performance (score of 0.19) than the SDGAN. In contrast, the SDAE is consistently the least performer in all examples.

We hypothesize that data plays a major role in the performance of the specific methods, affecting the performance of the global methods less significantly. The more representative the training data, the better the performance of the specific methods. Nevertheless, the DGANs seem to be less affected by the quality



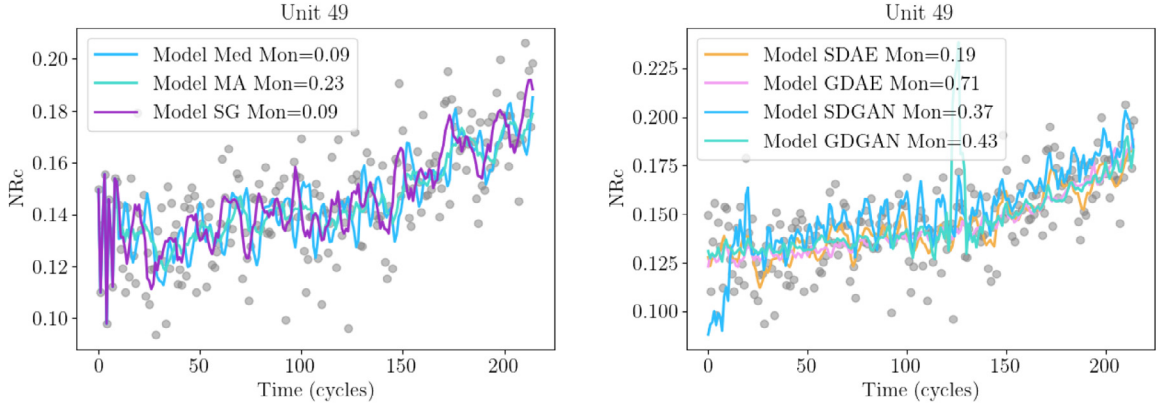
(a) Classical denoising

(b) Deep learning



(c) Classical denoising

(d) Deep learning



(e) Classical denoising

(f) Deep learning

Fig. 9. Visualization of denoising methods for randomly selected engines. On the left side, we show the classical techniques from statistics, and on the right side, the deep learning approaches. The deep learning approaches perform better than the classical denoising methods.

of the training data. The SDAE is more influenced. Overall, the global methods perform better, even though the specific DGAN exhibits better particular results punctually. The SDAE is the least performing method in the deep learning techniques.

The classical methods tend to have worst performance (except for the MA, which outperforms the SDAE) than deep learning. They are not edge-preserving as these methods are non-parametric and do not have the concept of training/learning. They require some points before they start processing the signal adequately. Initially, they do not have sufficient data points and tend to be more oscillatory. The initial edge is critical to the visual appearance of the signal and for the prognostics. For example, the initial points are essential to detect the elbow point location. To prevent this undesired effect, edge-preserving constraints can be formulated while preserving edges for a given, fixed window size. An example of such a method is in [93].

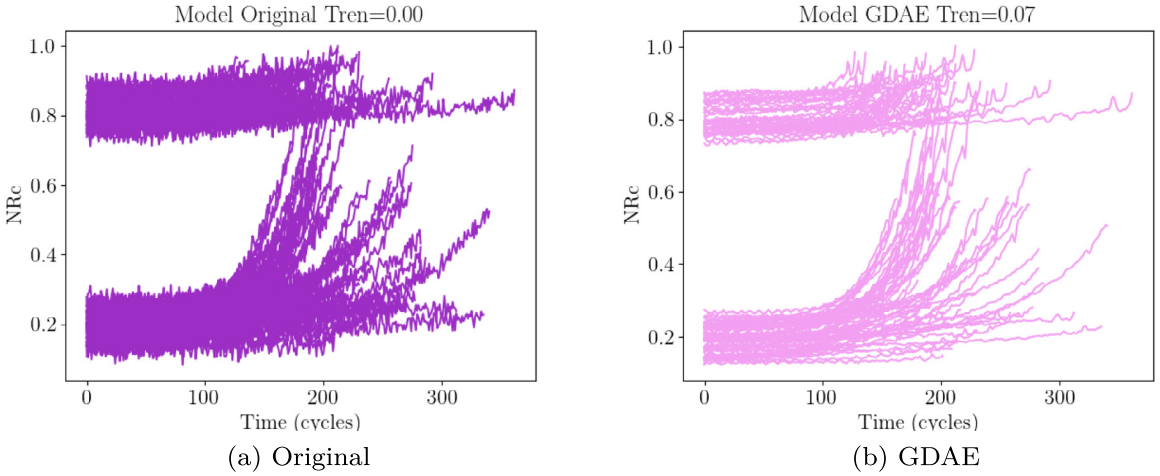


Fig. 10. Visualization of GDAE denoising of NRc. We show on the left side all the raw signals of NRc. On the right side we show the signals after the denoising by GDAE. As shown, the trendability is minimum because the signal can take very different shapes. By definition, denoising removes noise from the signals but does not change the underlying shape of the original signals.

Table 3
Performance of denoising methods measured by the trendability metric. The closer to one is the best monotonicity score. Best performing models are shown in bold and gray. The second-best performing models are shown in light gray.

Sensor	Raw	Classical Denoising			Deep Learning Denoising			
		Med	MA	SG	SDAE	GDAE	SDGAN	GDGAN
T24	0.38	0.57	0.70	0.66	0.69	0.86	**0.92**	0.87
T30	0.27	0.52	0.68	0.60	0.66	**0.84**	0.71	0.81
T50	0.60	0.75	0.82	0.80	0.83	**0.92**	0.91	0.00
P30	0.53	0.69	0.79	0.78	0.63	**0.93**	0.91	0.66
Nf	0.04	0.04	0.15	0.23	0.31	0.62	**0.68**	0.25
Nc	0.00	0.00	**0.01**	0.00	0.00	**0.01**	0.00	0.00
Ps30	0.67	0.82	0.89	0.84	0.88	**0.96**	0.95	0.00
phi	0.54	0.73	0.83	0.79	0.82	**0.95**	0.85	0.01
NRf	0.04	0.00	0.20	0.14	0.08	0.57	**0.63**	0.51
NRc	0.00	0.00	0.00	0.00	0.00	**0.07**	0.00	0.00
BPR	0.48	0.69	0.75	0.75	0.78	0.93	0.58	**0.94**
htBleed	0.33	0.49	0.55	0.58	0.60	0.90	0.90	**0.93**
W31	0.46	0.66	0.78	0.72	0.72	0.93	0.86	**0.94**
W32	0.47	0.70	0.79	0.73	0.78	**0.92**	**0.92**	**0.92**
Average:	0.34	0.48	0.57	0.54	0.56	**0.74**	0.70	0.49

4.2. Trendability

4.2.1. Results

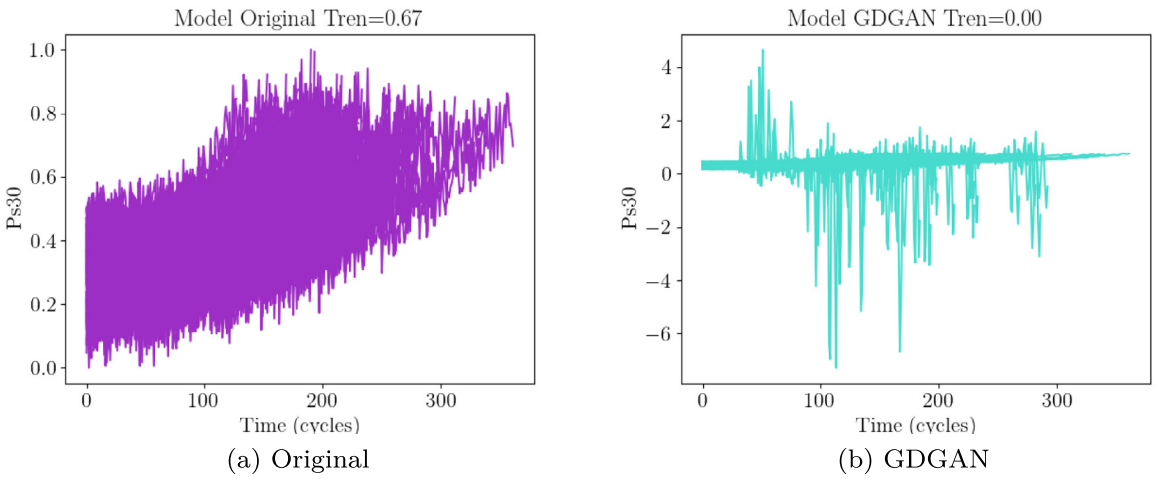
Regarding the metric of trendability, the results are shown in Table 3. Trendability is also a fundamental property in prognostics [78–83]. This measure computes the degree to which a population of signals exhibits the same functional shape. It is theoretically more straightforward for a prognostics model to capture degradation trends if different units show similar trajectories. Table 3 characterizes the impact denoising can have on the trendability of monitoring signals. The original signals have a trendability score between 0 and 0.67, with the different denoising methods significantly impacting these scores.

The best performing model is the GDAE (average score of 0.74), followed closely by the SDGAN (average score of 0.70). The SDGAN has nine signals in the top 2 performance while GDAE has 12. The GDAN also performs well, with six signals in the top 2. The SDGAN and the GDAN have zero trendability (2 sensors for SDGAN and 4 for GDGAN). This follows from the Pearson correlation used to measure the trendability between pairs of trajectories. The metric selects the absolute minimum correlation for each sensor. Zero trendability means that at least one pair of trajectories were very dissimilar.

4.2.2. Discussion

The denoising methods, in general, tend to increase the trendability of the features significantly. However, the trendability score is left almost unchanged for the sensors with zero trendability, Nc and NRc. This low trendability results from the existence of one or more abnormal units with a very distinct functional shape from the remaining signals in the original data. In Fig. 10 we show all the trajectories captured by sensor NRc. We show the trajectories before (Fig. 10(a)) and after denoising (Fig. 10(b)). The denoising has a minor effect on the trendability of the trajectories (score 0.00 to 0.07). By definition, denoising removes noise from signals, but denoising does not aim to change the original signals' underlying shape. The low trendability is hence due to the original shape of the signals and not the denoising.

Trendability can also signify that some trajectory was not correctly denoised, as illustrated in Fig. 11. In the Figure, the



(a) Original (b) GDGAN

Fig. 11. Visualization of GDGAN denoising of Ps30. We show on the left side all the raw signals of Ps30. On the right side we show all the Ps30 signals denoised by GDGAN. As shown, the trendability is minimum because the denoising produces erroneous and dissimilar signals to the original ones.

Table 4
Performance of denoising methods measured by the prognosability metric. The closer to one is the best monotonicity score. Best performing models are shown in bold and gray. The second-best performing models are shown in light gray.

Sensor	Raw	Classical Denoising			Deep Learning Denoising			
		Med	MA	SG	SDAE	GDAE	SDGAN	GDGAN
T24	0.81	0.89	0.92	0.83	0.90	0.91	**0.93**	0.91
T30	0.76	0.83	0.87	0.79	0.86	0.85	**0.89**	0.86
T50	0.87	0.91	**0.94**	0.88	0.93	0.93	0.93	0.59
P30	0.84	0.88	**0.90**	0.85	0.85	0.88	0.89	0.75
Nf	0.66	0.71	0.73	0.67	0.74	0.74	**0.76**	0.74
Nc	0.29	0.27	0.27	0.30	**0.38**	0.29	**0.38**	0.12
Ps30	0.87	0.92	0.94	0.89	0.92	**0.93**	0.92	0.30
phi	0.85	0.87	**0.88**	0.85	0.87	0.87	0.87	0.30
NRf	0.66	0.71	0.73	0.67	0.73	0.73	**0.74**	0.73
NRc	0.33	0.29	0.29	0.32	0.36	0.31	**0.48**	0.11
BPR	0.84	0.89	0.93	0.86	0.91	0.92	0.92	**0.93**
htBleed	0.79	0.88	**0.90**	0.80	0.86	0.88	0.87	0.89
W31	0.83	0.90	**0.93**	0.84	0.88	0.90	0.89	0.91
W32	0.81	0.89	**0.93**	0.84	0.89	0.91	0.90	0.91
Average:	0.73	0.77	0.80	0.74	0.79	0.79	**0.81**	0.65

trendability is low because the GDAN produces signals with very different shapes. The Global DGAN tends to have this erroneous behavior more often than the other methods. The specific DGAN appears to be more protected against this type of effect. In this case, the best option might be partitioning the signals into two (or more) sets of features and running the models separately.

4.3. Prognosability

4.3.1. Results

The results of the prognosability analysis are shown in Table 4. Prognosability is important as it measures the degree of variability of the condition indicators at failure. The influence of denoising in this respect is significant even though not as expressive as for the other two metrics. The best method, the SDGAN, obtains an increase of 9% going from 0.73 to 0.81 in average score. The SDGAN is the best method in six features. The second best method, the MA, obtains a similar performance to the SDGAN with a score of 0.80. The prognosability of the original signals is already high, with the denoising resulting in a slight improvement. This is understandable as removing noise from prognostics trajectories also removes noise at the end of life, resulting in less variability. However, since denoising does not aim to change the actual shape of the signals, prognosability only shows a moderate improvement. In addition, it is more challenging to address noise at the beginning of a trajectory than at the end, given the exponential shape of the trajectories.

4.3.2. Discussion

Overall, the tested denoising methods had comparable performance according to the prognosability metric, as seen in the last row of Table 4. This can be observed more clearly in Fig. 12. Here, the SDGAN and MA histograms of end-of-life measurements are not significantly different for the T24 sensor. The SDGAN and MA were the best prognosable methods. The density of the final measurements in the original data is flatter and sparser. This result suggests that denoising is also vital for prognosability and that deep learning (SDGAN) can produce results as good or better than classical denoising. The classical and deep learning methods have similar results regarding best performance.

Table 5
Performance of different denoising methods measured with prognostics metrics. Best performing models are shown in bold and gray. The second-best performing models are shown in light gray.

	Data	MAE	RMSE	TPR	TNR	Score
RF	Raw	28.52 ± 5.46	39.20 ± 8.14	0.39 ± 0.06	0.32 ± 0.07	5.97E+03
	Med	28.29 ± 6.97	39.01 ± 10.11	0.40 ± 0.06	0.33 ± 0.08	8.13E+03
	MA	28.03 ± 7.54	38.80 ± 10.75	0.40 ± 0.09	0.34 ± 0.09	1.09E+04
	SG	26.91 ± 6.93	37.48 ± 10.15	0.41 ± 0.06	0.35 ± 0.08	**5.23E+03**
	SDAE	26.96 ± 6.97	37.83 ± 10.31	0.43 ± 0.07	0.34 ± 0.09	7.78E+03
	GDAE	28.95 ± 8.74	41.37 ± 12.97	0.47 ± 0.10	0.31 ± 0.08	1.39E+05
	SDGAN	**25.46 ± 7.71**	**36.30 ± 11.44**	**0.47 ± 0.09**	**0.36 ± 0.10**	1.29E+04
	GDGAN	27.51 ± 7.20	38.53 ± 10.93	0.45 ± 0.09	0.31 ± 0.08	5.78E+04
MLP	Raw	31.25 ± 4.84	40.96 ± 7.81	0.28 ± 0.05	0.25 ± 0.06	7.05E+03
	Med	31.80 ± 5.22	41.56 ± 8.18	0.28 ± 0.06	0.24 ± 0.06	1.28E+04
	MA	30.49 ± 5.54	40.38 ± 8.20	0.31 ± 0.10	0.27 ± 0.07	3.68E+04
	SG	30.58 ± 4.93	40.22 ± 7.44	0.30 ± 0.09	0.26 ± 0.07	7.77E+03
	SDAE	30.30 ± 5.13	39.84 ± 7.97	0.30 ± 0.07	0.26 ± 0.06	4.57E+03
	GDAE	30.02 ± 5.54	39.29 ± 8.05	0.31 ± 0.11	0.25 ± 0.07	**4.20E+03**
	SDGAN	**28.66 ± 5.90**	**37.86 ± 8.93**	0.31 ± 0.10	**0.29 ± 0.08**	1.41E+04
	GDGAN	29.51 ± 5.28	39.06 ± 7.69	**0.33 ± 0.08**	0.25 ± 0.05	2.51E+12
SVR	Raw	33.40 ± 4.66	44.74 ± 8.74	0.30 ± 0.06	0.21 ± 0.03	2.31E+04
	Med	32.98 ± 4.84	44.29 ± 8.92	0.31 ± 0.06	0.22 ± 0.03	2.58E+04
	MA	32.63 ± 4.85	43.83 ± 8.96	0.30 ± 0.07	0.22 ± 0.03	2.54E+04
	SG	32.07 ± 4.79	43.23 ± 8.96	0.32 ± 0.07	0.22 ± 0.02	2.26E+04
	SDAE	32.00 ± 4.71	**43.12 ± 8.91**	0.32 ± 0.07	0.22 ± 0.03	**1.77E+04**
	GDAE	32.23 ± 4.79	43.53 ± 9.07	0.32 ± 0.07	0.22 ± 0.03	2.23E+04
	SDGAN	**31.83 ± 4.86**	43.28 ± 9.29	**0.34 ± 0.07**	**0.23 ± 0.04**	2.74E+04
	GDGAN	33.63 ± 5.02	45.62 ± 9.58	0.32 ± 0.07	0.22 ± 0.03	2.94E+11

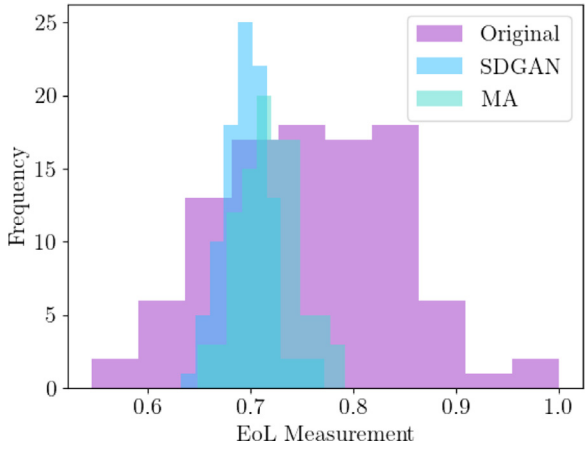


Fig. 12. Visualization of the histogram of prognosability for T24. We show the density of the end-of-life measurements of the two best methods and the raw data.

4.4. Prognostics performance

4.4.1. Results

It is important to evaluate the prognostics performance of the denoised trajectories since remaining useful life (RUL) estimation is generally the ultimate goal of prognostics. Table 5 presents the results of applying the denoised data to predict failure in the C-MAPSS case. We test three machine learning approaches: Random Forest (RF), Support Vector Regressor (SVR), and Multi-Layer Perceptron (MLP).

The Random Forest (RF) performs better after denoising for all metrics including the Scoring metric. The Scoring metric was used during the original PHM'08 challenge to evaluate the performance of the different algorithms. It was devised specifically for the C-MAPSS prognostics case and aimed to penalize late predictions over early ones. It is an important metric to consider when measuring a model's performance. The denoising models, except for the SG, did not produce as good results as the original dataset regarding the Scoring metric. Probably this resulted from the denoising methods having produced points with marked deviations from the expected behavior. Such points can lead to high scores. Because the Scoring metric penalizes errors exponentially, this could explain the high score values.

In all the other dimensions, the SDGAN is the best performing method for the RF. In regards to deep learning, the best models are in this order ($\succ$ represents better performance and $\sim$ represents similarity):
MAE, RMSE: SDGAN $\succ$ SDAE $\succ$ GDGAN $\succ$ GDAE
TPR: SDGAN $\sim$ GDAE $\succ$ SDAE $\succ$ GDGAN
TNR: SDGAN $\succ$ SDAE $\succ$ GDGAN $\sim$ GDAE

The SDAE model also has acceptable performance, coming second in the best-performing deep learning list in almost all metrics. This result suggests that specific deep learning, where a network is tailored for each data type, is more suitable for prognostics. The global methods GDAE and GDGAN usually come last in the different metrics and have significant PHM'08 scores. A positive remark is the True Positive Rate (TPR) of the GDAE and GDGAN. The high scores at this metric suggest that the Global models have a more significant proportion of "tolerable" late predictions than the other methods. An important consideration is to have over-predictions that do not surpass the limits of late estimation.

The SG filter has the second-best performance of all methods using the RF. This result suggests that simple denoising techniques can be almost as effective as deep learning methods when the RUL estimation is considerably sophisticated. Interestingly, the SG filter did not score significantly better than the other classical methods in the metrics of monotonicity (average score of 0.31), trendability (average score of 0.54), and prognosability

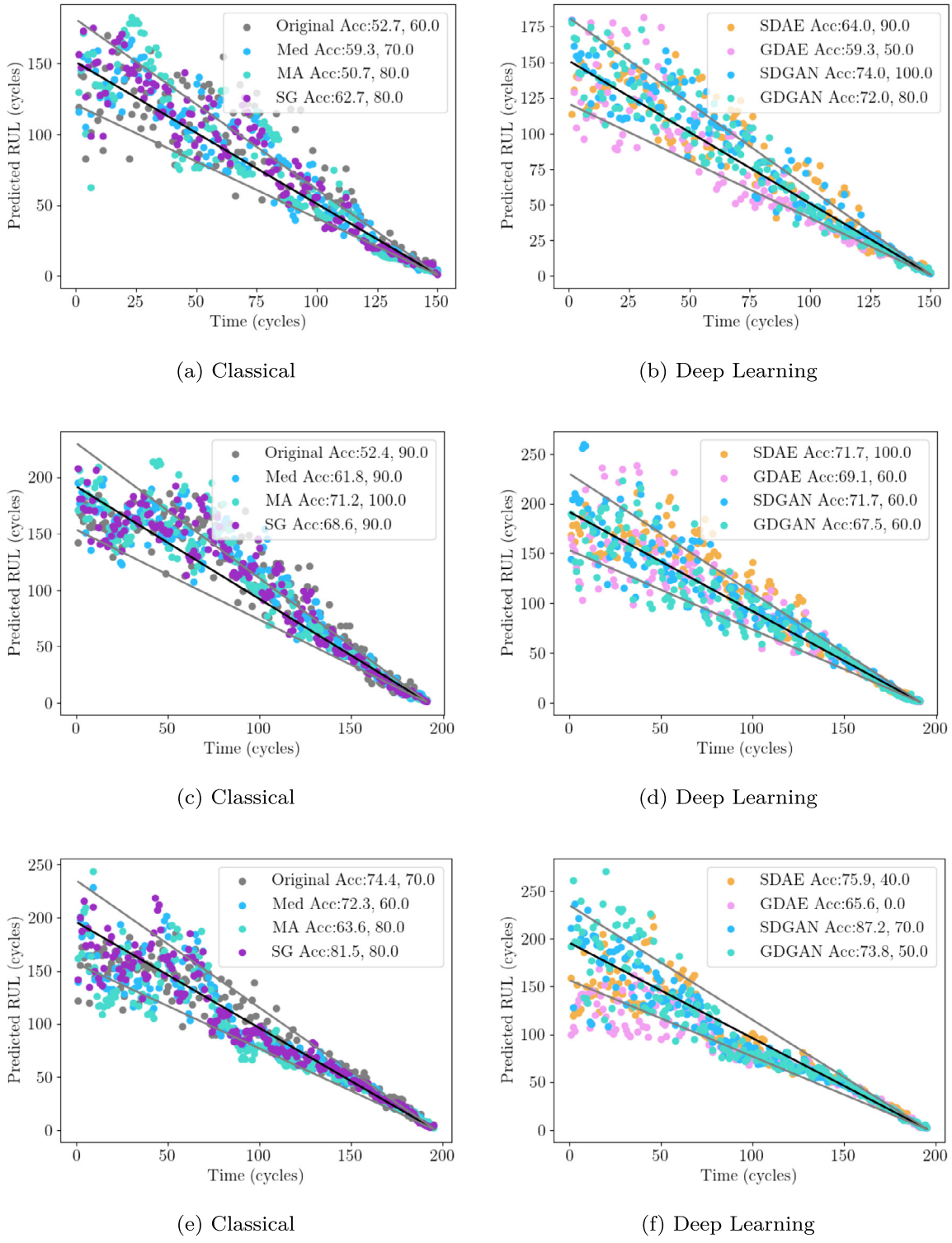
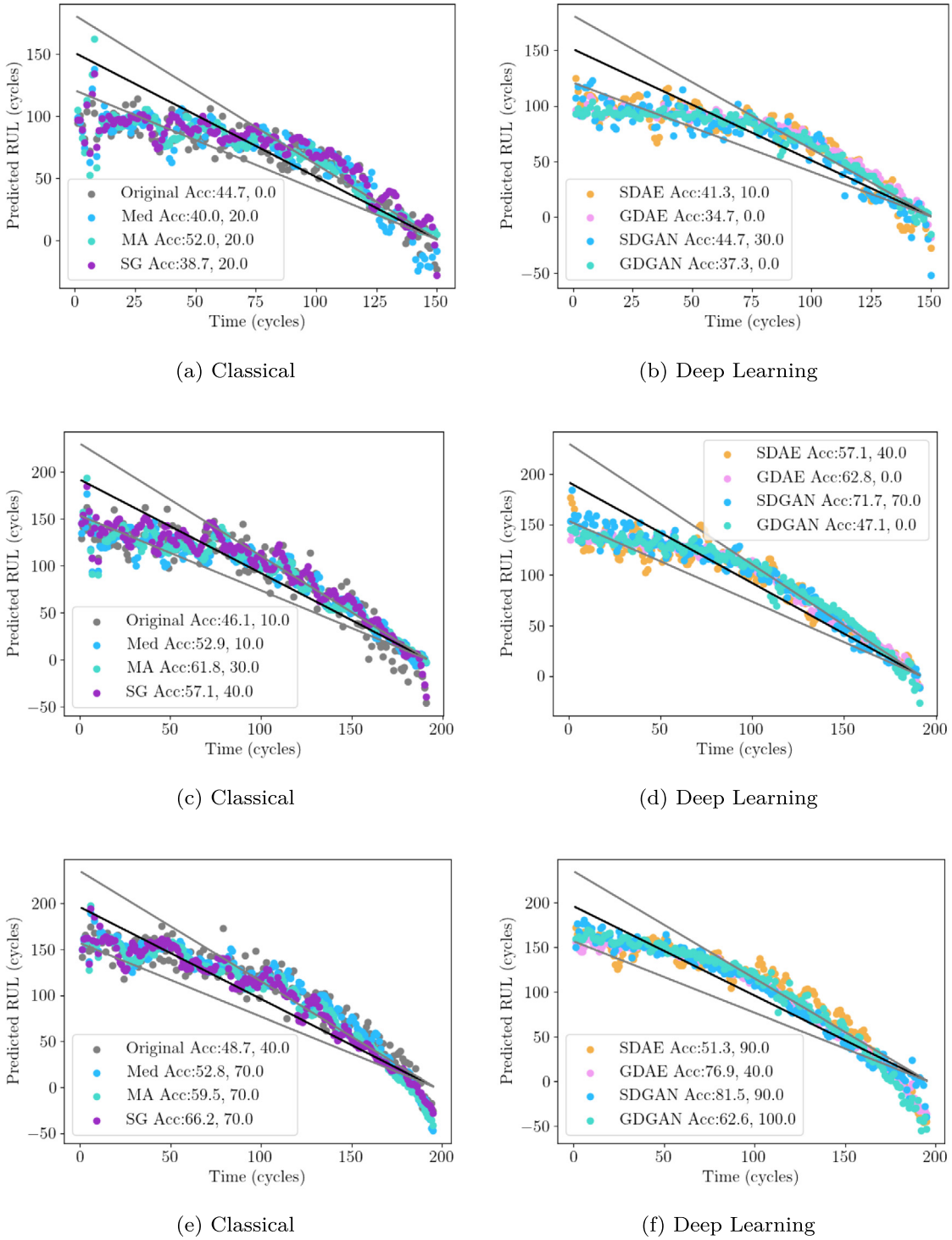


Fig. 13. Visualization of RUL Estimation for the Random Forest (RF) model. We show the *alpha*–λ accuracy on average for the unit and at the start of the unit.

(average score of 0.74). This suggests that the quality of a denoised predictor cannot be based solely on the classical metrics of feature quality.

The Multi-Layer Perceptron (MLP) performs better in all metrics after denoising. In contrast with the RF, denoising with deep learning DAE models promotes better PHM'08 score values with the MLP. In this respect, the best-performing models in the PHM'08 score are the SDAE and GDAE. In all other dimensions, except the TPR dimension, the SDGAN is the best-performing model, suggesting the superiority of this denoising for prognostics. In regards to deep learning, the best models are in this order ($\succ$ represents better performance and $\sim$ represents similarity):

MAE: SDGAN $\succ$ GDGAN $\succ$ GDAE $\succ$ SDAE
RMSE: SDGAN $\succ$ GDGAN $\succ$ SDAE $\succ$ GDAE
TPR: GDGAN $\succ$ SDGAN $\sim$ GDAE $\succ$ SDAE
TNR: SDGAN $\succ$ SDAE $\succ$ GDGAN $\sim$ GDAE
Score: GDAE $\succ$ SDAE $\succ$ SDGAN $\sim$ GDGAN



(a) Classical (b) Deep Learning

(c) Classical (d) Deep Learning

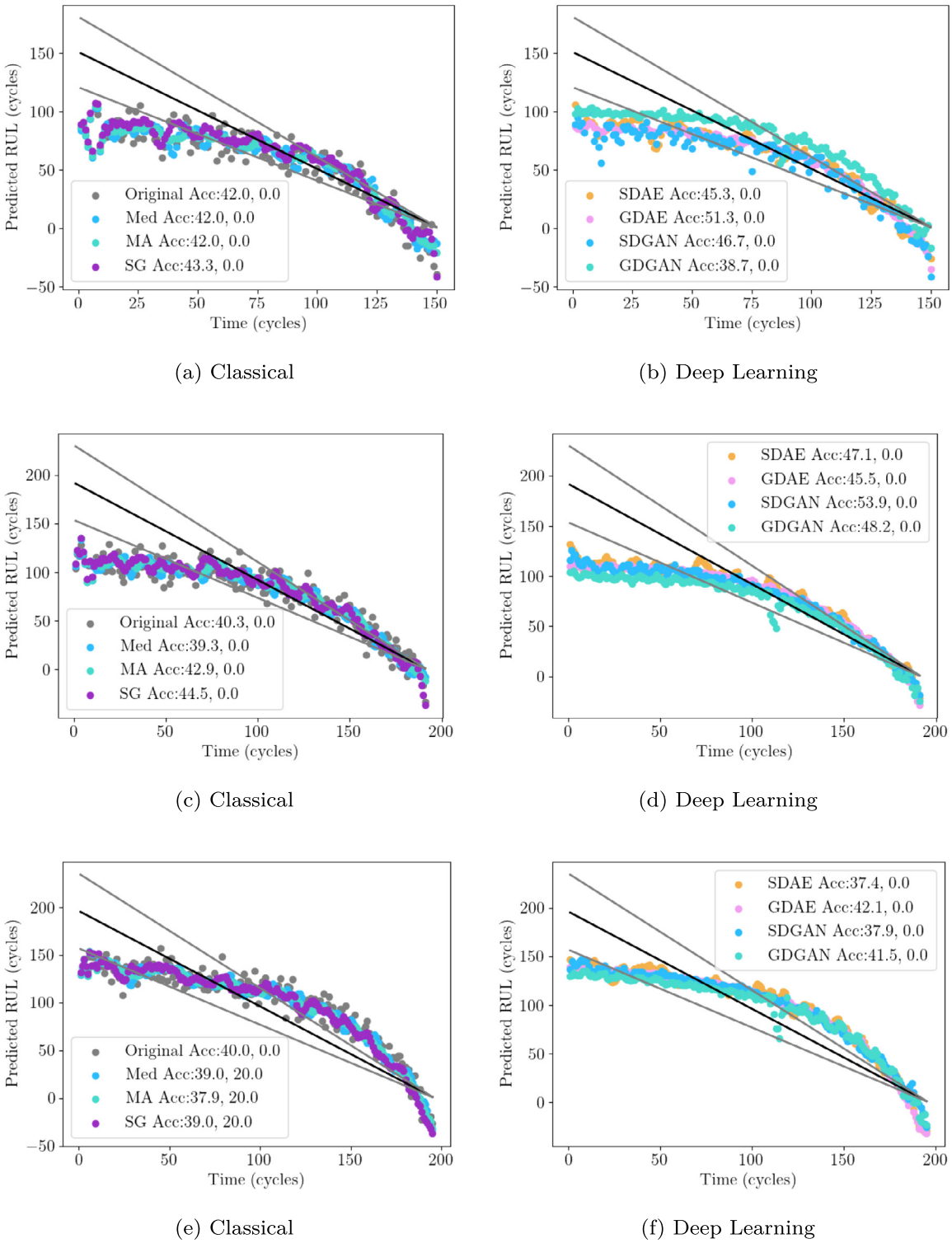
(e) Classical (f) Deep Learning

Fig. 14. Visualization of RUL Estimation for the Multi-Layer Perceptron (MLP) model. We show the *alpha*–λ accuracy on average for the unit and at the start of the unit.

The Support Vector Regression (SVR) performs better in all metrics after denoising. The results suggest the SDGAN is the best 1D-DGAN-PHM method, a model robust to changes in the prognostics model. The denoising of the SDGAN works best for more complex models, such as the RF and MLP, but can also be used with simpler models.

4.4.2. Discussion

The inaccuracies shown by the Median (Med), Moving Average (MA), and Savitzky and Golay (SG) filter during the first cycles can be detrimental when elbow point detection methods are used to improve the prognostics [84]. The deep learning methods appear to be potential approaches that can preserve the initial linear degradation patterns. GAN and AE models tend to produce trajectories that exhibit initial linear behavior in line with the overall trend of the signals. This result is a positive aspect of the deep denoising models in contrast to the classical methods. The lack of denoising accuracy at the first data points is often translated into a lack of prognostics accuracy in the first data points. For example, consider Figs. 14(a) and 14(b) where we



(a) Classical (b) Deep Learning

(c) Classical (d) Deep Learning

(e) Classical (f) Deep Learning

Fig. 15. Visualization of RUL Estimation for the Support Vector Regression (SVR) model. We show the *alpha*–λ accuracy on average for the unit and at the start of the unit.

show a randomly selected engine unit and the RUL estimates. The first points of Fig. 14(a) oscillate more than the remaining points, and the $\alpha - \lambda$ accuracy at the start ranges between 0 and 20%. In Fig. 14(b) we can observe a less oscillatory behavior at the start of the equipment life. The SDGAN, in particular can reach an $\alpha - \lambda$ accuracy at the start of 30%. This means that 30% of the predictions fall inside the cone of accuracy of $\alpha = 0.2$. This ability to preserve edges is a positive aspect of the deep denoising models in contrast with the classical methods.

Overall, the deep learning methods are more accurate as can be observed in the $\alpha - \lambda$ plots of Figs. 13 to 15. The RF and the MLP models produce acceptable predictions aided by the deep learning methods, as shown in Figs. 13 to 14. In the RF examples (Fig. 13), the $\alpha - \lambda$ accuracy of the unit is the highest for the

DSGAN, attesting again to the superiority of this method. The second best method varies according to the unit. It would be interesting to analyze fusion methods for the denoised trajectories and their impact on prognostics accuracy. In the MLP examples (Fig. 14), the accuracy is lower probably because of the method's lesser sophistication. The SDGAN continues to score the highest in $\alpha - \lambda$ average accuracy compared to the other methods. In the SVR examples (Fig. 15), the SDGAN only reaches the top score in one example. This result can be explained by the inability of the model to capture relevant patterns from the denoised data. Each model appears to react differently to each type of denoising, even though the best denoising method seems to operate better with the most sophisticated methods. Given these results we find enough evidence to support our initial research hypothesis:

H: Generative adversarial networks (GAN) can outperform the performance of baseline methods for denoising PHM data.

5. Conclusion

In this work, Generative Adversarial Networks (GANs) are proposed as a denoising approach suitable to remove noise and outliers from condition monitoring signals in Prognostics and Health Management (PHM). We validate our 1D-DGAN-PHM approach using the C-MAPSS case study. The 1D-DGAN-PHM network is compared against the classical methods of Median, Moving Average, denoising auto-encoder and Savitzky and Golay filters.

The deep learning approaches, the autoencoders and the 1D-DGAN-PHM models, appear to be better suited to preserve the initial quasi-linear degradation than the classical denoising methods. The prognostics of the denoised trajectories with the 1D-SDGAN-PHM (specific version) was superior in most evaluation measures. The model achieved the best prognosability (score of 0.81). The positive effects of the SDGAN model in several evaluation dimensions (MAE, RMSE, TPR, TNR, and $\alpha - \lambda$ accuracy) encourage further investigation of these models in other domains. The 1D-DGAN-PHM model can be used in different prognostics domains, from energy to aerospace. The only requirement is the existence of a synthetic generator of pure and noisy signal pairs.

The generality of the DGAN models depends on the quality of the training set. We have shown that tailoring a DGAN for a specific set of signals promotes a better denoising than exposing the network to all condition monitoring data. Through exposure to tailored and specific data, the deep generator can be more robust and produce better prognostics. For future work, it would be interesting to test alternative denoising strategies, transfer learning, and generation of training data.

Note: The 1D-DGAN-PHM project is open source, and the code is publicly available at **1D-DGAN-PHM**. The project is a stand-alone desktop application written in Python3.7.

CRediT authorship contribution statement

Marcia L. Baptista: Conceptualization, Methodology, Software, Writing – original draft, Data curation, Investigation, Validation, Writing – review & editing. **Elsa M.P. Henriques:** Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data is publicly available by NASA Research Ames.

Table A.6
Best distributions for fitting inflection point data. Goodness of fit and results of Kolmogorov–Smirnov test are presented. The best distribution for the specific application is the Burr distribution.

Distribution	Chi-square	*p*-value
Burr	16.12	0.65 (>0.05)
Exponential Weibull	19.86	0.51 (>0.05)
Johnson SU	20.11	0.91 (>0.05)
Johnson SB	20.13	0.62 (>0.05)
Log Gamma	20.26	0.91 (>0.05)
Generalized Extreme Value	20.90	0.71 (>0.05)
Frechet	20.90	0.71 (>0.05)
Weibull Max	20.90	0.71 (>0.05)
Beta	21.53	0.55 (>0.05)
Gumbel	27.34	0.57 (>0.05)

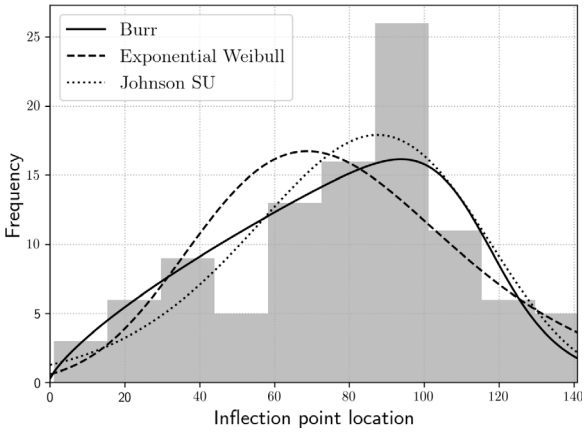


Fig. A.16. Density plots and histogram of inflection point location. We compared several distributions in their goodness of fit to our data and the Burr distribution presented the best results (lowest chi-square value and Kolmogorov–Smirnov *p*-value > 0.05).

Appendix. Data fitting (synthetic data generator)

A.1. Elbow point

In our case study, we tested several of the 81 continuous density functions available at the Python SciPy library. We performed goodness-of-fit tests on the 81 distributions to see which one was the best fit for the elbow point data in dataset C-MAPSS (FD001). In Table A.6 we present the ten best results (according to the Chi-Square test) of the data fit. We also present in Table A.7 the parameters of the best fitting distributions. The Burr, Exponential Weibull and Johnson SU distributions gave the best fitting results (see Fig. A.16). As shown in the histogram, the data reaches its peak at ≈100 cycles and the maximum inflection point is reached at 140 cycles, at sensibly 40% of the maximum equipment life (362 cycles). We present the probability density function of the three distributions with the best fit (Burr, Exponential Weibull and Johnson SU):

$$f(x, c, d) = cdx^{-c-1} / \left(1 + x^{-c}\right)^{d+1} \tag{A.1}$$

$$f(x, a, c) = ac \left[1 - \exp\left(-x^{c}\right)\right]^{a-1} \exp\left(-x^{c}\right) x^{c-1} \tag{A.2}$$

$$f(x, a, b) = \frac{b}{\sqrt{x^2 + 1}} \phi\left(a + b \log\left(x + \sqrt{x^2 + 1}\right)\right) \tag{A.3}$$

To shift and/or scale the distributions use the location and scale parameters. Specifically, with the transformation x' = (x - loc) / scale.

Table A.7
Parameters of best fit distributions for inflection point data. The formal equation for each Probability Density Function (PDF) is referenced in the table.

Distribution	PDF	Parameters			
		1	2	3 (Location)	4 (Scale)
Burr	Eq. (A.1)	$c = 12.43$	$d = 0.14$	−0.46	118.90
Exp. Weibull	Eq. (A.3)	$a = 680.88$	$c = 5.87$	−1174.96	900.43
Johnson SU	Eq. (A.3)	$a = 10.83$	$b = 5.49$	250.51	47.74

Table A.8
Best distributions for fitting data of the duration of faulty stage. Goodness of fit and results of Kolmogorov–Smirnov test are presented. The best distribution for the specific application is the Inverse Gamma distribution.

Distribution	Chi-square	p-value
Inverse Gamma	9.08	0.87 (>0.05)
Inverse Gaussian	9.28	0.96 (>0.05)
Log Normal	9.28	0.93 (>0.05)
Johnson SU	9.29	0.93 (>0.05)
Inverse Weibull	9.57	0.81 (>0.05)
NCT	9.68	0.79 (>0.05)
Fatigue Life	9.85	0.97 (>0.05)
Recip. Inv. Gaussian	10.59	0.97 (>0.05)
Alpha	12.21	0.79 (>0.05)
Exponential Weibull	13.89	0.93 (>0.05)

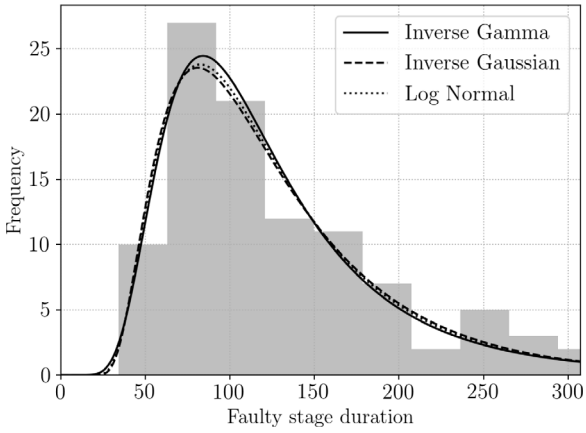


Fig. A.17. Density plots and histogram of faulty stage duration. We compared several distributions in their goodness of fit to our data with the Inverse Gamma distribution presenting the best results (lowest chi-square value and Kolmogorov–Smirnov p-value > 0.05).

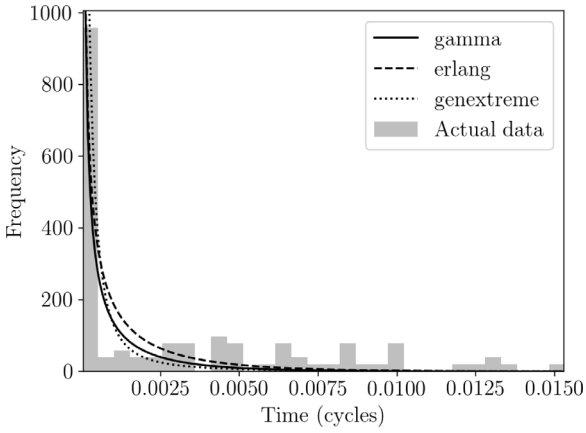


Fig. A.18. Histogram and best fitting density functions of slope for signal T24. It is necessary fit a density function to the slope to obtain the underlying quasi-linear degradation, and to capture the nominal behavior of each feature.

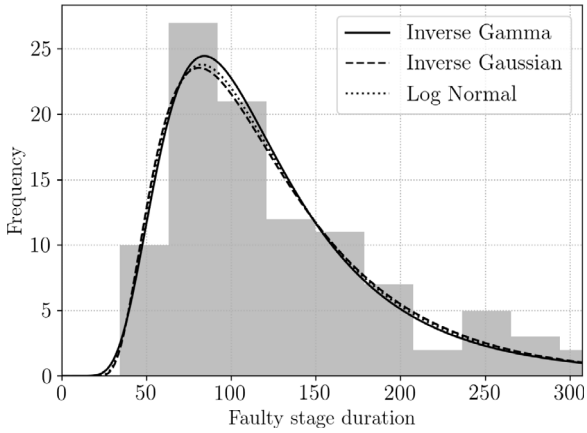


Fig. A.19. Density plots and histogram of faulty stage duration. We compared several distributions in their goodness of fit to our data with the Inverse Gamma distribution presenting the best results (lowest chi-square value and Kolmogorov–Smirnov p-value > 0.05).

To assess the asset life duration, we performed goodness-of-fit tests on 81 distributions to see which one was the best fit for the faulty stage data in dataset C-MAPSS (FD001). In Table A.12 we present the ten best results (according to the Chi-Square test) of the data fit. We also present in Table A.7 the parameters of the best fitting distributions. The Inverse Gamma, Inverse Gaussian and Log Normal distributions gave the best fitting results (see Fig. A.19). As shown in the histogram, the data reaches its peak at ≈100 cycles and the maximum faulty stage duration is around 300 cycles (≈80% of maximum total lifetime of 362 cycles). We present the probability density function of the three distributions with the best fit (Inverse Gamma, Inverse Gaussian and Log Normal) (see Tables A.8–A.10 and A.13).

$$f(x, a) = \frac{x^{-a-1}}{\Gamma(a)} \exp\left(-\frac{1}{x}\right) \tag{A.4}$$

$$f(x, \mu) = \frac{1}{\sqrt{2\pi x^3}} \exp\left(-\frac{(x-\mu)^2}{2x\mu^2}\right) \tag{A.5}$$

$$f(x, \mu, \sigma) = \frac{1}{\sigma x\sqrt{2\pi}} e^{\left(-\frac{(\ln(x)-\mu)^2}{2\sigma^2}\right)} \tag{A.6}$$

To shift and/or scale the distributions use the location and scale parameters. Specifically, with the transformation $x' = (x - \text{loc}) / \text{scale}$ (see Fig. A.17).

A.2. Nominal stage slope

To assess the slope, we performed goodness-of-fit tests on 81 distributions to see which one was the best fit for each feature of dataset C-MAPSS (FD001). In Table A.11 we present the best fit (distribution and parameters) for each of the C-MAPSS features. As an illustrative example, in Fig. A.18 we show the histogram of the slope of feature T24. The three best fitting functions are shown on top of the actual frequency data. The best fitting function is used to recreate the noise-free quasi-linear degradation behavior.

Table A.9
Parameters of best fit distributions for faulty stage duration data. The formal equation for each Probability Density Function (PDF) is referenced in the table.

Distribution	PDF	Parameters		
		1	2 (Location)	3 (Scale)
Inv. Gamma	Eq. (A.8)	$a = 0.29$	−10.49	595.03
Inv. Gaussian	Eq. (A.8)	$\mu = 0.35$	10.46	333.07
Log Normal	Eq. (A.9)	$\mu = 0.59$	$\sigma = 15.66$	94.24

Table A.10
Parameters of best fit distributions for slope data. The formal equation for each Probability Density Function (PDF) is referenced in the table.

Parameter	Distribution	Parameters			
		1	2	3	4
T24	Gamma	2.25E−01	−5.29E−32	2.86E−03	
T30	Johnson SU	−8.33E−01	1.71E−01	−2.33E−12	1.07E−11
T50	Half Gen. Normal	8.92E−02	−1.64E−08	5.09E−16	
P30	Levy	2.08E−05	6.38E−05		
Nf	Expon. Power	2.95E−01	−1.53E−31	1.72E−03	
Nc	Gen. Normal	2.43E−01	1.19E−26	6.93E−07	
Ps30	Log Normal	5.21E+00	−2.03E−15	2.12E−05	
Phi	Johnson SU	2.75E+00	2.12E−01	1.65E−10	1.91E−10
NRf	Gen. Gamma	1.47E−01	1.84E+00	−1.61E−32	8.84E−03
NRc	Johnson SU	−1.04E+00	9.63E−02	7.47E−12	5.56E−11
BPR	Half Gen. Normal	1.01E−01	−3.95E−11	1.63E−14	
htBleed	Expon. Power	2.27E−01	−9.54E−32	2.18E−03	
W31	Frechet	2.00E−01	1.58E−32	4.86E−03	
W32	Johnson SU	3.15E+00	2.39E−01	2.81E−10	2.32E−10

Table A.11
Parameters of best fit distributions for noise level data. The formal equation for each Probability Density Function (PDF) is referenced in the table.

Parameter	Distribution	Parameters			
		1	2	3	4
T24	Double Gamma	1.66E+00	5.92E−01	5.74E−02	
T30	Double Weibull	1.50E+00	6.54E−01	1.24E−01	
T50	Double Weibull	1.25E+00	4.26E−01	6.75E−02	
P30	Levy-Stable	1.71E+00	1.00E+00	4.65E−01	5.88E−02
Nf	Double Gamma	2.94E−01	4.26E−01	7.84E−02	
Nc	Inverse Gaussian	1.43E−02	−1.29E−01	2.29E+01	
Ps30	Laplace	3.85E−01	6.24E−02		
phi	Double Weibull	1.42E+00	4.11E−01	6.70E−02	
NRf	Generalized Normal	1.21E−01	4.21E−01	4.00E−10	
NRc	Double Weibull	1.34E+00	1.58E−01	2.91E−02	
BPR	Double Weibull	1.39E+00	5.25E−01	8.70E−02	
htBleed	Chi-Squared	1.61E−01	3.91E−01	3.51E−01	
W31	Johnson SB	6.17E−01	1.48E+00	2.39E−01	7.94E−01
W32	Double Gamma	1.35E+00	5.69E−01	5.74E−02	

A.3. Noise level

To assess the noise level, we performed goodness-of-fit tests on 81 distributions to see which one was the best fit for each feature of dataset C-MAPSS (FD001). In Table A.11 we present the best fit (distribution and parameters) for each of the C-MAPSS features.

A.4. Faulty stage duration

To assess the asset life duration, we performed goodness-of-fit tests on 81 distributions to see which one was the best fit for the faulty stage data in dataset C-MAPSS (FD001). In Table A.12 we present the ten best results (according to the Chi-Square test) of the data fit. We also present in Table A.7 the parameters of the best fitting distributions. The Inverse Gamma, Inverse Gaussian and Log Normal distributions gave the best fitting results (see Fig. A.19). As shown in the histogram, the data reaches its peak at ≈100 cycles and the maximum faulty stage duration is around 300 cycles (≈80% of maximum total lifetime of 362 cycles). We present the probability density function of the three distributions with the best fit (Inverse Gamma, Inverse Gaussian and Log Normal):

Table A.12
Best distributions for fitting data of the duration of faulty stage. Goodness of fit and results of Kolmogorov–Smirnov test are presented. The best distribution for the specific application is the Inverse Gamma distribution.

Distribution	Chi-square	*p*-value
Inverse Gamma	9.08	0.87 (>0.05)
Inverse Gaussian	9.28	0.96 (>0.05)
Log Normal	9.28	0.93 (>0.05)
Johnson SU	9.29	0.93 (>0.05)
Inverse Weibull	9.57	0.81 (>0.05)
NCT	9.68	0.79 (>0.05)
Fatigue Life	9.85	0.97 (>0.05)
Recip. Inv. Gaussian	10.59	0.97 (>0.05)
Alpha	12.21	0.79 (>0.05)
Exponential Weibull	13.89	0.93 (>0.05)

$$f(x, a) = \frac{x^{-a-1}}{\Gamma(a)} \exp\left(-\frac{1}{x}\right) \tag{A.7}$$

Table A.13
Parameters of best fit distributions for faulty stage duration data. The formal equation for each Probability Density Function (PDF) is referenced in the table.

Distribution	PDF	Parameters		
		1	2 (Location)	3 (Scale)
Inv. Gamma	Eq. (A.8)	$a = 0.29$	-10.49	595.03
Inv. Gaussian	Eq. (A.8)	$\mu = 0.35$	10.46	333.07
Log Normal	Eq. (A.9)	$\mu = 0.59$	$\sigma = 15.66$	94.24

$$f(x, \mu) = \frac{1}{\sqrt{2\pi x^3}} \exp\left(-\frac{(x-\mu)^2}{2x\mu^2}\right) \tag{A.8}$$

$$f(x, \mu, \sigma) = \frac{1}{\sigma x\sqrt{2\pi}} e^{\left(-\frac{(\ln(x)-\mu)^2}{2\sigma^2}\right)} \tag{A.9}$$

To shift and/or scale the distributions use the location and scale parameters. Specifically, with the transformation $x' = (x - \text{loc}) / \text{scale}$.

References

[1] K. Goebel, M. Daigle, A. Saxena, S. Sankararaman, R. I., J. Celaya, Prognostics: The Science of Making Predictions, 2017.
[2] Y. Lei, N. Li, L. Guo, N. Li, T. Yan, J. Lin, Machinery health prognostics: A systematic review from data acquisition to RUL prediction, Mech. Syst. Signal Process. 104 (2018) 799–834.
[3] C.-C. Liu, T.-Y. Sun, S.-J. Tsai, Y.-H. Yu, S.-T. Hsieh, Heuristic wavelet shrinkage for denoising, Appl. Soft Comput. 11 (1) (2011) 256–264.
[4] D. Wang, W. Guo, X. Wang, A joint sparse wavelet coefficient extraction and adaptive noise reduction method in recovery of weak bearing fault features from a multi-component signal mixture, Appl. Soft Comput. 13 (10) (2013) 4097–4104.
[5] F. Xu, Y.L. Tse, et al., Roller bearing fault diagnosis using stacked denoising autoencoder in deep learning and Gath–Geva clustering algorithm without principal component analysis and data label, Appl. Soft Comput. 73 (2018) 898–913.
[6] F. Xu, Z. Huang, F. Yang, D. Wang, K.L. Tsui, Constructing a health indicator for roller bearings by using a stacked auto-encoder with an exponential function to eliminate concussion, Appl. Soft Comput. 89 (2020) 106119.
[7] H. Lv, J. Chen, T. Pan, Z. Zhou, Hybrid attribute conditional adversarial denoising autoencoder for zero-shot classification of mechanical intelligent fault diagnosis, Appl. Soft Comput. 95 (2020) 106577.
[8] K. Lu, Y. Li, A robust locating multi-optima approach for damage identification of plate-like structures, Appl. Soft Comput. 75 (2019) 508–522.
[9] N. Roy, R. Ganguli, Filter design using radial basis function neural network and genetic algorithm for improved operational health monitoring, Appl. Soft Comput. 6 (2) (2006) 154–169.
[10] C. Saxena, D. Kourav, Noises and image denoising techniques: A brief survey, Int. J. Emerg. Technol. Adv. Eng. 4 (3) (2014) 878–885.
[11] M.M. Najafabadi, F. Villanustre, T.M. Khoshgoftaar, N. Seliya, R. Wald, E. Muharemagic, Deep learning applications and challenges in big data analytics, J. Big Data 2 (1) (2015) 1–21.
[12] I. Goodfellow, D. Warde-Farley, M. Mirza, A. Courville, Y. Bengio, Maxout networks, in: International Conference on Machine Learning, PMLR, 2013, pp. 1319–1327.
[13] D.K. Frederick, J.A. DeCastro, J.S. Litt, User's Guide for the Commercial Modular Aero-Propulsion System Simulation (C-MAPSS), Technical Report, 2007.
[14] A. Saxena, K. Goebel, D. Simon, N. Eklund, Damage propagation modeling for aircraft engine run-to-failure simulation, in: 2008 International Conference on Prognostics and Health Management, IEEE, 2008, http://dx.doi.org/10.1109/phm.2008.4711414.
[15] R.C. Smith, Uncertainty Quantification: Theory, Implementation, and Applications, Vol. 12, Siam, 2013.
[16] L. Li, L. Qu, Machine diagnosis with independent component analysis and envelope analysis, in: 2002 IEEE International Conference on Industrial Technology, 2002. IEEE ICIT '02., IEEE, http://dx.doi.org/10.1109/icit.2002.1189377.
[17] G. Gelle, M. Colas, C. Serviere, Blind source separation: A new pre-processing tool for rotating machines monitoring? IEEE Trans. Instrum. Meas. 52 (3) (2003) 790–795, http://dx.doi.org/10.1109/tim.2003.814356.
[18] X. Ma, Z. Hao, Multisensor data fusion based on independent component analysis for fault diagnosis of rotor, in: Advances in Neural Networks – ISNN 2004, Springer Berlin Heidelberg, 2004, pp. 744–749, http://dx.doi.org/10.1007/978-3-540-28647-9_122.
[19] B.P. Leão, J.P. Gomes, R.K. Galvão, T. Yoneyama, Application of blind source separation techniques for generation of phm useful information, in: Annual Conference of the Prognostics and Health Management Society, Prognostics and Health Management Society, 2009.
[20] B. Zhang, T. Khawaja, R. Patrick, G. Vachtsevanos, G. Vachtsevanos, M. Orchard, A. Saxena, Application of blind deconvolution denoising in failure prognosis, IEEE Trans. Instrum. Meas. 58 (2) (2009) 303–310, http://dx.doi.org/10.1109/tim.2008.2005963.
[21] S. Jiang, Z. Chen, Q. Shen, M. Wu, S. Ma, Damage detection and locating based on EEMD-fast ICA, J. Vib. Shock 35 (1) (2016) 203–209.
[22] Y. Yang, S. Nagarajaiah, Blind identification of damage in time-varying systems using independent component analysis with wavelet transform, Mech. Syst. Signal Process. 47 (1–2) (2014) 3–20, http://dx.doi.org/10.1016/j.ymssp.2012.08.029.
[23] B. Widrow, J. Glover, J. McCool, J. Kaunitz, C. Williams, R. Hearn, J. Zeidler, J.E. Dong, R. Goodlin, Adaptive noise cancelling: Principles and applications, Proc. IEEE 63 (12) (1975) 1692–1716, http://dx.doi.org/10.1109/proc.1975.10036.
[24] J. Antoni, R. Randall, Unsupervised noise cancellation for vibration signals: Part I—evaluation of adaptive algorithms, Mech. Syst. Signal Process. 18 (1) (2004) 89–101, http://dx.doi.org/10.1016/s0888-3270(03)00012-8.
[25] T. Su, S. Hattori, M. Ishida, T. Hori, Suppression control method for torque vibration of AC motor utilizing repetitive controller with Fourier transform, IEEE Trans. Ind. Appl. 38 (5) (2002) 1316–1325, http://dx.doi.org/10.1109/tia.2002.802894.
[26] C. Kar, A. Mohanty, Vibration and current transient monitoring for gearbox fault detection using multiresolution Fourier transform, J. Sound Vib. 311 (1–2) (2008) 109–132, http://dx.doi.org/10.1016/j.jsv.2007.08.023.
[27] F. Al-Badour, M. Sunar, L. Cheded, Vibration analysis of rotating machinery using time–frequency analysis and wavelet techniques, Mech. Syst. Signal Process. 25 (6) (2011) 2083–2101, http://dx.doi.org/10.1016/j.ymssp.2011.01.017.
[28] Z. Zhang, Y. Wang, K. Wang, Fault diagnosis and prognosis using wavelet packet decomposition, Fourier transform and artificial neural network, J. Intell. Manuf. 24 (6) (2012) 1213–1227, http://dx.doi.org/10.1007/s10845-012-0657-2.
[29] H.-C. Lin, Y.-C. Ye, B.-J. Huang, J.-L. Su, Bearing vibration detection and analysis using enhanced fast Fourier transform algorithm, Adv. Mech. Eng. 8 (10) (2016) 168781401667508, http://dx.doi.org/10.1177/1687814016675080.
[30] H.-C. Lin, Y.-C. Ye, Reviews of bearing vibration measurement using fast Fourier transform and enhanced fast Fourier transform algorithms, Adv. Mech. Eng. 11 (1) (2019) 168781401881675, http://dx.doi.org/10.1177/1687814018816751.
[31] D. Donoho, De-noising by soft-thresholding, IEEE Trans. Inf. Theory 41 (3) (1995) 613–627, http://dx.doi.org/10.1109/18.382009.
[32] J. Antoni, R. Randall, Optimisation of SANC for separating gear and bearing signals, in: Condition Monitoring and Diagnostic Engineering Management, Elsevier, 2001, pp. 89–96, http://dx.doi.org/10.1016/b978-008044036-1/50010-x.
[33] S. Hussain, H.A. Gabbar, Vibration analysis and time series prediction for wind turbine gearbox prognostics, Int. J. Progn. Health Manag. 4 (2013) 69–79.
[34] W. He, Q. Miao, M. Azarian, M. Pecht, Health monitoring of cooling fan bearings based on wavelet filter, Mech. Syst. Signal Process. 64–65 (2015) 149–161, http://dx.doi.org/10.1016/j.ymssp.2015.04.002.
[35] J. Wen, H. Gao, S. Li, L. Zhang, X. He, W. Liu, Fault diagnosis of ball bearings using synchrosqueezed wavelet transforms and SVM, in: 2015 Prognostics and System Health Management Conference, PHM, IEEE, 2015, http://dx.doi.org/10.1109/phm.2015.7380084.
[36] D. Helm, M. Timusk, Extraction of weak bearing fault signatures from non-stationary signals using parallel wavelet denoising, in: Applied Condition Monitoring, Springer International Publishing, 2019, pp. 3–10, http://dx.doi.org/10.1007/978-3-030-11220-2_1.
[37] K.-D. Kammeyer, K. Kroschel, Rekursive filter, in: Digitale Signalverarbeitung, Vieweg & Teubner Verlag, 2002, pp. 77–156, http://dx.doi.org/10.1007/978-3-663-09805-8_4, URL.
[38] H.-C. Trinh, Y.-K. Kwon, An empirical investigation on a multiple filters-based approach for remaining useful life prediction, Machines 6 (3) (2018) 35, http://dx.doi.org/10.3390/machines6030035.

[39] T. Kohler, D. Lorenz, A Comparison of Denoising Methods for One Dimensional Time Series, Vol. 131, University of Bremen, Citeseer, Bremen, Germany, 2005.
[40] R. Gopinath, C. Burrus, Wavelet transforms and filter banks, in: Wavelets, Elsevier, 1992, pp. 603–654, http://dx.doi.org/10.1016/b978-0-12-174590-5.50024-1.
[41] P. Vincent, H. Larochelle, Y. Bengio, P.-A. Manzagol, Extracting and composing robust features with denoising autoencoders, in: Proceedings of the 25th International Conference on Machine Learning - ICML'08, ACM Press, 2008, http://dx.doi.org/10.1145/1390156.1390294.
[42] Y. Bengio, Learning deep architectures for AI, Found. Mach. Learn. 2 (1) (2009) 1–127, http://dx.doi.org/10.1561/2200000006.
[43] B. Xia, C. Bao, Wiener filtering based speech enhancement with weighted denoising auto-encoder and noise classification, Speech Commun. 60 (2014) 13–29, http://dx.doi.org/10.1016/j.specom.2014.02.001.
[44] S. Araki, T. Hayashi, M. Delcroix, M. Fujimoto, K. Takeda, T. Nakatani, Exploring multi-channel features for denoising-autoencoder-based speech enhancement, in: 2015 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP, IEEE, 2015, http://dx.doi.org/10.1109/icassp.2015.7177943.
[45] Đ.T. Grozdić, S.T. Jovičić, M. Subotić, Whispered speech recognition using deep denoising autoencoder, Eng. Appl. Artif. Intell. 59 (2017) 15–22, http://dx.doi.org/10.1016/j.engappai.2016.12.012.
[46] S. Kumar, S.P. Rath, Far-field speech enhancement using heteroscedastic autoencoder for improved speech recognition, in: Interspeech 2019, ISCA, 2019, http://dx.doi.org/10.21437/interspeech.2019-2032.
[47] D.X. Wu, W. Pan, L.D. Xie, C.X. Huang, An adaptive stacked denoising auto-encoder architecture for human action recognition, Appl. Mech. Mater. 631–632 (2014) 403–409, http://dx.doi.org/10.4028/www.scientific.net/amm.631-632.403.
[48] P. Xiong, H. Wang, M. Liu, S. Zhou, Z. Hou, X. Liu, ECG signal enhancement based on improved denoising auto-encoder, Eng. Appl. Artif. Intell. 52 (2016) 194–202, http://dx.doi.org/10.1016/j.engappai.2016.02.015.
[49] P. Xiong, H. Wang, M. Liu, F. Lin, Z. Hou, X. Liu, A stacked contractive denoising auto-encoder for ECG signal denoising, Physiol. Meas. 37 (12) (2016) 2214–2230, http://dx.doi.org/10.1088/0967-3334/37/12/2214.
[50] E. Fotiadou, T. Konopczyński, J. Hesser, R. Vullings, Deep convolutional encoder-decoder framework for fetal ECG signal denoising, in: 2019 Computing in Cardiology Conference, CinC, Computing in Cardiology, 2019, http://dx.doi.org/10.22489/cinc.2019.015.
[51] J.-Y. Liu, Y.-H. Yang, Denoising auto-encoder with recurrent skip connections and residual regression for music source separation, in: 2018 17th IEEE International Conference on Machine Learning and Applications, ICMLA, IEEE, 2018, http://dx.doi.org/10.1109/icmla.2018.00123.
[52] P. Liu, P. Zheng, Z. Chen, Deep learning with stacked denoising auto-encoder for short-term electric load forecasting, Energies 12 (12) (2019) 2445, http://dx.doi.org/10.3390/en12122445.
[53] H. Ma, S. Ma, Y. Xu, M. Zhu, Deep marginalized sparse denoising auto-encoder for image denoising, J. Phys. Conf. Ser. 960 (2018) 012033, http://dx.doi.org/10.1088/1742-6596/960/1/012033.
[54] J. Yu, D. Huang, Z. Wei, Unsupervised image segmentation via stacked denoising auto-encoder and hierarchical patch indexing, Signal Process. 143 (2018) 346–353, http://dx.doi.org/10.1016/j.sigpro.2017.07.009.
[55] D. Aspandi, O. Martinez, F. Sukno, X. Binefa, Robust facial alignment with internal denoising auto-encoder, in: 2019 16th Conference on Computer and Robot Vision, CRV, IEEE, 2019, http://dx.doi.org/10.1109/crv.2019.00027.
[56] S.A. Dar, Deep variational auto encoder for dimensionality reduction, denoising in MNIST datasets using TensorFlow and keras, SSRN Electron. J. (2020) http://dx.doi.org/10.2139/ssrn.3578118.
[57] Z. Meng, X. Zhan, J. Li, Z. Pan, An enhancement denoising autoencoder for rolling bearing fault diagnosis, Measurement 130 (2018) 448–454, http://dx.doi.org/10.1016/j.measurement.2018.08.010.
[58] H. Liu, J. Zhou, Y. Zheng, W. Jiang, Y. Zhang, Fault diagnosis of rolling bearings with recurrent neural network-based autoencoders, ISA Trans. 77 (2018) 167–178, http://dx.doi.org/10.1016/j.isatra.2018.04.005.
[59] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, L. Bottou, Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion, J. Mach. Learn. Res. 11 (12) (2010).
[60] X. Guo, C. Shen, L. Chen, Deep fault recognizer: An integrated model to denoise and extract features for fault diagnosis in rotating machinery, Appl. Sci. 7 (1) (2016) 41, http://dx.doi.org/10.3390/app7010041.
[61] Z. Wang, T. Sun, X. Tian, Fault diagnosis of rolling bearing based on SDAE and PSO-DBN, in: 2019 Chinese Control and Decision Conference, CCDC, IEEE, 2019, http://dx.doi.org/10.1109/ccdc.2019.8833353.
[62] Q. Hu, P. Sun, S. Zhang, Research on fault diagnosis method based on KPCA-SDAE, J. Phys. Conf. Ser. 1314 (2019) 012085, http://dx.doi.org/10.1088/1742-6596/1314/1/012085.
[63] R. Liu, S. Feng, Y. Cai, M. Liu, State assessment and fault prediction method of distribution terminal based on SDAE and hierarchical Bayesian, in: 2019 IEEE Sustainable Power and Energy Conference, ISPEC, IEEE, 2019, http://dx.doi.org/10.1109/ispec48194.2019.8975050.
[64] C. Han, H. Hayashi, L. Rundo, R. Araki, W. Shimoda, S. Muramatsu, Y. Furukawa, G. Mauri, H. Nakayama, GAN-based synthetic brain MR image generation, in: 2018 IEEE 15th International Symposium on Biomedical Imaging, ISBI 2018, IEEE, 2018, http://dx.doi.org/10.1109/isbi.2018.8363678.
[65] E. Bolluyt, C. Comaniciu, Collapse resistant deep convolutional GAN for multi-object image generation, in: 2019 18th IEEE International Conference on Machine Learning and Applications, ICMLA, IEEE, 2019, http://dx.doi.org/10.1109/icmla.2019.00229.
[66] Q. Jin, X. Luo, Y. Shi, K. Kita, Image generation method based on improved condition GAN, in: 2019 6th International Conference on Systems and Informatics, ICSAI, IEEE, 2019, http://dx.doi.org/10.1109/icsai48974.2019.9010120.
[67] J. Islam, Y. Zhang, GAN-based synthetic brain PET image generation, Brain Inform. 7 (1) (2020) http://dx.doi.org/10.1186/s40708-020-00104-2.
[68] Y. Cai, X. Wang, Z. Yu, F. Li, P. Xu, Y. Li, L. Li, Dualattn-GAN: Text to image synthesis with dual attentional generative adversarial network, IEEE Access 7 (2019) 183706–183716, http://dx.doi.org/10.1109/access.2019.2958864.
[69] Text to image translation using cycle GAN, Int. J. Eng. Adv. Technol. 9 (4) (2020) 1294–1297, http://dx.doi.org/10.35940/ijeat.d8703.049420.
[70] P. Perera, M. Abavisani, V.M. Patel, In2I: Unsupervised multi-image-to-image translation using generative adversarial networks, in: 2018 24th International Conference on Pattern Recognition, ICPR, IEEE, 2018, http://dx.doi.org/10.1109/icpr.2018.8545464.
[71] A. Cherian, A. Sullivan, Sem-GAN: Semantically-consistent image-to-image translation, in: 2019 IEEE Winter Conference on Applications of Computer Vision, WACV, IEEE, 2019, http://dx.doi.org/10.1109/wacv.2019.00196.
[72] H. Emami, M.M. Aliabadi, M. Dong, R. Chinnam, SPA-GAN: Spatial attention GAN for image-to-image translation, IEEE Trans. Multimedia (2020) 1, http://dx.doi.org/10.1109/tmm.2020.2975961.
[73] Z. Wang, J. Wang, Y. Wang, An intelligent diagnosis scheme based on generative adversarial learning deep neural networks and its application to planetary gearbox fault pattern recognition, Neurocomputing 310 (2018) 213–222, http://dx.doi.org/10.1016/j.neucom.2018.05.024.
[74] L. Zou, Y. Li, F. Xu, An adversarial denoising convolutional neural network for fault diagnosis of rotating machinery under noisy environment and limited sample size case, Neurocomputing 407 (2020) 105–120, http://dx.doi.org/10.1016/j.neucom.2020.04.074.
[75] Q. Lyu, M. Guo, Z. Pei, DeGAN: Mixed noise removal via generative adversarial networks, Appl. Soft Comput. 95 (2020) 106478.
[76] A. Savitzky, M.J.E. Golay, Smoothing and differentiation of data by simplified least squares procedures., Anal. Chem. 36 (8) (1964) 1627–1639, http://dx.doi.org/10.1021/ac60214a047.
[77] A. Saxena, K. Goebel, D. Simon, N. Eklund, Damage propagation modeling for aircraft engine run-to-failure simulation, in: 2008 International Conference on Prognostics and Health Management, IEEE, 2008, pp. 1–9.
[78] J. Coble, J.W. Hines, Identifying optimal prognostic parameters from data: A genetic algorithms approach, in: Annual Conference of the Prognostics and Health Management Society, vol. 27, 2009.
[79] J. Coble, An Automated Approach for Fusing Data Sources to Identify Optimal Prognostic Parameters (Ph.D. thesis), Dissertation, University of Tennessee Knoxville, TN, 2010.
[80] J. Coble, J.W. Hines, Identifying suitable degradation parameters for individual-based prognostics, in: Diagnostics and Prognostics of Engineering Systems, IGI Global, 2013, pp. 135–150, http://dx.doi.org/10.4018/978-1-4666-2095-7.ch007.
[81] S.A. Niknam, J. Kobza, J.W. Hines, Techniques of trend analysis in degradation-based prognostics, Int. J. Adv. Manuf. Technol. 88 (9–12) (2016) 2429–2441, http://dx.doi.org/10.1007/s00170-016-8909-5.
[82] M. Rigamonti, P. Baraldi, E. Zio, I. Roychoudhury, K. Goebel, S. Poll, Ensemble of optimized echo state networks for remaining useful life prediction, Neurocomputing 281 (2018) 121–138, http://dx.doi.org/10.1016/j.neucom.2017.11.062.
[83] Y. Lei, N. Li, L. Guo, N. Li, T. Yan, J. Lin, Machinery health prognostics: A systematic review from data acquisition to RUL prediction, Mech. Syst. Signal Process. 104 (2018) 799–834, http://dx.doi.org/10.1016/j.ymssp.2017.11.016.
[84] M.L. Baptista, E.M. Henriques, K. Goebel, More effective prognostics with elbow point detection and deep learning, Mech. Syst. Signal Process. 146 (2021) 106987, http://dx.doi.org/10.1016/j.ymssp.2020.106987.
[85] V. Satopaa, J. Albrecht, D. Irwin, B. Raghavan, Finding a "kneedle" in a haystack: Detecting knee points in system behavior, in: 2011 31st International Conference on Distributed Computing Systems Workshops, IEEE, 2011, http://dx.doi.org/10.1109/icdcsw.2011.20.
[86] S. Czesla, T. Molle, J. Schmitt, A posteriori noise estimation in variable data sets – with applications to spectra and light curves, Astron. Astrophys. 609 (2018) 1–10.

[87] A. Radford, L. Metz, S. Chintala, Unsupervised representation learning with deep convolutional generative adversarial networks, 2015, arXiv preprint arXiv:1511.06434.
[88] R. Ayachi, M. Afif, Y. Said, M. Atri, Strided convolution instead of max pooling for memory efficiency of convolutional neural networks, in: International Conference on the Sciences of Electronics, Technologies of Information and Telecommunications, Springer, 2018, pp. 234–243.
[89] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, B. Frey, Adversarial autoencoders, 2015, arXiv preprint arXiv:1511.05644.
[90] J.E. Dzakowic, G.S. Valentine, Advanced techniques for the verification and validation of prognostics & health management capabilities, Mach. Fail. Prev. Technol. (MFPT 60) (2007) 1–11.
[91] J. Figueroa Barraza, E. López Droguett, M.R. Martins, Towards interpretable deep learning: A feature selection framework for prognostics and health management using deep neural networks, Sensors 21 (17) (2021) 5888.
[92] A. Saxena, J. Celaya, E. Balaban, K. Goebel, B. Saha, S. Saha, M. Schwabacher, Metrics for evaluating performance of prognostic techniques, in: International Conference on Prognostics and Health Management, IEEE, 2008, pp. 1–17.
[93] G. Treece, The bitonic filter: Linear filtering in an edge-preserving morphological framework, IEEE Trans. Image Process. 25 (11) (2016) 5199–5211.