Anomaly Detection on Time Series with Wasserstein GAN applied to PHM

Mélanie Ducoffe[1], Ilyass Haloui[2], and Jayant Sen Gupta[3]

[1,3] *Airbus AI Research, Toulouse, France*
melanie.ducoffe@airbus.com
jayant.sengupta@airbus.com

[2] *Airbus Operations, Toulouse, France*
ISAE Supaero, Toulouse, France
ilyass.haloui@airbus.com

ABSTRACT

Modern vehicles are more and more connected. For instance, in the aerospace industry, newer aircraft are already equipped with data concentrators and enough wireless connectivity to transmit sensor data collected during the whole flight to the ground, usually when the airplane is at the gate. Moreover, platforms that were not designed with such capability can be retrofitted to install devices that enable wireless data collection, as is done on Airbus A320 family. For military and heavy helicopters, HUMS (Health and Usage Monitoring System) also allows the collection of sensor data. Finally, satellites send continuously to the ground sensor data, called telemetries. Most of the time, fortunately, the platforms behave normally, faults and failures are thus rare.

In order to go beyond corrective or preventive maintenance, and anticipate future faults and failures, we have to look for any drift, any change, in systems' behavior, in data that is normal almost all the time. Moreover, collected sensor data is time series data. The problem is then anomaly detection or novelty detection in time series data.

Among machine learning techniques that can be used to analyze data, Deep Learning, especially Convolutional Neural Networks, is very popular since it has surpassed human capacities for image classification and object detection. In this field, Generative Adversarial Networks are a technique to generate data similar to a potentially high dimension original dataset. In our case, generate new data could be useful to enrich the learning dataset with generated abnormal data to make it less unbalanced. Yet we are more interested in the potential of such techniques to perform anomaly detection for high dimensional data, comparing newly observed data with data that could have been generated from a distribution built from normal examples.

Mélanie Ducoffe et al. This is an open-access article distributed under the terms of the Creative Commons Attribution 3.0 United States License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

1. INTRODUCTION

For years, aeronautics industry has been working on prognostics and health management in order to anticipate faults on aircraft and provide predictive maintenance services for its customers. Several PHM tools have been developed that successfully anticipate faults and give advice to airlines in order to reduce the operational impact. The main difficulty is usually to determine the health indicators that allow to accurately estimate components health and predict remaining useful life. First, we have used data that was available to build health indicators for components that had the most operational impact for our customers. The data was composed of snapshots of sensor values taken at specific moments of flight and event data, such as fault occurrences and maintenance actions. Building health indicators was done by gathering data analysts with system experts together to make sense of data by comparing sensor values in faulty conditions and in normal conditions. Thanks to good results with this first way of working, it was possible to convince our customers to give us access to their full flight sensor data, that became the cornerstone for the Skywise platform. With this new type of data available, we continued to work with the same setup, just adapting to the much richer data available.

The next step we are preparing is building health indicators for faults that have never occurred on the fleet. To achieve this objective, we cannot rely on labeled data and system expertise to determine health indicators. The strategy is to use anomaly detection and use the anomaly score as a health indicator candidate. As degradation progresses with the aging of the component, the anomaly score will increase until reaching

the end of life. Of course, as we do not know what will be the level of anomaly score when reaching the end of life, it is not possible to make a prediction without adding additional knowledge. The potential usage of anomaly scores will be discussed in the last section of the paper.

We have tested different approaches for anomaly detection in multivariate time series data. In this paper, we will focus on a technique using deep learning and generative models, namely Generative Adversarial Networks (GAN).

In the first section of this paper, we will present related works for anomaly detection, anomaly detection using deep learning and anomaly detection using GAN techniques. The second section will describe the adaptation we propose on GAN for anomaly detection, followed by a section describing experiments on MNIST datasets to benchmark with classical methods and on an accelerometer dataset to give an example of fault detection. In the fourth section, we will discuss how anomaly detection can be used concretely in a PHM context, addressing different questions that are important when dealing with anomaly detection.

2. RELATED WORK

Anomaly detection sums up techniques whose goal is to prevent the apparition of non-conforming data, denoted as abnormal data. It has been applied in various domains, ranging from fraud detection, cyber-security or image classification... It is a prevalent topic into the PHM community, with applications such as aircraft engine fault detection (Chandola, Cheboli, & Kumar, 2009) or wind turbine fault detection (Tolani, Yasar, Ray, & Yang, 2006) among others.

Firstly, we briefly overview shallow models for anomaly detection. Note that we do not consider cases where we have access to both normal and abnormal data and thus focus on the unsupervised setting consisting of training data made only of normal data.

Among the possible methods adopted in the literature to perform anomaly detection, Local Outlier Factor (LOF) and Isolation Forest are widely used candidates (Breunig, Kriegel, Ng, & Sander, 2000),(Liu, Ting, & Zhou, 2008). LOF aims to detect whether a sample is isolated or not by computing its distances to its neighbors. Whereas Isolation Forest computes a random partitioning of the features and it results that anomalies require less splitting to be identified. Although those previous methods may be efficient when considering low dimensional input data, their performance is limited when dealing with high dimensional data. Eventually, they have been naturally extending to the high dimensional cases using deep features: in (Deecke, Vandermeulen, Ruff, Mandt, & Kloft, 2018), Deecke used LOF, and Isolation Forest and combine them with deep features to represent the input data. Such deep features are intermediate vector representations outputs by popular networks trained on ImageNet. Those methods underperform deep learning methods for anomaly detection; thus they have been mainly used as a baseline. Since shallow methods are inadequate for the analysis of high-dimensional data, recent works have tackled anomaly detection using deep neural networks. Part of those works extend one class support vector machines to detect anomalies, also known as OC-SVM (Chen, Zhou, & Huang, 2001; Erfani, Rajasegarar, Karunasekera, & Leckie, 2016), but replace the support vector machine by a deep network: most of the normal input data are mapped into a hypersphere whereas mapping of anomalies is supposed to fall outside the hypersphere. Similarly, One Class Neural Network (OC-NN) learns a hyperplane on top of a neural network to separate normal from abnormal data (Chalapathy, Menon, & Chawla, 2018).

Note that certain works are image specific and rely on geometrical transformation invariance which is out of the scope of this paper. A recent work casts deep anomaly detection as a four player game. They assume that normal training data are local minimizers of an unknown function (*a standard assumption for anomaly scoring*), thus any neighbor to a normal training data that minimizes locally this function is necessarily an anomaly, based on the previous assumption. They use this strategy to generate "hard anomalies" that challenge the training process of a classifier that predicts whether the data is normal or not. Finally, this classifier is used for anomaly detection. However, the training of this method is highly unstable, and create hard anomalies without physical structure (Wolf, Benaim, & Galanti, 2018).

Generative Adversarial networks have been proposed as a new framework for estimating generative models via an adversarial process by (Goodfellow et al., 2014). Two models are trained simultaneously: a generative model **G** that captures the data distribution and a discriminative model **D** that estimates the probability that a sample came from the training data rather than from **G**. Figure 1 describes the training procedure.

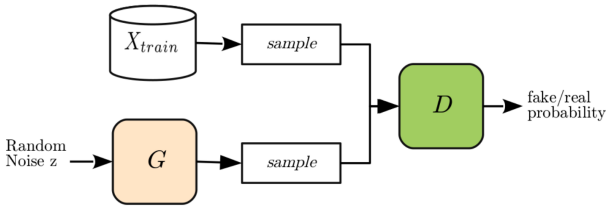


Figure 1. Generative Adversarial Networks

Indeed, several works derive generative deep modeling (either variational autoencoders or GANs) for anomaly detection (An & Cho, 2015; Schlegl, Seeböck, Waldstein, Schmidt-Erfurth, & Langs, 2017; Deecke et al., 2018). The principle is as follows: they learn the induced distribution and then assert whether a sample was part of this distribution, by mapping it to

the closest sample in the generated distribution. They tackled this mapping, either by learning an autoencoder learnt during the training like VAEs or DCAEs (An & Cho, 2015; Masci, Meier, Cireşan, & Schmidhuber, 2011; Donahue, Krähenbühl, & Darrell, 2016), or by doing optimization scheme in the latent dimension directly (Schlegl et al., 2017),(Deecke et al., 2018).

(Schlegl et al., 2017), (Deecke et al., 2018), whose methods are respectively denoted as AnoGAN and AD-GAN, optimize the latent dimension with gradient descent to approximate at best a given input. If they do succeed in reconstructing the data up to a certain threshold, then the data is recognized as normal. Otherwise, it is detected as anomalous. The main difference between both works consists of the loss function used for the reconstruction: while AD-GAN uses solely the euclidian distance between the input and the generator's output, AnoGAN balances this loss with the euclidian distance between a hidden layer of the discriminator. Indeed, AnoGAN assumes that the discriminator should not make any difference between the real sample and its projection. Moreover, AnoGAN runs the optimization only once by setting the initial value of the latent dimension randomly, while AD-GAN uses several seeds. Also, AD-GAN finetunes the generator during the optimization.

When it comes to generating time series data, previous works proposed to use Recurrent Neural Networks for both the generator **G** and the discriminator **D** to take into account the sequential nature of the input data. Eventually they generated fake time series with sequences from a random noise space. Note that those works still requires the duration of the input signal to be fixed.

None of (Schlegl et al., 2017), (Deecke et al., 2018) yield an efficient way of estimating the input noise that outputs a given input data. Moreover, training GANs is a challenging optimization task, even more, when considering non-image data, so that adding the encoder during the training stage as in (Donahue et al., 2016) may harness the generated samples quality.

In this work, we propose an end to end approach to perform anomaly detection with deep generative models. Unlike previous approaches using GANs for anomaly detection, we train an encoder that maps the input data to the noise distribution post-hoc. Simultaneous and independent work by (Schlegl, Seeböck, Waldstein, Langs, & Schmidt-Erfurth, 2019) also considers this approach to perform anomaly detection on medical images.

3. WASSERSTEIN GAN FOR ANOMALY DETECTION

3.1. Motivations

The original formulation of GAN usually suffer from the mode collapse problem. Instead of learning a good representation of the data, the generator only learns to reproduce a small fraction of the variability of the dataset. This is mainly due to GAN training procedure: since the generator is rewarded if it produces good realistic samples (by fooling the discriminator), it is not encouraged to produce other samples that might be not as good for the discriminator as the ones already found. Nevertheless, these other samples might help capture other existing "modes" in the dataset. To address this problem, a recent investigation focused on directly learning the distribution of the dataset using the 1-Wasserstein distance.

The Wasserstein distance is a powerful tool based on the theory of optimal transport to compare data distributions with wide applications in image processing, computer vision, and machine learning.

More formally, let X be a metric space endowed with a metric d_X. Let $p \in (0, \infty)$ and $\mathcal{P}_{p(X)}$ the space of all Borel probability measures μ on X with finite moments of order p, i.e. $\int_X d_X(x, x_0)^p d_{\mu(x)} < \infty \; \forall \; x_0 \in X$. The p-Wasserstein distance between μ and ν is defined as:

$$\mathcal{W}_p(\mu, \nu) = \inf_{\pi \in \Pi(\mu,\nu)} \Big(\int \int_{X \times X} d(x, y)^p d\pi(x, y) \Big)^{\frac{1}{p}} \quad (1)$$

When p = 1, $\mathcal{W}_1$ is also known as Earth Mover's distance (EMD) when considering discrete distributions. In this case, EMD can be computed exactly using linear programming at a cubic cost. Wasserstein Generative Adversarial Networks (W-GAN) (Arjovsky, Chintala, & Bottou, 2017) were first introduced as a solution to the mode collapse problem. Indeed, since the Wasserstein distance is continuous and is a global function, it forces the network not to focus on a subset of the distribution. Wasserstein GAN learns a generator on a noise distribution (generally gaussian) so that the output distribution matches the groundtruth distribution. They measure the quality of the generated distribution with the 1-Wasserstein distance. In its primal form, the Wasserstein distance requires to measure the expectation of the distances between two continuous distributions which may not be tractable in high dimension. Instead, the formulation of Wasserstein GAN relies on the dual expression of the 1-Wasserstein distance, which allows nicer optimization properties:

$$\mathcal{W}_1(\mathbb{P}_r, \mathbb{P}_\theta) = \sup_{f, ||f||_L \leq 1} \big(\mathbb{E}_{x \sim \mathbb{P}_r}[f(x)] - \mathbb{E}_{x \sim \mathbb{P}_\theta}[f(x)] \big) \quad (2)$$

Based on the primal form of the 1-Wasserstein distance, W-GAN considers solving the following objective function:

$$\max_{D, ||D||_L \leq 1} \big(\mathbb{E}_{x \sim \mathbb{P}_r}[D(x)] - \mathbb{E}_{z \sim p(z)}[D(G(z))] \big) \quad (3)$$

Assuming that we would reach this supremum, then one could optimize the generator **G** to minimize the Wasserstein dis-

tance between the distribution of generated samples with $\mathbb{P}_r$. Eventually, by iterating on this process, the output of the generator would theoretically converge to the true distribution $\mathbb{P}_r$ (Arjovsky et al., 2017).

The formulation of W-GAN relies on using a universal approximator of 1 Lipschitz function for the discriminator. Lipschitz functions are defined as follows:

Definition: Lipschitz

Let a function f be M-Lipschitz continuous if there exist a constant M > 0 such that:

$$\forall\, x_1, x_2 \ \in X,\ d_Y\big(f(x_1), f(x_2)\big) \leq M d_X\big(x_1, x_2\big)$$

While deep neural networks are Lipschitz by design [1], there are no necessary conditions than their Lipschitz constant equals 1, as required in Eq.(2). However, note that if we optimize Eq.(2) using a M bounded Lipschitz discriminator f, we end up converging to $M * \mathcal{W}_1(\mathbb{P}_r, \mathbb{P}_\theta)$, thus optimizing the generator to a cost function proportional to the exact Wasserstein distance. This implies that we only need to enforce that the discriminator has bounded Lipschitz constant. In practice, the only known method to upper-bound the Lipschitz constant of a neural network is by using weight clipping. Although this regularization scheme is theoretically sounded, its usage renders the training stage unstable. As a consequence, soft approximations are preferred in the literature, such as gradient regularization (Gulrajani, Ahmed, Arjovsky, Dumoulin, & Courville, 2017) or spectral regularization (Miyato, Kataoka, Koyama, & Yoshida, 2018).

3.2. Dealing with time series of various duration

In practice, data generated by sensors are of varying length. Except for satellites that have a mission with a periodic pattern, airplanes and helicopters have missions that do not last always the same time. On the other end, Wasserstein distance assumes that the samples live in the same space and thus have the same dimension: the same number of parameters and the same time length.

In the case where the phenomenon of interest is much smaller than the total mission length of the asset, the most common way is to fix the time length and cut the total mission into smaller chunks of fixed size. The choice of this length is sometimes tricky. It has to be small in order to avoid too many dimensions but large enough to capture the phenomenon. In the case where the phenomenon is the full-time series of the mission, one can pad the time series to obtain a unique time length (the largest of the dataset), but the user must be aware of the impact of padding and the false positive it might create, as the way the sequences are extended will become normality. Another solution consists in embedding the data using for example dictionary learning (Yazdi, Douzal-Chouakria, Gallinari, & Moussallam, 2018) where the atoms are time series of different length. However, this extension is left as future work. Finally, Dynamic Time Warping is a technique to compute distances between time series of different length and can be used to project on a common time mesh. Nevertheless, as we want to capture the dynamics of the signal, warping time is not desirable as it impacts the dynamics.

3.3. Evaluating and Monitoring Wasserstein GAN on time series

Among the possible issues occurring when training a GAN, the lack of correlation between the losses and the quality of the generated samples is probably the most challenging. When it comes to image generation, it has been frequently observed that the generator cost and the samples' quality were not always correlated: often the generator's cost increases but the samples' quality is actually improving. Hence, tuning a GAN requires a human in the loop to assert the loss visually. Although, when dealing with image generation, specific scores have been proposed like Parzen windows (Borji, 2019), those estimates can be deceptive (Theis, Oord, & Bethge, 2015). Thus, GANs for image generation are usually asserted by human evaluation using a distributed platform for human labor such as Mechanical-Turk. However, when dealing with time series, those scores are not adapted and the evaluation generally requires a domain expert. Also, (Li et al., 2019) attempt to evaluate the quality of their GAN using derived statistics. Indeed, unlike image generation, it may not be possible to evaluate visually how well the generator has captured the underlying distribution of time series data. Moreover, optimizing the hyperparameter setting become challenging as one cannot provide an evaluation metric. For all this reason designing a thoughtful metric to analyze the performance of the generator appears highly relevant.

When it comes to 1-Wasserstein distance, an approximation would consist in measuring the loss of the Discriminator. Indeed since the discriminator should converge up to a multiplicative constant, to the Wasserstein distance, we can use the discriminator's loss as a reference when the discriminator has converged. Also, a well-known estimator of the Wasserstein distance is the Earth Mover distance. In (Weed & Bach, 2017), authors demonstrate that the EMD of n random samples from both distributions is an estimator of the 1-Wasserstein distance. The quality of this estimator naturally depends on the size of the subset, as it convergences at a $\frac{1}{n^d}$ speed with d the number of parameters. Hence, the main limitation of this estimator is the computational cost of EMD, which is polynomial in n, and thus not tractable for large dimensions (many sensors, many time steps). A possible trick is to approximate the EMD with faster metric distributions such as Sinkhorn, or

[1] By feed-forward neural network we mean a function composed by affine transformations and point-wise non-linearities which are smooth Lipschitz functions (such as the sigmoid, tanh, elu, softplus, etc)

Sliced Wasserstein distance (Cuturi, 2013; Kolouri, Nadjahi, Simsekli, Badeau, & Rohde, 2019).

4. ANOMALY DETECTION

Anomaly detection for time series consists of identifying whether the testing data conform to the normal data distribution. We depict the overall architecture of the proposed method in Figure 2

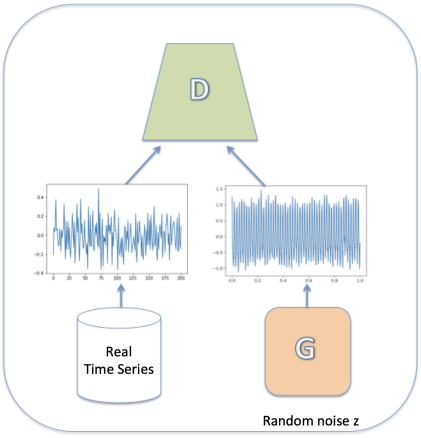
First, we train a Wasserstein GAN: a discriminator **D** tries to maximize the expectation of its predictions over natural data minus the expectation of its predictions over generated data, as depicted in Eq. (3). In a second step, a generator **G** tries to fool the discriminator by maximizing the expectation of the discriminator's predictions on generated data. This iterative process approximates the minimization of the 1-Wasserstein distance between the distribution of normal samples and the distribution of generated samples.

A third step consists in deriving the generative process into an anomaly scorer. This stage is independent of the training of GANs adopted. In that aim, previous works decoupled the anomaly scoring into two sub-metrics:

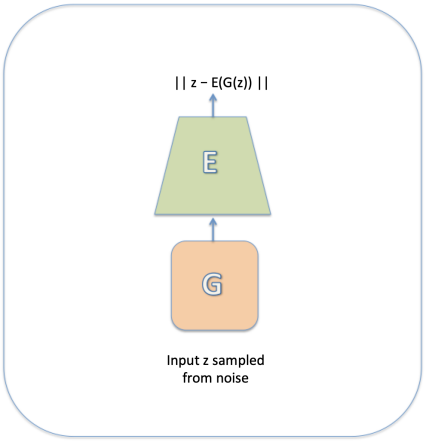
1. A Discriminator-based Anomaly Scorer:
   Since the Discriminator is trained to recognize normal data from generated data, (Schlegl et al., 2017) proposed to use the euclidian loss on some latent dimensions of the discriminator. However, we have no proof whether the anomaly is part of any distribution (either normal or generated). Thus, we do not know how the discriminator will behave when confronted with an anomaly.

2. A Reconstruction-based Anomaly Scorer:
   One of the main flaw raised towards those methods is that they are computationally extensive since they require to optimize the noise distribution to fit a sample. Other GAN frameworks integrate into their loss a cost to reconstruct the training data thanks to an encoder. Hence, they learn a bijective mapping such as VAEs or BiGAN. Eventually, thanks to the encoder, they compute the reconstruction error that will be used as an anomaly scorer. However, requiring a bijective mapping through the training impacts the quality of the Wasserstein GAN. Indeed, optimizing a Wasserstein GAN with a deterministic encoder comes to solving the Monge Mapping problem which is a non-convex optimization problem for which neither the existence nor the unicity is guaranteed in the general case (Brenier, 1991). When it comes to the Wasserstein distance, also known as the Kantorovich mapping, it assumes a probabilistic mapping between the distribution of normal data and the distribution of generated data.

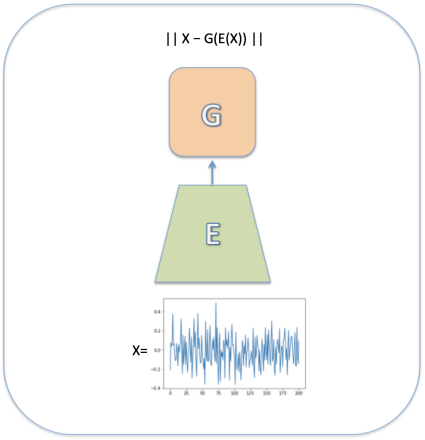
Wasserstein distance assumes a probabilistic mapping. Adding an encoder during the training phase creates an additional de-



(a) Step 1: training a W-GAN



(b) Step 2: freeze the generator and train an encoder



(c) Step 3: use the reconstruction error as an anomaly scorer

Figure 2. Description of the different steps involved in our anomaly detection

terministic constraint that makes the training of the generator harder. Nevertheless, at convergence of the W-GAN, it seems reasonable to assume that an almost deterministic mapping has been built. Eventually, it appears that a natural trade-off between learning a deterministic encoder and optimizing over the noise space is to train an encoder after training the generator. For this peculiar reason, we take a counter step to the previous approaches and train an encoder after training a W-GAN. Instead of constraining the output of the encoder to be sampled from a Gaussian distribution, we freeze the generator and stack it on top of the encoder to minimize the reconstruction over white noise.

5. EXAMPLES

We studied two use cases: one anomaly detection on images and the second one is an industrial Airbus use case made of one-dimensional time series. Our code and the description of the related hyperparameters will be made available on a public repository github.

5.1. First example on images

In this section, we investigate the usage of the Wasserstein GAN for anomaly detection on MNIST dataset.

5.1.1. Description of the image use case

The dataset has 60,000 digits for training and 10,000 for testing. Each digit is an image of 28x28 pixels and represents the first step towards the study of high dimensional data. We perform anomaly detection considering one class of digits as being abnormal and train the W-GAN and the encoder on the other digits of the training dataset. Testing is performed on the test dataset plus all the anomalous samples. This procedure has been used in (Zenati, Foo, Lecouat, Manek, & Ramaseshan Chandrasekhar, 2018) to investigate the performance of GAN in the field of anomaly detection. We find this approach closer to PHM context since normal data is usually multimodal and we want to represent this normality in order to measure the distance of anomalous samples to this normal representation. Note that the results presented here are different from the normal MNIST ones where normality is learnt on one digit, all other digits being considered as anomalies.

5.1.2. Results and comparison to AnoGAN

We train the W-GAN with encoder and perform the anomaly detection for each digit considered abnormal. The area under the precision-recall curve is computed and the model is compared to VAE (An & Cho, 2015), BiGAN and AnoGAN as shown in Table 1. Results are taken from (Zenati et al., 2018) and we can see that W-GAN with encoder outperforms these models. We believe that this is achieved thanks to the focus of the learning procedure on the distribution of normal data rather than samples. We can see from Figure 3 that when

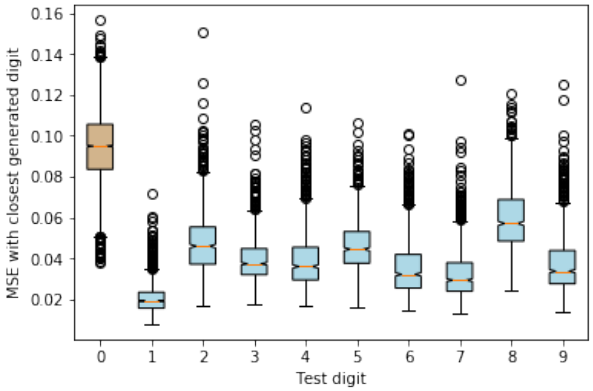


Figure 3. One vs. all: highlighting anomaly for digit 0

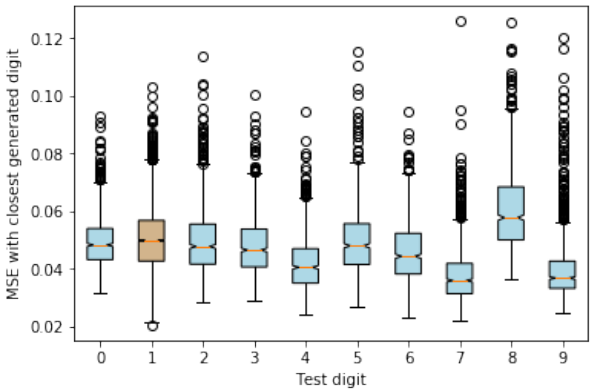


Figure 4. One vs. all: highlighting anomaly for digit 1

the model learns all digits except 0, the reconstruction error increases and becomes a significant anomaly detection score. The resulting reconstruction can be seen in Figure 5 where the model tries to interpret *zero* as another digit. When digit *one* is considered abnormal, the model has difficulties to recognize it as such as it is shown in the box-plots of reconstruction errors in Figure 4. We believe it is because digit 7 is in the training data and it is difficult to detect 1 when 7 is normal. It is the same when one tries to detect 7 when 1 is considered part of normality.

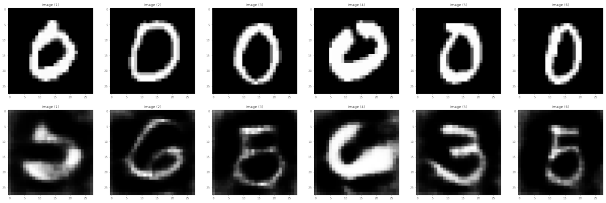


Figure 5. Images of zeros and their reconstruction when the digit is considered abnormal

Table 1. Area Under Precision Recall Curves (AUPRC) for each digit considered an anomaly

Abnormal digit	VAE	AnoGAN	BiGAN	our model
0	51.7 %	63 %	80 %	**97** %
1	6.3 %	30 %	30 %	**51** %
2	64.4 %	57 %	70 %	**89** %
3	25.1 %	45 %	55 %	**78** %
4	33.7 %	43 %	50 %	**83** %
5	32.5 %	44 %	55 %	**72** %
6	43.2 %	47 %	63 %	**87** %
7	14.8 %	40 %	40 %	**57** %
8	49.9 %	40 %	57 %	**90** %
9	10.4 %	37 %	38 %	**70** %

5.2. Example on sensor data

5.2.1. Description of sensor use case

In this section, we investigate the usage of the Wasserstein GAN for anomaly detection on time series data. We analyze the performance of W-GAN on time series from an external Airbus challenge: '*Airbus Helicopters Accelerometer Challenge*'. The challenge was open between January and May 2019 on the platform AIGym (*https://aigym.airbus.com*).

The challenge consists in learning the normality from time series of 1-minute duration made of accelerometer measures during the flight test phase of helicopters. The training set is made of 1677 time series and the validation and test sets are made of 594 and 1917 time series made of both normal and abnormal time series. The participants of the contest have to detect abnormal sequences in test and validation dataset, outputing 1 if the sequence is considered abnormal and 0 if this sequence is considered normal. Table2 provides more details on the dataset. The main difficulty is to learn only from normal data with lack of data, data diversity and also high dimension because of the temporal domain. The dataset consists of accelerometers signal from different directions, different positions, different helicopters and different flights, sampled at 1024Hz. It results into more than 60,000 time steps for each time series.

Two types of anomaly coexist in both the validation and test sets. Note that these are provided for information purposes only, as in an unsupervised setting, the type of anomalies is not known beforehand. For the challenge, competitors did not know that there were two types of anomalies. They were able to do multiple submissions on the validation dataset and only two submissions on the test dataset. Note that for the last dataset, there was another type of anomaly that was never seen in training and validation. Note that the type II of anomaly is local and thus harder to detect with a global reconstruction-based anomaly scorer.

We illustrate the shape of *normal* time series and the anomalies encounter in the test phase in Figure 6:

Table 2. Description of the 3 subset of data of the '*Airbus Helicopters Accelerometer Challenge*'. Type 0 samples are normal samples and Type I and II refers to the number of anomalies given their nature.

Type	0	I	II	Total
TRAIN	1677	-	-	1677
VALID	297	297	-	594
TEST	1497	108	312	1917

1. **Type I**: Loss of signal envelope symmetry

2. **Type II**: Frequency outlier

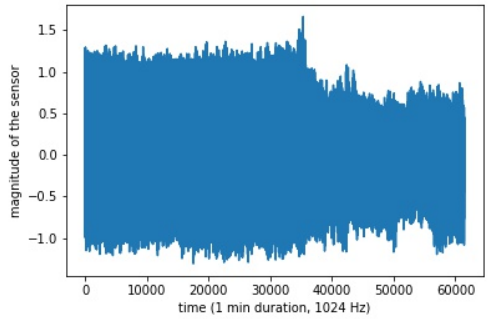
5.2.2. Pre-processing

The training set is made of few samples of high dimension, which is challenging to train a GAN. Theoretically, Wasserstein GAN may require an exponential number of training samples given the dimension in order to converge to the induced distribution of the data.

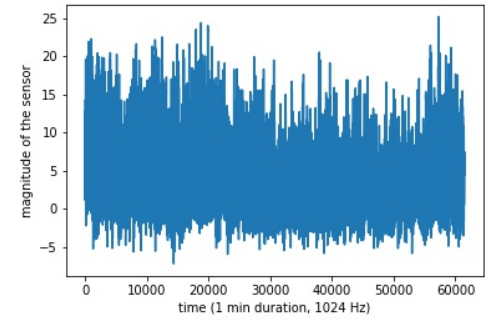
One possible solution, since we know the sampling frequency of the data is to express our time series into the frequency domain. This preprocessing is interesting for two reasons. First, due to the high frequency nature of our time series, we can expect that a reconsruction loss in the time domain will not be able to capture the diversity of the normal distribution. Secondly, the size of our normal training set is quite restricted; it is necessary to reduce the input dimension of our problem to apply deep learning for anomaly detection.

One solution to reduce the dimension is to convert our time series into the frequency domain. Although, it is a common practice to normalize the data when training a GAN, the magnitude of the power spectrum contains a lot of information of the time series, and removing it is likely to discard relevant information for the anomaly detection task. Hence we need to incorporate the magnitude of the power spectrum into the GAN. Eventually, our preprocessing consists in a vector whose last value is the norm of the power spectrum concatenated with the normalized power spectrum (also known as spectral density). Here are the full descriptions of our preprocessing. Thanks to this operation, we convert a time series of more than 60,000 time steps into a sparse vector of 514 values. Figure 7 illustrates the pre-processing step inherent with this method.

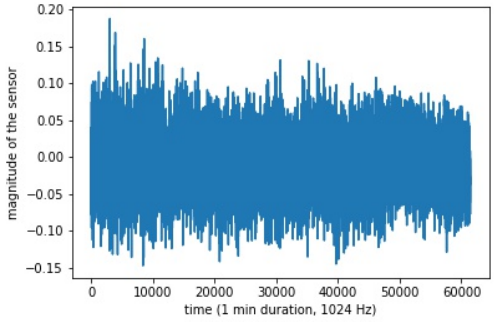
- we normalize the power spectrum composed of power computed for 513 frequencies;
- we concatenate the log of the normalization coefficient as the 514th dimension of this vector. We use the log of norm of the power spectrum as it can be orders of magnitude higher than the normalized values of the power spectrum.



(a) Normal (Type 0)



(b) Abnormal (Type I)



(c) Abnormal (Type II)

Figure 6. Visualization of non-normalized normal time series and their related anomalies in the *Airbus Helicopters Accelerometers Challenge* Challenge

Table 3. AUPRC on the two continuous anomaly scorer: the norm in the latent space ($|| z ||$) and the reconstruction error $|| x - \hat{x} ||$

AUPRC %	$\|\| z \|\|$	$\|\| x - \hat{x} \|\|$
VALID	73.22	88.11
TEST	79.08	80.83

The L1 distance on which the GAN is trained could lead the GAN to learn only the normalization value rather than correctly reproducing the full range of frequencies.

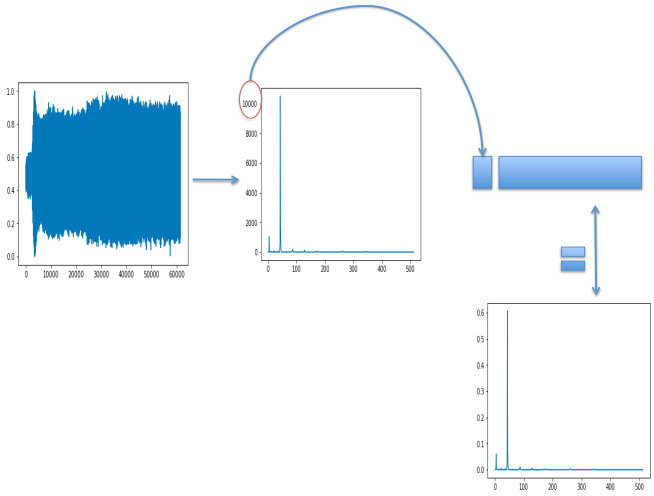


Figure 7. Description of Methodology

5.2.3. Results of W-GAN in the frequency domain

We evaluate the efficiency of our W-GAN by either testing whether the projection is sampled on the noise distribution, or by testing the reconstruction error. The AUPRC scores are presented in Table 3.

Eventually, we can establish an *optimal* threshold based on the F1 score along the reconstruction error to illustrate the best performance of the system. We observe on both the validation and test set that the normal time series are rarely considered abnormal. However, if the first type of anomalies is usually detected, as we can observe in the confusion matrices in Tables 5 and 6, it appears that the anomalies in the test set are more challenging for the W-GAN. However, we noticed that the anomalies of the test set are also challenging for other type of reconstruction-based anomaly scorer. Indeed, we compared the results obtained by the W-GAN with two baselines:

- FPCA in the time domain: we train a FPCA whose number of components is truncated over 1000 components to minimize the variance of the error reconstruction. Each time series is normalized by removing its mean component and dividing by its standard deviation (Ferraty &

Table 4. AUPRC on the W-GAN, FPCA and VAE on the reconstruction error

AUPRC %	W-GAN	FPCA	VAE
VALID	88.11	91.66	92.45
TEST	80.83	35.62	35.10

Table 5. Confusion matrix on the Validation set when setting the threshold using the F1 score

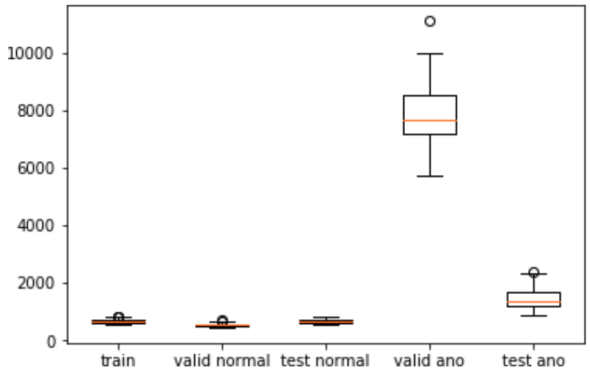
	Normal	**Abnormal**
Normal	296	1
Abnormal	199	98

Vieu, 2006).

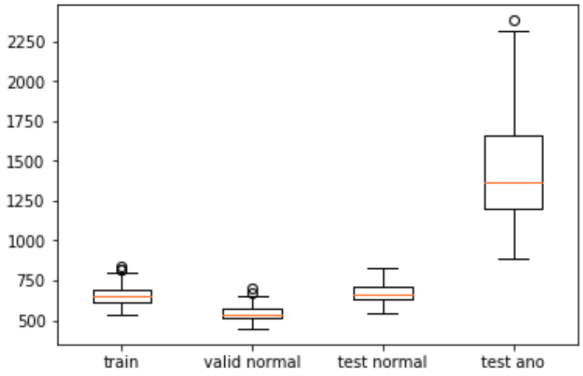
- VAE in the frequency domain: since the training of GANs is known to be challenging, we compare its performance with a variational autoencoder that is easier to train but is known to generate blurry samples. We used the same set of hyperparameters as the one proposed in the official Keras documentation (An & Cho, 2015).

In Table 4, we observe that W-GAN has a lower AUPRC on the validation set, compared to FPCA or VAE, but maintains its performances on the test set, unlike the baselines. It happens that W-GAN captures other information on the induced distribution than the VAE which helps it on the test set.

To understand why the detection of anomalies on the test set is challenging, we approximated the Wasserstein distance between the distribution of the training samples and the distributions of the validation and test samples (normal and anormal) using EMD. Indeed, since W-GAN approximates the Wasserstein distance, so does EMD. Thus EMD is a good indicator of the performances of W-GAN. These measures have been performed in the frequency domain. Plots in Figure 8 illustrate the different EMDs obtained on random subsets of size 100x100. If it appears that the Wasserstein distance between training samples and test's anomalies are higher than the Wasserstein distance between training samples and normal test samples, it is still lower than the Wasserstein distance with anomalies on the validation set. This gap may explain the differences in the results between the validation and test set.



(a) EMD between random subsets of size 100x100 between pairwise distributions, respectively (TRAIN, TRAIN), (TRAIN, VALID normal), (TRAIN, VALID abnormal), (TRAIN, TEST normal), (TRAIN, TEST abnormal)



(b) EMD between random subsets of size 100x100 between pairwise distributions, respectively (TRAIN, TRAIN), (TRAIN, VALID normal), (TRAIN, TEST normal), (TRAIN, TEST abnormal)

Figure 8. EMD between random subsets of size 100x100 between pairwise distributions. Since the Wasserstein distance between the distribution of normal training samples and the distribution of anomalies of the validation set is really high, we discard those distances in the second plot.

6. DISCUSSIONS

6.1. How to choose training data?

Assuming that delivered assets are in perfect condition, the first idea is to select data coming from the first flights of each asset of the fleet. Depending on the type of systems that you want to monitor, it may be worth taking into account a

Table 6. Confusion matrix on the Test set when setting the threshold using the F1 score

	Normal	Abnormal
Normal	1460	37
Abnormal (Type I)	44	64
Abnormal (Type II)	307	5

short running-in period where the behavior of the system may be a bit different from what should be considered as normal. Whether you should take into account a run-in period and how long it should be set shall be decided by system experts.

The second parameter is the duration of the training period. There is no definitive method to choose the duration of the training period but we can propose guidelines. The first criterion is to make sure that the training dataset covers as much as possible all operating and environmental conditions. This is easier if all assets of the fleet are used. Clustering the fleet by an operator may be a good idea if they operate in different operational and environmental conditions but it reduces the number of assets per training dataset and thus should be collected on a longer period to compensate. The second criterion is the size of the training dataset. Deep Learning models need a lot of data to train correctly as they have a lot of degrees of freedom. The last criterion concerns what part of the mission should we concentrate on. A first idea would be to take all the data available. With enough data and a sufficiently complex neural network, it could be possible in principle to learn normal behavior. In practice, it is more efficient to reduce the data to conditions where experts think the system behaves in a similar way. For instance, there is less variability when the aircraft is flying steadily than when it turns, accelerates, makes complex maneuvers.

6.2. Going from anomaly score at sequence level to anomaly score at cycle/mission level

In the techniques we are presenting, we have to choose a length for the time series sequence we want to generate with the GAN. Usually, it is smaller than the total mission or cycle length and after analyzing a complete flight, we have a list of anomaly score, one score per time sequence.

Aggregation of these anomaly scores at mission level is rather classic. The first way is to consider this list as a vector v and you can apply any norm of vector:

- $\|v\|_\infty$: maximum value of score, which would lead to a very sensitive anomaly scorer at cycle level
- $\|v\|_2$: Mean Square Norm that would tend to hide local bursts of anomaly
- ...

The second way is to consider this list as a time series itself and you can apply scan statistics. In this case, after applying a threshold to transform into time series of 0 (if the sequence is normal) or 1 (if the sequence is abnormal), scan statistics will filter out anomalies that are not sustained in time inside a mission or cycle.

6.3. How to set a norm and a threshold?

Usually, in anomaly detection techniques, a threshold is included in the algorithm that accounts for the percentage of an anomaly in the training dataset. For OCSVM, it would be parameter ν. For LOF and IsolationForest, it is sometimes called contamination, like in scikit-learn.

In our method, we use a reconstruction error between the initial data and the reconstructed one by a W-GAN. There is no built-in threshold in the method and it can only be set by choosing, for instance, a quantile of the distribution of the reconstruction error on the training dataset. In case there is only normal data in the training dataset, the notion of quantile can be extrapolated using extreme value theory. In practice, only expert feedback can help to adjust the threshold. This feedback should be asked using priority based on the anomaly score.

A question that also needs to be studied is the level of quality that is needed for a GAN to perform anomaly detection. In the case of image processing, L^2-norm is used and is coherent with how human vision works. In the case of time series, there is no equivalent and it is not that simple to choose the right norm to compare two time-series.

6.4. When using anomaly detection in PHM process?

As can be seen in this paper, unsupervised learning is more difficult than supervised learning. One key problem is that one does not know what problem to expect. The time series challenge showed that choosing a representation that was efficient to detect certain types of problems could make one completely blind to other types of problems.

As these methods are designed to discover unknown anomalies, it is impossible to link such anomalies to a maintenance action. It makes no sense to derive an advice from these anomalies. These methods are more dedicated for internal use, creating alerts that can be analyzed by engineers that can understand where these anomalies come from. This feedback from the experts can be used either to update the detection threshold in case it is too sensitive and also to assess the criticality and corrective action that could be put in place to solve the problem before it impacts the operations.

It is only when the problem is understood, the maintenance solution is chosen and the threshold is correctly set that the anomaly score can eventually be used as a proper health indicator.

7. CONCLUSION

In this paper, anomaly detection for time series data is presented. Among the different existing methods, we focused on a method based on Generative Adversarial Networks. Among the numerous variants of GAN, we chose the Wasserstein GAN that have the good property of avoiding mode collapse. On top of a W-GAN, we build an encoder that transforms the initial data into the latent space and combine it with the generator to build an auto-encoder. Results are shown on two use cases. The first one is MNIST image dataset in which we want to detect single digits as anomalies, considering the rest as normal, which we thought more coherent with reality than detecting all digits as abnormal except one. Except for 1 and 7 which are very similar, the method performs better than similar studies in the literature. The second use case is a time series use case that what used in 2019 for an Airbus time series challenge. Results are still to be improved but the method shows a good capacity to detect anomalies that were never seen during the training phase. As shown in this paper, significant pre-processing is required and finding a method that would make it unnecessary is yet to be found.

Different questions rise when using unsupervised learning for PHM. The difficulty of choosing a proper threshold is key but as described in the paper, these unsupervised methods can be used to rank the priority of analyzing flights among a large number of flights. Expert knowledge is of utmost importance and with a given expert workload, these methods can identify the most interesting flights to analyze. From this analysis, threshold setting, interpretation of anomalies, assessment of the criticality and definition of the proper maintenance action are outputs that can be leverage to transform the unsupervised task into a supervised one that can lead to the definition of a proper health indicator.

In a area where data is massively collected, this way of working may become the new paradigm. Nevertheless, we acknowledge there is still a lot of improvement and work to be done.

REFERENCES

An, J., & Cho, S. (2015). Variational autoencoder based anomaly detection using reconstruction probability. *Special Lecture on IE*, *2*, 1–18.

Arjovsky, M., Chintala, S., & Bottou, L. (2017). Wasserstein gan. *arXiv preprint arXiv:1701.07875*.

Borji, A. (2019). Pros and cons of gan evaluation measures. *Computer Vision and Image Understanding*, *179*, 41–65.

Brenier, Y. (1991). Polar factorization and monotone rearrangement of vector-valued functions. *Communications on pure and applied mathematics*, *44*(4), 375–417.

Breunig, M., Kriegel, H.-P., Ng, R., & Sander, J. (2000). Lof: identifying density-based local outliers. , *29*(2), 93-104.

Chalapathy, R., Menon, A. K., & Chawla, S. (2018). Anomaly detection using one-class neural networks. *arXiv preprint arXiv:1802.06360*.

Chandola, V., Cheboli, D., & Kumar, V. (2009). Detecting anomalies in a time series database.

Chen, Y., Zhou, X. S., & Huang, T. S. (2001). One-class svm for learning in image retrieval. In *Icip (1)* (pp. 34–37).

Cuturi, M. (2013). Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in neural information processing systems* (pp. 2292–2300).

Deecke, L., Vandermeulen, R., Ruff, L., Mandt, S., & Kloft, M. (2018). Image anomaly detection with generative adversarial networks. In *Joint european conference on machine learning and knowledge discovery in databases* (pp. 3–17).

Donahue, J., Krähenbühl, P., & Darrell, T. (2016). Adversarial feature learning. *arXiv preprint arXiv:1605.09782*.

Erfani, S. M., Rajasegarar, S., Karunasekera, S., & Leckie, C. (2016). High-dimensional and large-scale anomaly detection using a linear one-class svm with deep learning. *Pattern Recognition*, *58*, 121–134.

Ferraty, F., & Vieu, P. (2006). *Nonparametric functional data analysis: theory and practice*. Springer Science & Business Media.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems* (pp. 2672–2680).

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., & Courville, A. C. (2017). Improved training of wasserstein gans. In *Advances in neural information processing systems* (pp. 5767–5777).

Kolouri, S., Nadjahi, K., Simsekli, U., Badeau, R., & Rohde, G. K. (2019). Generalized sliced wasserstein distances. *arXiv preprint arXiv:1902.00434*.

Li, D., Chen, D., Shi, L., Jin, B., Goh, J., & Ng, S.-K. (2019). Mad-gan: Multivariate anomaly detection for time series data with generative adversarial networks. *arXiv preprint arXiv:1901.04997*.

Liu, F. T., Ting, K. M., & Zhou, Z.-H. (2008). Isolation forest. In *2008 eighth ieee international conference on data mining* (pp. 413–422).

Masci, J., Meier, U., Cireşan, D., & Schmidhuber, J. (2011). Stacked convolutional auto-encoders for hierarchical feature extraction. In *International conference on artificial neural networks* (pp. 52–59).

Miyato, T., Kataoka, T., Koyama, M., & Yoshida, Y. (2018). Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957*.

Schlegl, T., Seeböck, P., Waldstein, S. M., Langs, G., & Schmidt-Erfurth, U. (2019). f-anogan: Fast unsupervised anomaly detection with generative adversarial networks. *Medical image analysis*, *54*, 30–44.

Schlegl, T., Seeböck, P., Waldstein, S. M., Schmidt-Erfurth, U., & Langs, G. (2017). Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In *International conference on information processing in medical imaging* (pp. 146–157).

Theis, L., Oord, A. v. d., & Bethge, M. (2015). A note on the evaluation of generative models. *arXiv preprint arXiv:1511.01844*.

Tolani, D., Yasar, M., Ray, A., & Yang, V. (2006). Anomaly detection in aircraft gas turbine engines. *Journal of Aerospace Computing, Information, and Communication*, *3*(2), 44–51.

Weed, J., & Bach, F. (2017). Sharp asymptotic and finite-sample rates of convergence of empirical measures in wasserstein distance. *arXiv preprint arXiv:1707.00087*.

Wolf, L., Benaim, S., & Galanti, T. (2018). Unsupervised learning of the set of local maxima.

Yazdi, S. V., Douzal-Chouakria, A., Gallinari, P., & Moussallam, M. (2018). Time warp invariant dictionary learning for time series clustering: application to music data stream analysis. In *Joint european conference on machine learning and knowledge discovery in databases* (pp. 356–372).

Zenati, H., Foo, C. S., Lecouat, B., Manek, G., & Ramaseshan Chandrasekhar, V. (2018, February). Efficient GAN-Based Anomaly Detection. *ArXiv e-prints*.

BIOGRAPHIES

Mélanie Ducoffe holds an engineering diploma from Ecole Polytech Nice and a research master degree from Ecole Normale Supérieur de Cachan and a Ph.D. in computer science. She is working in Airbus corporate research center since 2019 on anomaly detection for time series data. Her research interests include machine learning, theoretical machine learning, explanations for back box systems and anomaly detection.

Ilyass Haloui is an engineer who graduated from ISAE-Supaero in 2018 and currently enrolled in a Ph.D. on behalf of Airbus Operations SAS focusing on maximizing the efficiency of predictive maintenance using sequential decision making. His research interests include theoretical frameworks for planning under uncertainty and predictive maintenance algorithms.

Jayant Sen Gupta holds an engineer diploma from Ecole Polytechnique in 2000, a master degree from Ecole Normale Supérieure de Cachan in 2001 and a PhD in computational mechanics from Ecole Normale Supérieure de Cachan in 2005. He has worked in Airbus corporate research center since 2005 working on boundary element for structural mechanics and high performance computing, modeling and propagation of uncertainties through design models, prognostics and health management and data science for predictive maintenance. His former publications on PHM concerned classification of PHM methods, continuous validation of PHM models, PHM at system level and implementation of PHM in industry.