Remaining Useful Life Prediction of Turbofans with Virtual Health Indicator: A Comparison of Particle Filter-Based Approaches

Etienne Jules, Francesco Cancelliere, Cécile Mattrand, Jean-Marc Bourinet

Remaining useful life prediction of turbofans with virtual health indicator: a comparison of particle filter-based approaches

1st Etienne Jules
Université Clermont Auvergne
Institute Pascal
Clermont-Ferrand, France
etienne.jules@sigma-clermont.fr

2nd Francesco Cancelliere
PHIMECA Engineering
LIMOS
Paris, France
cancelliere@phimeca.com

3rd Cécile Mattrand
SIGMA Clermont
Institute Pascal
Clermont-Ferrand, France
cecile.mattrand@sigma-clermont.fr

4th Jean-Marc Bourinet
SIGMA Clermont
LIMOS
Clermont-Ferrand, France
jean-marc.bourinet@sigma-clermont.fr

***Abstract*—The accurate estimation of the remaining useful life plays a crucial role in ensuring the reliability and efficiency of turbofan engines. In this study we address this objective by resorting to a virtual health indicator, previously developed by the authors for the estimation of the turbofan state of health, which is propagated in the future for estimating the engine end of life. The proposed approach consists in a combination of a multi-layer perceptron whose parameters are identified by a particle filter, in which the network act as a surrogate model for the hidden degradation state. The model is initialized on the basis of known degradation trajectories, and is recursively updated by the particle filter when new observations are available, providing flexibility to the method. To assess the effectiveness of the proposed approach, a comparison is made with a commonly used combination in the literature, which utilises a particle filter and a sum of two exponential functions. The results of the comparison demonstrate that the new approach achieves at least comparable results, and in the majority of cases, it outperforms the usual combination.**

***Index Terms*—remaining useful life (RUL), particle filter (PF), multi-layer perceptron (MLP), prognosis and health monitoring (PHM), health indicator (HI)**

I. Introduction

Prognosis and health management (PHM) seeks to enhance maintenance strategies and gain insights into the internal dynamics of degradation in industrial assets or complex systems by leveraging monitoring data. Data-driven models in PHM primarily address tasks such as anomaly detection [1], [2], degradation level assessment [3]–[5], remaining useful life (RUL) prediction [3]–[6], and failure mode identification [7]–[9]. The use of health indicators (HIs) is potentially at the core of most of these tasks. While HIs are fundamental to degradation level assessment and widely used for RUL prognostics [3]–[6], they are also valuable in anomaly detection, where monitoring HIs for sudden changes is an effective approach [1]. Additionally, HI trajectories can be analyzed to identify failure modes [10]. Recognizing the central role of HI construction in PHM, the development of HI-based strategies is therefore promoted.

In cases where a measurable quantity of interest (QoI) directly linked to the system health exists, it can serve as a physical HI. However, when no such QoI is available, a virtual HI must be constructed by fusing multiple monitored time series. Various methods have recently been proposed for VHI construction. Dimensionality reduction techniques, such as principal component analysis (PCA), holds significant importance [11]–[13], while supervised neural networks (NN) have emerged as alternative approaches [4], [14]. Genetic algorithms have also been used to construct VHIs, ensuring properties like monotonicity or scale-similarity [15], [16]. Other recent approaches employ unsupervized NN to generate VHIs while also optimizing the monotonicity property [10], [17].

In this study, we employ the HI generated through the approach proposed in [10]. This approach involves training a triplet siamese NN to encode available signal samples into a latent space. The siamese NN is then trained using a triplet loss, which drives samples to be represented in the latent space based on their temporal proximity to other samples. The HI is then constructed by measuring the distance in the latent space between a sample at current time and a sample taken at the start of life. By adopting this HI construction method, the present work proposes a novel approach for propagating the HI for RUL prognosis, which is the primary objective of the study.

The developed HI allows us to identify the current state of health (SOH) of the system, which can be used for monitoring and diagnostic purposes. To extend its utility to also perform the tasks of prognostics and estimation of the system RUL, it is necessary to develop a methodology to propagate the

indicator in the future, giving a prediction of the system SOH. Different methodologies have already been proposed in the literature to tackle this task, as summarized in [18] and [19], with applications respectively on machinery and lithium-ion batteries. RUL prediction techniques are generally classified in model-based, data-driven and hybrid, according to their reliance on the system physics.

Among all, particle filter (PF) has been widely used in the last years for prognostic purposes in different fields of application: [20] provides a list of different application cases in which PF based algorithm have been used as a prognostic tool, varying from crack growth, Li-Ion batteries, rolling bearings to wind turbines and jet engines. The reason of its success in this field can be summarized (but not limited) in:

- It is a recursive Bayesian algorithm, therefore well suited to solve real-time estimation problem.
- It is capable of accounting for the stochasticity of the problem and the noise affecting the measurements.
- It is applicable to non-linear and non-gaussian processes
- It is able to dynamically adjust model parameters

Recently, in the context of Li-Ion batteries and crack growth propagation, a PF coupled with a multi-layer perceptron (MLP) model has been proposed for RUL prediction based on HI trajectories. In the context of Turbofan prognostic however, particle filter is often coupled with a sum of two exponential functions to estimate the end of life (EOL) on the basis of some VHI [21]–[23]. The exponential-like degradation of turbofans explains this common choice, which would be ill-suited in the context of Li-Ion batteries. In this work, we recourse to the virtual health indicator presented in [10], and we propose a combination of a PF with a new architecture of multi-layer perceptron (MLP) model better suited for exponential-like degradation. The MLP acts as a surrogate model used to describe the process (or state) equation in the PF state space model. The PF thus aims at continuously optimizing the parameters of the MLP based on the already observed HI values. Additionally, we propose to initialize the particles with parameters found when fitting MLP on known VHI trajectories. Finally we investigate the use of an internal perturbation dynamics of the particles proposed in [24] for the first time in a context of RUL prediction. A brief explanation on particle filters is laid out in Section II, then we detail the proposed algorithm in Section III. The method is finally tested in Section IV on the turbofan experimental dataset and compared with a PF coupled with a sum of two exponential models, commonly used in the literature.

II. Particle filter

Particle filter, also called sequential Monte-Carlo [25], is an algorithm to sequentially estimate the posterior probability density function (PDF) of a hidden state x_k, given a series of noisy observations $z_{0:k-1}$, k denoting the current time step. The state-space evolution is described as a hidden Markov model defined by:

$$x_k = f(x_{k-1}, \omega_{k-1}) \tag{1}$$

$$z_k = g(x_k, k) + \eta_k \tag{2}$$

where f in (1) describes the process equation and g in (2) the measurement equation, and where ω and η are the process and measurement noises, respectively. To recursively estimate the posterior PDF $p(x_k|z_{0:k})$, the algorithm uses the prediction-update recurrence: first, a prior distribution $p(x_k|z_{0:k-1})$ is computed using the measurement available until time step $k-1$. The distribution is built by generating n_p state trajectories, also called particles, based on the process model (1). The second step consists in an update of the distribution: an importance weight is assigned to each particle:

$$\begin{gathered} w_k^i = w_{k-1}^i p(z_k|x_k^i); \\ i = 1, ..., N_s \end{gathered} \tag{3}$$

in which $p(z_k|x_k^i)$ is the likelihood $\mathcal{L}_k^i$ of the particle i. η_k is being considered normally distributed with zero mean and variance σ_η^2 independent from k. The likelihood is therefore computed as:

$$\mathcal{L}_k^{(i)} = p(z_{0:k}|x_k^i) = ((2\pi)^{k+1}|\Sigma_\eta|)^{-0.5} \exp\left\{-\frac{1}{2}\left(z_{0:k} - g(x_k^i, 0:k)\right)^T \Sigma_\eta^{-1}\left(z_{0:k} - g(x_k^i, 0:k)\right)\right\} \tag{4}$$

representing the probability of observing z_k given the state x_k^i. Here Σ_η and Σ_η^{-1} denotes for $(k+1)\times(k+1)$ diagonal matrices respectively filled with σ_η^2 and $\frac{1}{\sigma_\eta^2}$ on their diagonals, i.e. the covariance matrix and its inverse. $z_{0:k}$ and $g(x_k^i, 0:k)$ are $(k+1)$-sized vectors gathering the values until time k respectively of, the observations and trajectory of particle i at time k. It is worth noticing that this two terms $z_{0:k}$ and $g(x_k^i, 0:k)$ in Eq. (4) indicate that $\mathcal{L}_k^{(i)}$ is computed for the entire set of available measurements; this is in contrast with the classical PF, in which the likelihood is usually computed just based on the last observation, exploiting the Markov hypothesis.

In its original form, PF suffers from the degeneracy problem [26]: all particles but one tend to have an importance weight close to zero, hence the entire distribution collapses to one single particle. A common way of tackling this issue is the sampling importance resampling (SIR) scheme [27]: the importance weights are normalized as

$$\hat{w}_k^i = \frac{w_k^i}{\sum_{j=1}^{N_s} w_k^j} \tag{5}$$

and the trajectories x_k^i are hence resampled on the basis of the normalized importance weights. This means that the trajectories with high importance weights are more likely to be resampled, thus avoiding the degeneracy problem, introducing on the other hand a loss of diversity on the particles [25].

A. Surrogate-based PF

The classical PF methodology relies on the availability of the process model Eq. (1) and on the measurement model Eq. (2). However, the use of a VHI, with no physical meaning

and thus no evolution model, renders the process and measurement equations unknown. To solve this issue, a common method is to use a parameterized surrogate model to which a white noise is added for the measurement equation. The states are then considered as the parameters of this surrogate model to which a white noise is added at each cycle. For more details please refer to Eqs. (6) and (7). The choice of the surrogate model is therefore crucial. In this work we compare two different hybrid methods: in one case we resort to a combination of PF and a double-exponential model (i.e. sum of two exponential functions) as it was used extensively in the literature [21]–[23], while in the second case, a new architecture of multi-layer perceptron (MLP) is used as a surrogate model.

The PF state-space formulation when using a surrogate model f with parameters x is now rewritten as:

$$x_k = x_{k-1} + \omega_{k-1} \tag{6}$$

$$z_k = g(x_k, k) + \eta_k \tag{7}$$

in which x_k is now a state vector containing the parameters of the surrogate model at time step k.

The evolution of the model parameter in Eq. (6) is a random perturbation of the previous state vector x_{k-1}, which performs an exploration of the parameterized function family of g, depending on the Gaussian process noise ω_{k-1} with zero mean. The choice of ω_{k-1} variance is of prime importance in this application: a too low value will not guarantee a large enough exploration of the state, while a too large value will results in the algorithm instability. This choice is detailed in III-C.

As mentioned previously two different surrogate models are compared in this work: a sum of two exponential also referred as double exponential (DE) in the literature, and a multi-layer perceptron (MLP).

1) DE surrogates: DE is defined by a sum of two exponential functions as follows:

$$g(k, (a_k, b_k, c_k, dk)) = a_k \exp b_k k + c_k \exp d_k k \tag{8}$$

where (a_k, b_k, c_k, d_k) are the function parameters and thus the states x_k in the PF state-space equations in Eqs. (6) and (7). This choice of surrogate model is very common in the literature of HI prediction with PF. This is due to the degradation dynamics of most of the complex systems that tend to follow an exponential growth. It is in particular verified for the application case studied here in Section IV, where HI trajectories show an exponential-like behaviour when the degradation appears (see Fig. 2).

2) MLP surrogates: A combination of particle filter and neural network has been firstly proposed by [28] and later investigated in different applications fields [29]. In this work we resort to a simple MLP architecture composed by four layers (one input, two hidden layers and one output layer), with three neurons per hidden layer. The input is a single value, i.e. the time step k, and the output is composed of a single neuron returning the predicted HI value. The MLP consists then of 15 weights and 7 biases, for a total of 22 parameters to completely describe the network represented in Fig. 1. The input-output relationship is described as:

$$g(\Theta_k, b_k, k) = h_O(\sum_{j=1}^{3} \Theta_k^{(j)} H_2^{(j)}(k) + b_k^{(0)}) \tag{9}$$

$$H_2^{(j)}(k) = h_2(\sum_{i=1}^{3} \Theta_k^{(i,j)} H_1^{(i)}(k) + b_k^{(j)}) \tag{10}$$

$$H_1^{(i)}(k) = h_1(\Theta_k^{(i)} k + b_k^{(i)}) \tag{11}$$

where Θ_k and b_k represent the MLP weights and biases and thus the states x_k in the PF state-space equations in Eqs. (6) and (7). i and j respectively refer to the neuron number of the first and second hidden layers, while h_O, h_2, h_1 refer to the activation functions of the output, second hidden and first hidden layers, respectively. In this work, the activation functions $h_O(\cdot)$, $h_2(\cdot)$ and $h_1(\cdot)$ are respectively set as linear, exponential and scale exponential linear unit (SELU) functions. The choice of the two last layers activation functions is to perform a sum of exponential, which is well fitted to an exponential-like degradation. The first hidden layer activation function choice is for previously adding some non-linearity, expanding the possibilities of the defined parameterized function family, and therefore improve its adaptability to degradation trajectories containing exponential growths. It is also advised in the literature on neural network to use pseudo-linear unit activation functions e.g. rectified linear unit (ReLU), SELU. The choice of such a simple network architecture is motivated by the low computational efficiency of PF, whose task is to recursively estimate the MLP parameters. Increasing the network complexity could on one side improve the approximation capability of the algorithm, but on the other side it would drastically increase the number of parameters and the computational burden of the entire PF algorithm.

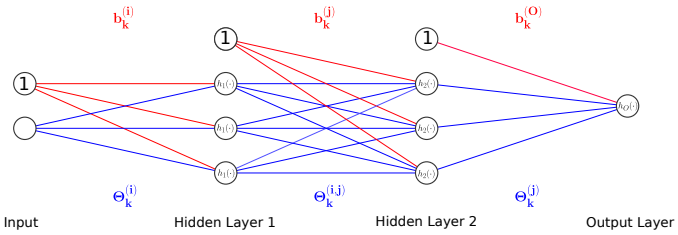

Fig. 1: Proposed MLP architecture

III. Algorithm Details

A. Particles initialization and reintroduction

The initialization of the PF with plausible particles plays a crucial role in the algorithm performances. Randomly initializing the particles, i.e. the parameters x_0 in Eq. (7), would lead to poor results, especially with g lying in a large parameterized function family as defined in the previous section. To do so, it is here proposed to initialize the particles with parameters found when fitting g on training HI trajectories. Once all the

training trajectories are fitted with their optimal parameters, the initial particles are initialized by randomly taking the parameters of one training trajectory fit. Additionally, to avoid a decrease in the diversity of the particles along the PF cycles, initial particles are randomly reintroduced at each resampling stage. The percentage of particle reintroduced instead of being resampled is arbitrarily fixed at 10%. For fitting $g(x, k)$ with MLP surrogate models, a classical neural network training is achieved to minimize the mean square error. For the DE surrogate model a least-square minimization if performed with the Levenberg-Marquardt algorithm.

B. Particle likelihoods variance

To determine the likelihood variance σ_η^2 used for the particle weight estimation performed in Eq. (4), it is first needed to characterize the noise η defined in Eq. (7). This noise is estimated by the difference between known HI trajectories (i.e., the training set) and their respective fits $g_{(x,k)}$ with optimal parameters $\widehat{x}$. Fortunately these best fit parameters for each training trajectories are already computed for the initialization of particles so this does not require any additional computational burden. The estimated mean $\widehat{\mu}_\eta$ and variance $\widehat{\sigma_\eta^2}$ of η are therefore estimated over all time steps of the training trajectories. It was found by the authors that the mean was close to zero for both the MLP and DE surrogate models, which justifies the hypothesis of zero mean gaussian noise η. In the particle filter, the value used for σ_η^2 should not be lower than $\widehat{\sigma_\eta^2}$. Otherwise, a low weight would be assigned even to the particles that best fit the HI trajectory, resulting in minimal distinction between the weights of good and poor particles. Therefore, in this study, different values for the parameter σ_η^2 were explored, based on the true noise variance $\widehat{\sigma_\eta^2}$ obtained for a given function f (i.e., MLP or double exponential). Specifically, the values $[\widehat{\sigma_\eta^2}, 2\widehat{\sigma_\eta^2}, 5\widehat{\sigma_\eta^2}, 10\widehat{\sigma_\eta^2}, 20\widehat{\sigma_\eta^2}]$ were tested. For both MLP and DE, $\widehat{\sigma_\eta^2}$ was estimated to be approximately 0.015. And the best results of PF were achieved with $\widehat{\sigma_\eta^2} = 0.075$ for both models.

C. Jittering variance

The selection of the jittering variance σ_ω^2 is of paramount importance in PF when employing the particle jittering strategy defined in Eq. (6). A common choice is first to adapt this jittering variance at each step, and not to use a fixed value. [29] proposed an exponentially decreasing jittering variance across time steps, equal for each parameter. The idea is to initially let the particles explore a wide possibility of parameters, and then decrease the jittering to focus the exploration on the found trajectories. In our case the beginning of the HI are often constant around zeros (see Fig. 2) so exploring at the beginning is not a very effective strategy. [30] proposed to update each parameter $x_k^{(i)}$ of f, based on the values range observed in the current particle set, the number of particles and the number of parameters. [24] refined this strategy with a variance depending also on the parameter value range of the current particles, but also depending on the effective sample size of the particle filter at the same time step. This last approach in jittering is the one used in this work. At each time step k the jittering variance for each component $x_k[d]$ of any particle x_k^i is defined as:

$$(\sigma_\omega^2)^k[d] = J \cdot \widehat{IQR}_{x_k[d]} \cdot ESS_k^{-\frac{1}{3}} \tag{12}$$

where $\widehat{IQR}_{x_k[d]}$ is the normalized inter-quartile range of values of the parameter x_d at time k over all the particles, and ESS_k is the effective sample size (ESS) at time k defined as:

$$ESS = \frac{1}{\sum_{i=1}^{n_p} (\hat{w}_i^k)^2} \tag{13}$$

where n_p denotes for the number of particles. The ESS evaluate the degeneracy of particles. If all particles tend to have similar weights then ESS will tend to n_p, if all particles but a few have weights close to zero, then ESS will tend to this few number of particles with high weights. With this jittering dynamic, a low ESS triggers an increase in jittering variance and thus a greater exploration of the particles, while a high ESS decreases this exploration. In [24] the authors found a value of $J = 1.68$ based on a general hypothesis; however it is unlikely that this value is optimal for all use cases, therefore J is here set as an hyperparameter for optimizing the jittering dynamic. In this work multiple values have been tried and the best performances were obtained with $J = 1.5$.

D. Additional a priori on particles

The weighting of the particles given their likelihood in Eq. (4) ensures the consistency between particles and measurements until the observation time t. To further improve the quality of the prediction, additional a priori can be enforced on the trajectories via filtering the particles. This consists in assigning a zero weight to the particles whose trajectories do not comply to some predefined rules. In this work we filtered out the particles based on:

Monotonicity
: HIs are by definition monotonous. However, the noise in the observations can depict some non-monotonicity; this can results in a non-monotonous prediction, that is incorrect per se.

Threshold reaching
: Particles are propagated in the future for a fixed horizon, defined as 110% of the longest HI trajectories on the training set. If a particle does not encounter the threshold before the horizon, it is filtered out.

Max derivative
: The parameterized function family of g (sum of two exponential or MLP with exponential activation function) encompass functions with sudden exponential increases that do not fit well with the HI trajectories. To filter these cases out, an a priori is set such that a particle can not have a derivative higher than the highest derivative found in the training set.

The a priori filtering is analogous to a prior distribution constructed from the training set. It serves to filter out particles that exhibit substantial deviations or are improbable based on this prior distribution.

E. RUL Prediction and confidence intervals

The set of n_p particles represent a plausible representation of the HI trajectories. At each time step k and for each particle i it is possible to define a end of life (EOL_k^i) as the future time step at which the trajectory reaches a failure threshold. The threshold is set in this work as the median of the highest values for each training HI trajectory. Consequently, it is possible to define the RUL of each particle as:

$$RUL_k^i = EOL_k^i - k \tag{14}$$

representing the number of time steps to reach the EOL from the current time k. The $\widehat{RUL_k}$ is then computed as the median of all RUL_k^i. The choice of the median with respect to the mean is motivated by the exponential evolution of the HI, which calls for a robust predictive statistics, i.e. the median. Finally, a confidence interval (CI) of this RUL prediction is built by taking the 5% and 95% quantiles of the RUL found by all the particles.

IV. Experimental results

A public turbofan engine dataset [31] was used to assess the performance of proposed method. It results from simulated runs of the commercial modular aero-propulsion system simulation (C-MAPSS) developed at NASA Army Research Laboratory. For each engine simulated, a multivariate times series of 21 monitored signals and 3 operating condition signals are available, for a total of 24 dimensions. The sampling frequency is one value per flight simulation. The monitored signals all represent a specific physical measure on the turbofan engine, e.g. temperature or pressure at critical spot of the engine, core or fan rotation speed. In this work a subset of 100 engines are used. Each of them is simulated under the same operating conditions, but can experience one of two failure modes, i.e. high-pressure compressor degradation or fan degradation.
Among the 100 engines, the 80 first engines are used as the training set and 20 last ones as the testing set. HI trajectories are here obtained from the siamese neural network proposed in [10] which is optimized on the training set. Once trained this HI model is used to produce HI trajectories for each of the engines in the training and testing set. The obtained HI trajectories of the testing set are drawn in Fig. 2. The proposed PF method for online RUL prediction is performed on each instance of the testing set. The training HI trajectories are used for initializing the particles.
To assess the performance of the on-line RUL prediction two indicators are computed. The first indicator is the cumulative relative error (CRE) defined as:

$$CRE = \sum_{k \in K} \frac{|\widehat{L_k} - L_k^*|}{L_k^*} \tag{15}$$

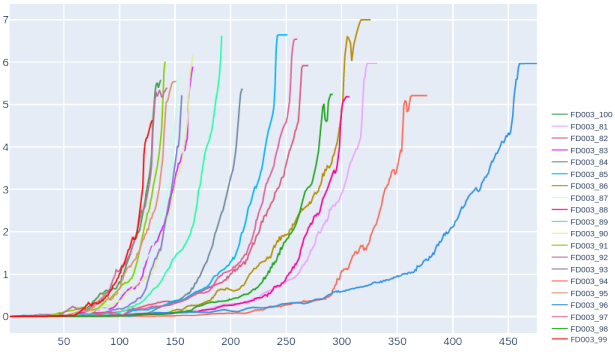

Fig. 2: HI trajectories of the testing set

where k a particular time step, K a set of time step where the RUL prediction performance is assessed, $\widehat{L_k}$ the predicted RUL and L_k^* the true RUL. This indicator relates to the point prediction performance, and needs to be minimized.
A second indicator is developed to assess the performance of the CI prediction. It is referred as CI coverage (CIC) and defined as follow:

$$CIC = \sum_{k \in K} \mathbb{1}_{L_k^* \in \widehat{CI_k}} \frac{L_k^*}{W_{\widehat{CI_k}}} \tag{16}$$

where $\mathbb{1}_{L_k^* \in \widehat{CI_k}}$ is the indicator function that takes one if the true RUL lies in the predicted RUL CI, and $W_{\widehat{CI_k}}$ is the CI width. The objective of this indicator, which is to be maximised, is to first reward RUL CIs that include the true RUL, but also to reward more the CIs containing the true RUL that are narrower, with this reward modulated by the true RUL. A wide CI when the true RUL is low is more penalized than when it is high.

To get a better insight on the performance of the two tested strategies, their results are analyzed over the entire life of the engines and in the last quarter of their life where the RUL prediction is more critical. The results of the online RUL prediction are shown in Fig. 3 for the five first testing engines. The overall performance indicators, summed up over the entire testing set, are summarized in Table I.

While the CRE indicates that the model with the double exponential has a better prediction accuracy over the entire life, it also reveals a better prediction accuracy of MLP model during the last 25% of an engine life. For predictive maintenance it can be argued that a point prediction is only relevant towards the end of a component life for planning maintenance operation, and therefore the MLP model would be more satisfactory on that matter. Moreover, the confidence interval of the MLP model is better calibrated than the one of DE model as shown by the CIC indicator both on the full life and last 25% life of an engine. These results can also be observed in Fig. 3: while the CI of the DE model always encompasses the true ground truth during the first half of life, it rarely does towards the end. By comparison the MLP model almost always encompasses the ground truth RUL towards the end of life while in the beginning of life for

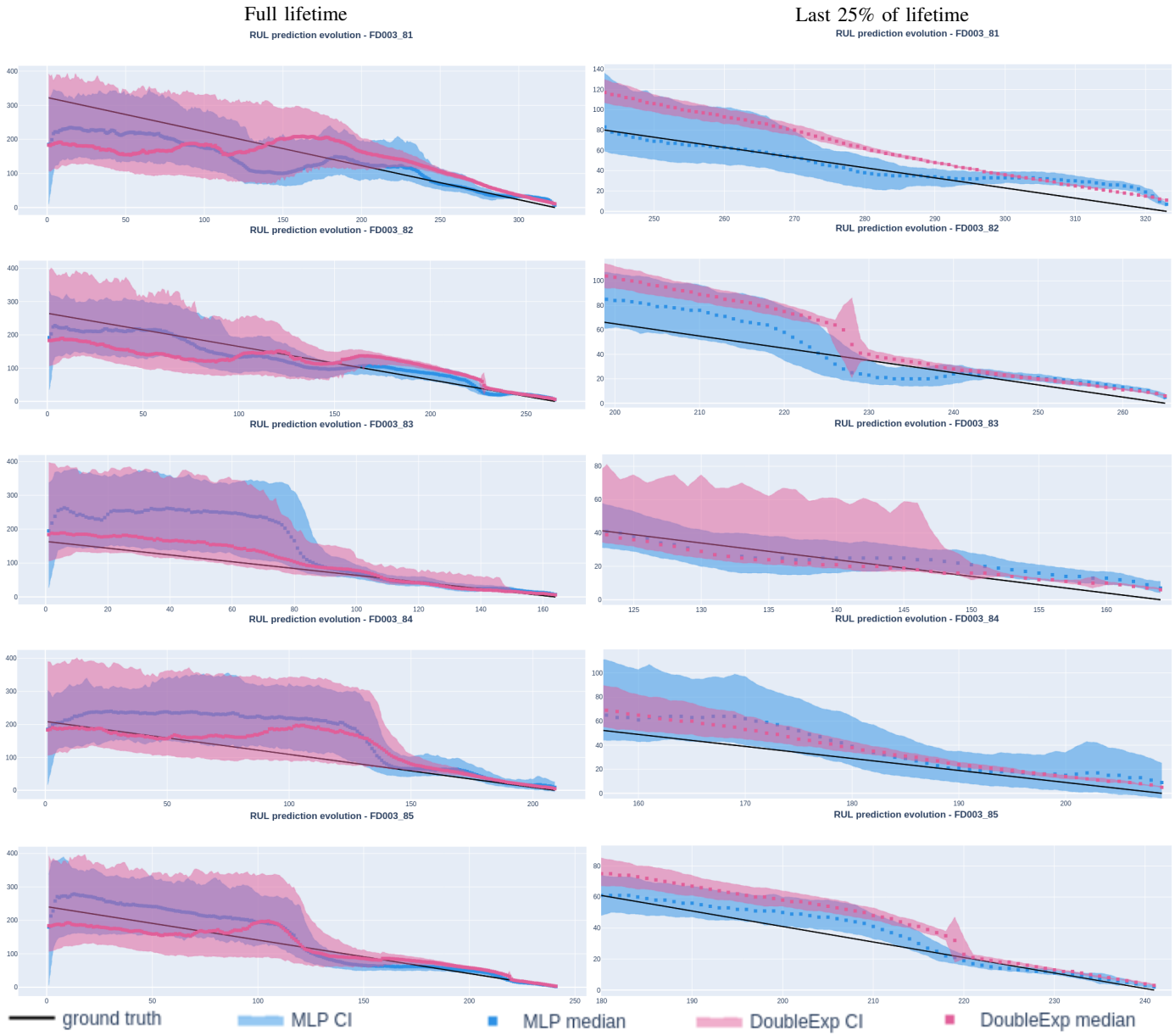

Fig. 3: On-line RUL prediction results for entire lifetime and last 25% of lifetime for turbofans 81-85

turbofans 82 and 83 it misses it. Finally when both models CI encompass the ground truth, the one issued from the MLP model is always narrower which indicates a better precision. Therefore the results observed in Fig. 3 are consistent with the indicators of Table I and indicate that MLP model is an appealing alternative to the double exponential one, most often used in the literature.

V. Conclusions

In this study an improved MLP based PF model for RUL prognostic is proposed. This specifically aims at improving the MLP architecture of the surrogate model compared to the one proposed in the literature, and also using existing robust particle exploration techniques that were never used in the context of HI prediction. The model performance is assessed and compared with the same PF model when using a sum of two exponential (DE) instead of the MLP, which is a common choice for surrogate models in the literature. It was found that the proposed model is at least as good as and mostly outperforms the DE surrogate strategy in the context of jet engine degradation. This confirms previous findings on the suitability of shallow and thin neural networks to act as surrogate models in PF, especially when the observation equation is unknown. It specifically demonstrates the flexibility of MLP to fit in different application cases, i.e. turbofans, Li-Ion batteries and crack growth where degradation trajectories are of different nature. During this work it was found that the initialization of particles, their jittering and reintroduction

Model	Relative error (full)	Relative error (75-100%)	CI coverage (full)	CI coverage (75-100%)
MLP	1120	**325**	**3956**	**1133**
Double exp.	**866**	335	3054	388

TABLE I: Performance indicators over the entire testing set

dynamics were of significant importance. Unfortunately their precise influence could not be discussed here for conciseness, however a thorough study on these parameters influence would be of particular interest. The same stands for studying the performance of MLP-based PF on other and more complex application cases such as rolling bearing. These results encourage further research on the topic of PF based RUL prognosis. More particularly investigating the possibility of using other machine learning general models instead of MLP, e.g. support vector machines (SVM).

ACKNOWLEDGMENT

The project leading to this application has received funding from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No 955393

We are grateful to the Mésocentre Clermont Auvergne University for providing computing and storage resources. All computations have been performed on the supercomputer facilities of the Mésocentre Clermont Auvergne.

REFERENCES

[1] X. Jin, Y. Sun, Z. Que, Y. Wang, and T. W. S. Chow, "Anomaly Detection and Fault Prognosis for Bearings," *IEEE Transactions on Instrumentation and Measurement*, vol. 65, no. 9, pp. 2046–2054, 2016.

[2] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff, "LSTM-based Encoder-Decoder for Multi-sensor Anomaly Detection," *ArXiv160700148 Cs Stat*, 2016.

[3] P. Baraldi, G. Bonfanti, and E. Zio, "Differential evolution-based multi-objective optimization for the definition of a health indicator for fault diagnostics and prognostics," *Mechanical Systems and Signal Processing*, vol. 102, pp. 382–400, 2018.

[4] L. Guo, N. Li, F. Jia, Y. Lei, and J. Lin, "A recurrent neural network based health indicator for remaining useful life prediction of bearings," *Neurocomputing*, vol. 240, pp. 98–109, 2017.

[5] K. Medjaher, N. Zerhouni, and J. Baklouti, "Data-driven prognostics based on health indicator construction: Application to PRONOSTIA's data," in *2013 European Control Conference (ECC)*. IEEE, 2013, pp. 1451–1456.

[6] X. Liu, P. Song, C. Yang, C. Hao, and W. Peng, "Prognostics and Health Management of Bearings Based on Logarithmic Linear Recursive Least-Squares and Recursive Maximum Likelihood Estimation," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 2, pp. 1549–1558, 2018.

[7] M. M. Luengo and A. Kolios, "Failure Mode Identification and End of Life Scenarios of Offshore Wind Turbines: A Review," *Energies*, vol. 8, no. 8, pp. 8339–8354, 2015.

[8] D.-C. Feng, Z.-T. Liu, X.-D. Wang, Z.-M. Jiang, and S.-X. Liang, "Failure mode classification and bearing capacity prediction for reinforced concrete columns based on ensemble machine learning algorithm," *Advanced Engineering Informatics*, vol. 45, p. 101126, 2020.

[9] S. G. Arunajadai, S. J. Uder, R. B. Stone, and I. Y. Tumer, "Failure Mode Identification through Clustering Analysis," *Quality and Reliability Engineering International*, vol. 20, no. 5, pp. 511–526, 2004.

[10] E. Jules, C. Mattrand, and J.-M. Bourinet, "Similarity learning for predictive maintenance: health indicator construction based on siamese neural networks and contrastive loss," *[Under Review]*, 2023.

[11] A. Widodo and B.-S. Yang, "Application of relevance vector machine and survival probability to machine degradation assessment," *Expert Systems with Applications*, vol. 38, no. 3, pp. 2592–2599, 2011.

[12] P. Loukopoulos, G. Zolkiewski, I. Bennett, S. Sampath, P. Pilidis, X. Li, and D. Mba, "Abrupt fault remaining useful life estimation using measurements from a reciprocating compressor valve failure," *Mechanical Systems and Signal Processing*, vol. 121, pp. 359–372, 2019.

[13] Y. Liu, X. Hu, and W. Zhang, "Remaining useful life prediction based on health index similarity," *Reliability Engineering & System Safety*, vol. 185, pp. 502–510, 2019.

[14] D. Chen, Y. Qin, Y. Wang, and J. Zhou, "Health indicator construction by quadratic function-based deep convolutional auto-encoder and its application into bearing RUL prediction," *ISA Transactions*, vol. 114, pp. 44–56, 2021.

[15] K. T. P. Nguyen and K. Medjaher, "An automated health indicator construction methodology for prognostics based on multi-criteria optimization," *ISA Transactions*, vol. 113, pp. 81–96, 2021.

[16] P. Wen, S. Zhao, S. Chen, and Y. Li, "A generalized remaining useful life prediction method for complex systems based on composite health indicator," *Reliability Engineering & System Safety*, vol. 205, p. 107241, 2021.

[17] Y. Qin, J. Zhou, and D. Chen, "Unsupervised Health Indicator Construction by a Novel Degradation-Trend-Constrained Variational Autoencoder and Its Applications," *IEEE/ASME Transactions on Mechatronics*, vol. 27, no. 3, pp. 1447–1456, 2022.

[18] Y. Lei, N. Li, L. Guo, N. Li, T. Yan, and J. Lin, "Machinery health prognostics: A systematic review from data acquisition to RUL prediction," *Mechanical Systems and Signal Processing*, vol. 104, pp. 799–834, 2018.

[19] S. Ansari, A. Ayob, M. S. Hossain Lipu, A. Hussain, and M. H. M. Saad, "Remaining useful life prediction for lithium-ion battery storage system: A comprehensive review of methods, key factors, issues and future outlook," *Energy Reports*, vol. 8, pp. 12 153–12 185, 2022.

[20] M. Jouin, R. Gouriveau, D. Hissel, M.-C. Péra, and N. Zerhouni, "Particle filter-based prognostics: Review, discussion and perspectives," *Mechanical Systems and Signal Processing*, vol. 72–73, pp. 2–31, 2016.

[21] N. Li, Y. Lei, T. Yan, N. Li, and T. Han, "A Wiener-Process-Model-Based Method for Remaining Useful Life Prediction Considering Unit-to-Unit Variability," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 3, pp. 2092–2101, 2019.

[22] K. Peng, R. Jiao, J. Dong, and Y. Pi, "A deep belief network based health indicator construction and remaining useful life prediction using improved particle filter," *Neurocomputing*, vol. 361, pp. 19–28, 2019.

[23] R. Jiao, K. Peng, J. Dong, and C. Zhang, "Fault monitoring and remaining useful life prediction framework for multiple fault modes in prognostics," *Reliability Engineering & System Safety*, vol. 203, p. 107028, 2020.

[24] T. Flury and N. Shephard, "Learning and filtering via simulation: Smoothly jittered particle filters," University of Oxford, Economic Series working paper 469, 12 2009.

[25] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statistics and Computing*, vol. 10, no. 3, pp. 197–208, 2000.

[26] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, 2002.

[27] N. Kantas, A. Doucet, S. S. Singh, J. Maciejowski, and N. Chopin, "On Particle Methods for Parameter Estimation in State-Space Models," *Statistical Science*, vol. 30, no. 3, 2015.

[28] de Freitas J. F. G., M. Niranjan, A. H. Gee, and A. Doucet, "Sequential Monte Carlo Methods to Train Neural Network Models," *Neural Computation*, vol. 12, no. 4, pp. 955–993, 2000.

[29] F. Cadini, C. Sbarufatti, F. Cancelliere, and M. Giglio, "State-of-life prognosis and diagnosis of lithium-ion batteries by data-driven particle filters," *Applied Energy*, vol. 235, pp. 661–672, 2019.

[30] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *IEE Proceedings F (Radar and Signal Processing)*, vol. 140, no. 2, pp. 107–113, 1993.
[31] A. Saxena, K. Goebel, D. Simon, and N. Eklund, "Damage Propagation Modeling for Aircraft Engine Prognostics," *International Conference on Prognostics and Health Management*, p. 9, 2008.