Mechanical Systems and Signal Processing 85 (2017) 827–848

Contents lists available at ScienceDirect

Mechanical Systems and Signal Processing

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

Particle-filtering-based failure prognosis via sigma-points: Application to Lithium-Ion battery State-of-Charge monitoring

David E. Acuña *, Marcos E. Orchard

Department of Electrical Engineering, Faculty of Mathematical and Physical Sciences, University of Chile, Av. Tupper 2007, Santiago, Chile

ARTICLE INFO

Article history:
Received 16 March 2016
Received in revised form
17 July 2016
Accepted 22 August 2016

Keywords:
Prognostics and health management
Uncertainty characterization
Particle filters
Battery State-of-Charge

ABSTRACT

This paper presents a novel prognostic method that allows a proper characterization of the uncertainty associated with the evolution in time of nonlinear dynamical systems. The method assumes a state-space representation of the system, as well as the availability of particle-filtering-based estimates of the state posterior density at the moment in which the prognostic algorithm is executed. Our proposal significantly improves all particle-filtering-based prognosis frameworks currently available in two main aspects. First, it provides a correction for the expression that is used for the computation of the Time-of-Failure (ToF) probability mass function in the context of online monitoring schemes. Secondly, it presents a method for improved characterization of the tails of the ToF probability mass function via sequential propagation of sigma-points and the computation of Gaussian Mixture Models (GMMs). The proposed algorithm is tested and validated using experimental data related to the problem of Lithium-Ion battery State-of-Charge prognosis.

1. Introduction

One of the most critical aspects to consider in the elaboration of decision-making processes is the manner in which uncertainty and risk are quantified. The latter, due to the impact that these uncertainty sources may have on the final cost associated with all available choices; costs that in some cases are related to the operational continuity of industrial systems. In this regard, several techniques intend to address the problem of uncertainty characterization from different points of view. Some methods follow possibility-based approaches, fuzzy logic, or evidence theory [1–4]. Nevertheless, most of researchers have preferred probability-based methods for the implementation of online failure prognostic algorithms [5], because these approaches allow including the notion of uncertainty under a well-known and accepted mathematical formalization.

In the context of probability theory, Bayesian approaches [6] arise as a suitable option for most online learning systems. Bayesian schemes allow the implementation of filtering algorithms (also known as Bayesian processors) for uncertainty characterization in nonlinear dynamic processes [7–9]. This task is performed via the computation of posterior estimates of the state vector, where both prior information (provided by the system model) and measurements (acquired in real-time) are efficiently used. In failure prognosis, these states typically correspond to critical variables whose future evolution in time might significantly affect the system health, thus yielding into a failure condition. However, except for a reduced number of

* Corresponding author.
E-mail address: davacuna@ing.uchile.cl (D.E. Acuña).

http://dx.doi.org/10.1016/j.ymssp.2016.08.029
0888-3270/

cases, it is not possible to obtain an analytic solution for the filtering problem. A well-known solution for linear Gaussian systems is the Kalman filter (KF) [10,11], which provides an optimal estimate in the mean-square error sense. Although some nonlinear systems may be approximated by linearized versions within bounded regions of the state space, KF implementations will generally offer a suboptimal solution. Moreover, underlying uncertainties may distribute differently from a Gaussian Probability Density Function (PDF). In this regard, Sequential Monte Carlo (SMC) methods, a.k.a. Particle Filters (PFs) [6], provide an interesting solution for the filtering problem in the context of nonlinear non-Gaussian systems. PFs approximate the underlying PDF by a set of weighted samples, recursively updated, yielding an empirical distribution from where statistical inference is performed.

Although PFs allow characterizing uncertainty associated with the state vector in filtering problems, our interest in prognosis is really focused on describing the future evolution of uncertainty [12]; all to take preventive measures and avoid failures that may lead to catastrophic events [13] that could affect system interoperability. Monte Carlo (MC) simulation [14] offers a solution to this problem. However, the computational cost associated with MC simulation complicates the implementation of real-time MC-based prognostic schemes. In this regard, the Prognostic and Health Management (PHM) community has chosen particle-filtering-based algorithms [15] as the state-of-the-art in this matter, since those algorithms offer an efficient alternative to Monte Carlo simulation in the implementation of real-time prognosis approaches.

Our research focuses on two aspects that help to significantly improve all particle-filtering-based prognosis frameworks currently available in literature. Firstly, it provides a correction for the expression that is typically used for the computation of the Time-of-Failure (ToF) Probability Mass Function (PMF) in the context of online monitoring schemes; providing a better understanding of the risk associated with future operation and extending the ToF PMF definition to general nonlinear systems that may present regenerative behavior. Secondly, it presents a novel method for improved characterization of the tails of the ToF PMF via sequential propagation of sigma-points [16–18] (a set of deterministically defined weighted points that preserve the first two moments of a PDF) and the computation of Gaussian Mixture Models (GMMs). The proposed algorithm is tested and validated using experimental data related to the problem of battery State-of-Charge prognosis.

The structure of the paper is as follows. Section 2 presents theoretical background on SMC methods and sigma-points. Section 3 presents a modification to the expression that is used to compute the ToF PMF. Section 4 shows a general description of the proposed prognostic algorithm and, subsequently, Section 5 shows a validation case study on Lithium-Ion battery State-of-Charge prognosis, a.k.a. End-of-Discharge prognosis. Section 6 shows conclusions and future work.

2. Theoretical background

This section aims at providing a proper theoretical framework on some of the concepts that are included in the design of our proposed prognostic algorithm, as well as motivating the intuition of the reader.

2.1. Particle filters

Particle Filters are computational methods designed for recursively estimating the evolving posterior distribution of a nonlinear, non-Gaussian, dynamic system, given a set of sequentially acquired measurements (problem also known as *optimal filtering*).

Let $X = \{X_k,\, k \in \mathbb{N}\}$ be a first order Markov process denoting a n_x-dimensional system state vector with initial distribution $p(x_0)$ and transition probability $p(x_k|x_{k-1})$. Also, let $Y = \{Y_k,\, k \in \mathbb{N}\backslash\{0\}\}$ denote n_y-dimensional conditionally independent noisy observations. PFs assume a state-space representation of the dynamic nonlinear system:

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

$$y_k = g(x_k,\, v_k), \tag{2}$$

where ω_k and v_k denote independent non-Gaussian random variables. The objective is to estimate the posterior distribution $p(x_k|y_{1:k})$. As this is a difficult task to achieve, estimators require the implementation of structures based on Bayes' rule where, under Markovian assumptions, the filtering posterior distribution can be written as

$$p(x_k|y_{1:k}) = \frac{p(y_k|x_k)p(x_k|y_{1:k-1})}{p(y_k|y_{1:k-1})}. \tag{3}$$

Sequential Monte Carlo methods (SMC), a.k.a. PFs, efficiently simulate complex stochastic systems to approximate the posterior PDF (3) by a collection of N_p weighted samples or *particles* $\{x_k^{(i)},\, W_k^{(i)}\}_{i=1}^{N_p}$, $\sum_{i=1}^{N_p} W_k^{(i)} = 1$, such that:

$$\hat{p}(x_k|y_{1:k}) \approx \sum_{i=1}^{N_p} W_k^{(i)} \delta_{x_k^{(i)}}(x_k). \tag{4}$$

The weighting process is performed by applying *sequential importance resampling* (SIR) algorithms in two stages: Sequential Importance Sampling and Resampling.

2.1.1. Sequential importance sampling

Assume that it is of interest to compute the expectation

$$\hat{f}(x_k) = E_{X|Y}\{f(x_k)\} = \int_{\Re^{n_x}} f(x_k)p(x_k|y_{1:k})dx_k. \tag{5}$$

If N_p independent identically distributed random samples are drawn from $p(x_k|y_{1:k})$, (5) is then approximated by:

$$\hat{f}(x_k) \approx \int_X f(x_k)\left(\frac{1}{N_p}\sum_{i=1}^{N_p}\delta_{x_k^{(i)}}(x_k)\right)dx_k \tag{6}$$

$$\hat{f}(x_k) \approx \frac{1}{N_p}\sum_{i=1}^{N_p} f(x_k)\delta_{x_k^{(i)}}(x_k) \tag{7}$$

$$\hat{f}(x_k) \approx \frac{1}{N_p}\sum_{i=1}^{N_p} f\left(x_k^{(i)}\right) \tag{8}$$

Typically, it is difficult to directly sample from $p(x_k|y_{1:k})$. *Sequential Importance Sampling* (SIS) avoids the aforementioned issue by drawing samples from an alternative distribution called *importance distribution*, $\pi(\tilde{x}_k^{(i)}|\tilde{x}_{0:k-1}^{(i)}, y_{1:k})$. SIS proposes to update the weights of each particle as

$$w_k^{(i)} = W_{k-1}^{(i)} \cdot \frac{p(y_k|\tilde{x}_k^{(i)})p(\tilde{x}_k^{(i)}|x_{k-1}^{(i)})}{\pi(\tilde{x}_k^{(i)}|\tilde{x}_{0:k-1}^{(i)}, y_{1:k})}, \tag{9}$$

where $\{\tilde{x}_k^{(i)}\}_{i=1}^{N_p}$ is a set of N_p random samples drawn from $\pi(\cdot|\tilde{x}_{0:k-1}^{(i)}, y_{1:k})$. Moreover, defining normalized weights

$$W_k^{(i)} = \frac{w_k^{(i)}}{\sum_{i=1}^{N_p} w_k^{(i)}}, \tag{10}$$

then the posterior distribution can be approximated by Eq. (4).

2.1.2. Resampling

In the SIS framework, the variance of particle weights tends to increase as time goes on. Moreover, most of the probability mass ends concentrating in only a few samples. This problem, known as *sample degeneracy*, is addressed by including a *resampling* step, leading to the *Sequential Importance Resampling* (SIR) algorithm.

A measure of particle degeneracy is provided in terms of the *effective particle sample size*; see Eq. (11):

$$\hat{N}_{eff}(k) = \frac{1}{\sum_{i=1}^{N_p}(W_k^{(i)})^2} \tag{11}$$

The resampling step [19,20] samples the particle population using probabilities that are proportional to the particle weights. Resampling is applied if $\hat{N}_{eff} \le N_{thres}$, with N_{thres} a fixed threshold.

2.2. Outer feedback correction loops

Artificial evolution [13,21] is a method used for unknown model parameter estimation that proposes the augmentation of the state vector of the system. The state equations related to unknown model parameters assume random walk models to implement learning procedures that allow simultaneous estimation of parameters and original state variables of the system. *Outer Feedback Correction Loops* (OFCL) [22] are a class of algorithms that propose to manipulate the hyper-parameters that characterize the aforementioned random walk models. For example, let us consider the system:

$$x_{k+1} = g(x_k, \theta, \omega_k) \tag{12}$$

where x_k is the state vector, θ is the parameter vector and ω_k is process noise. If $\theta = \begin{pmatrix}\theta^c \\ \theta^{uc}\end{pmatrix}$, where θ^{uc} is a vector of uncertain parameters, artificial evolution proposes to consider the system

$$x_{k+1} = g(x_k, \theta_k, \omega_k) \tag{13}$$

$$\theta_{k+1}^{uc} = \theta_k^{uc} + \omega_k^{\theta} \tag{14}$$

where ω_k^{θ} is a random variable. OFCL can modify the hyper-parameters of ω_k^{θ} to gradually reduce the uncertainty associated to θ^{uc}.

2.3. Unscented transform and sigma-points

The *Unscented Transform* (UT) [16,17] is a method designed for approximating nonlinear transformations of random variables. Although some nonlinear functions can be approximated by truncating Taylor expansions, UT focuses on preserving some of the statistics related to the nonlinear transformation, using sets of deterministic samples called *sigma-points*.

Let us suppose $x \in \Re^{n_x}$ is a random variable with mean $\bar{x}$ and covariance matrix P_x, and $g(\cdot)$ is a nonlinear real-valued function. The objective is to approximate

$$y = g(x). \tag{15}$$

The UT method states the following. Firstly, compute a set of $2n_x + 1$ deterministically chosen weighted samples, namely the sigma-points $\mathcal{S}_i = \{\mathcal{X}^{(i)}, \mathcal{W}^{(i)}\}$, $i = 0, \ldots, 2n_x$, such that

$$\mathcal{X}^{(0)} = \bar{x} \quad i = 0 \tag{16}$$

$$\mathcal{X}^{(i)} = \bar{x} + \left(\sqrt{(n_x + \kappa)\cdot P_x}\right)_i \quad i = 1, \ldots, n_x \tag{17}$$

$$\mathcal{X}^{(i)} = \bar{x} - \left(\sqrt{(n_x + \kappa)\cdot P_x}\right)_i \quad i = n_x + 1, \ldots, 2n_x \tag{18}$$

where

$$\mathcal{W}^{(0)} = \frac{\kappa}{n_x + \kappa} \quad i = 0 \tag{19}$$

$$\mathcal{W}^{(i)} = \frac{1}{2(n_x + \kappa)} \quad i = 1, \ldots, 2n_x. \tag{20}$$

The variable i indexes the deterministic weighted samples $\{\mathcal{X}^{(i)}, \mathcal{W}^{(i)}\}_{i=0}^{2n_x}$ that approximate the first two moments of x as

$$\bar{x} = \sum_{i=0}^{2n_x} \mathcal{W}^{(i)} \mathcal{X}^{(i)},$$

$$P_x = \sum_{i=0}^{2n_x} \mathcal{W}^{(i)} (\mathcal{X}^{(i)} - \bar{x})(\mathcal{X}^{(i)} - \bar{x})^T.$$

Weights satisfy $\sum_{i=0}^{2n_x} \mathcal{W}^{(i)} = 1$. On the other hand, $\kappa > 0$ is a scaling parameter which determines how far from the mean sigma-points are set.

Applying the nonlinear transformation on each sigma-point

$$\mathcal{Y}^{(i)} = g(\mathcal{X}^{(i)}) \quad i = 0, \ldots, 2n_x, \tag{21}$$

then the statistics of y can be approximated as:

$$\bar{y} \approx \sum_{i=0}^{2n_x} \mathcal{W}^{(i)} \mathcal{Y}^{(i)}, \tag{22}$$

$$P_y \approx \sum_{i=0}^{2n_x} \mathcal{W}^{(i)} (\mathcal{Y}^{(i)} - \bar{y})(\mathcal{Y}^{(i)} - \bar{y})^T, \tag{23}$$

$$P_{xy} \approx \sum_{i=0}^{2n_x} \mathcal{W}^{(i)} (\mathcal{X}^{(i)} - \bar{x})(\mathcal{Y}^{(i)} - \bar{y})^T. \tag{24}$$

As it can be seen in expressions (22), (23) and (24), the variable i now indexes the transformed weighted samples that define the mean, covariance, and cross-covariance matrices of the random variable y, respectively.

UT provides a methodology that at least preserves accurately the first two moments of the associated distributions.

2.4. Weighted expectation maximization for GMMs

Expectation Maximization (EM) [23] is an iterative algorithm for finding maximum-likelihood estimates of parameters in statistical models with incomplete data. Each iteration of the algorithm performs two steps. In the first step, *E*-step, the expectation of the likelihood (denoted by Q) is evaluated using currently available parameter estimates. The second step, *M*-step, modifies parameter estimates by maximizing the previously computed expectation. This process is repeated until convergence.

Let us assume $\mathcal{Z} = (\mathcal{X}, \mathcal{Y})$, where $\mathcal{Z}$ denotes a complete data set, $\mathcal{X}$ is observed data, and $\mathcal{Y}$ are unobserved data. Hence, if data is obtained from a PDF whose parameters are denoted by Θ, then

$$p(z|\Theta) = p(x, y|\Theta) = p(y|x, \Theta)p(x|\Theta). \tag{25}$$

The *E*-step consist of evaluating the deterministic function

$$Q(\Theta, \Theta^{(i-1)}) = \mathbb{E}[\log(p(\mathcal{Y}|\mathcal{X}, \Theta))|\mathcal{X}, \Theta^{(i-1)}], \tag{26}$$

where $\mathcal{X} = x$ is known and $\Theta^{(i-1)}$ is the maximum likelihood parameter estimate of the previous iteration. Note that as $\mathcal{Y}$ is unobservable, it remains as a random variable; however Q becomes a deterministic function of Θ. Next, $Q(\Theta, \Theta^{(i-1)})$ is maximized with respect to Θ in the *M*-step, resulting in

$$\Theta^{(i)} = \underset{\Theta}{\operatorname{argmax}} Q(\Theta, \Theta^{(i-1)}). \tag{27}$$

This research effort uses the EM algorithm in kernel density estimation, particularly to fit GMMs due to their ability to approximate any PDF with accuracy directly related to the number of components in the mixture. In this regard, $p(x|\Theta)$ can be expressed as

$$p(x|\Theta) = \sum_{j=1}^{M} \alpha_j p_j(x|\mu_j, \Sigma_j) \tag{28}$$

with $\alpha_j > 0$ the mixture weights such that $\sum_{j=1}^{M} \alpha_j = 1$, and

$$p_j(x|\mu_j, \Sigma_j) = \frac{1}{(2\pi)^{d/2}|\Sigma_j|^{1/2}} e^{-\frac{1}{2}(x-\mu_j)^T \Sigma_j^{-1}(x-\mu_j)} \tag{29}$$

where μ_j and Σ_j are the mean vector and matrix covariance of the j-th component of the mixture. Therefore, the parameters of the GMM distribution are defined as $\Theta = (\alpha_1, \ldots, \alpha_M, \mu_1, \ldots, \mu_M, \Sigma_1, \ldots, \Sigma_M)$.

Considering a set of observations $x = (x_1, \ldots, x_N)$, and

$$h_j^{(i)}(x_k) = \frac{\alpha_j^{(i)} p_j(x_k|\mu_j^{(i)}, \Sigma_j^{(i)})}{p(x_k|\Theta^{(i)})} \tag{30}$$

with $p(x_k|\Theta^{(i-1)}) = \sum_{j=1}^{M} \alpha_j^{(i-1)} p_j(x_k|\mu_j^{(i-1)}, \Sigma_j^{(i-1)})$, the application of the EM algorithm in this case yields the following update of the parameters for the i-th iteration:

$$\alpha_j^{(i)} = \frac{1}{N} \sum_{k=1}^{N} h_j^{(i-1)}(x_k) \tag{31}$$

$$\mu_j^{(i)} = \frac{\sum_{k=1}^{N} x_k h_j^{(i-1)}(x_k)}{\sum_{k=1}^{N} h_j^{(i-1)}(x_k)} \tag{32}$$

$$\Sigma_j^{(i)} = \frac{\sum_{k=1}^{N} (x_k - \mu_j^{(i)})(x_k - \mu_j^{(i)})^T h_j^{(i-1)}(x_k)}{\sum_{k=1}^{N} h_j^{(i-1)}(x_k)}. \tag{33}$$

In particular, we use in our implementation a weighted version of the EM algorithm called *Weighted Expectation Maximization* (WEM) [23,24]. The only modifications in WEM to Eqs. (31)–(33) are associated with the manner in which $p_j(x_k|\mu_j^{(i-1)}, \Sigma_j^{(i-1)})$ is computed: the right side of Eq. (29) must include a multiplication by the weight related to x_k.

3. Prognosis and time-of-failure estimation

Even though real-time health monitoring schemes may provide information on the system condition via the estimation of critical variables, it is only natural to seek a characterization of the future evolution of the system once a fault has been detected. This knowledge eases decision-making processes, generating more reliable results, and allowing to take measures that aim at extending the remaining useful life (RUL) or, alternatively, the end of life (EoL) of the system [25]. Both concepts are related as:

$$RUL = EoL - k_p, \tag{34}$$

where it is assumed that the system is healthy at least at the k_p-th instant, moment in which prognostics routines are executed. However, instead of talking about RUL or EoL, concepts closely related to degradation problems, we prefer to use the concept of Time-of-Failure (ToF): the time instant where a failure takes place. The ToF concept is more general and suitable for a wider range of applications.

ToF estimation is relevant as it provides information about the operational continuity of a system, i.e., it allows to know for how much time the system can be operated before incurring a failure that could be catastrophic. Given the complexity of

industrial systems, in general it is not possible to calculate the ToF with arbitrary precision, and thus it is relevant to properly characterize the uncertainty associated with ToF estimates. According to [26], ToF prediction methods can be broadly classified in two groups: online and offline schemes. Offline methods typically make use of computationally expensive algorithms that offer highly accurate (or precise) results. In contrast, online methods should offer a delicate balance between efficiency and efficacy. In [26] some examples of offline methodologies are mentioned for the characterization of uncertainty related to crack growth dynamics [27,28], structural damage [29,30], electronics [31], and mechanical bearings [32]. Examples for the implementation of online prognosis and RUL estimation schemes can be found in [12]. Also, various efforts aimed at quantifying uncertainty associated with RUL (or ToF) estimates are found for the problem of energetic autonomy and capacity degradation of Lithium-Ion batteries [13,33–36], and degradation of pneumatic valves [37]. Methodologies inspired on efficient sampling techniques [38] and analytical methods [39] have also been recently explored.

PF-based prognostic algorithms use approximations of the true state PDF based on empirical distributions; i.e., the state PDF is characterized by a collection weighted samples. Specifically, in [13,35,36] ToF estimates are computed using a set of N_p particles (each particle being a duple $\{x_k^{(i)}, W_k^{(i)}\}$) such that

$$\mathcal{P}(ToF = k) = \sum_{i=1}^{N_p} W_k^{(i)} p(failure|X = x_k^{(i)}), \tag{35}$$

where $p(failure|X)$ corresponds to the probability of system failure, conditional to a value of the state vector $x \in \Re^{n_x}$ (value of "fault dimension" if the concept of *hazard zone* [40,25] is used instead). In fact, Eq. (35) is a particular case of

$$\mathcal{P}(ToF = k) = \int_{\Re^{n_x}} p(failure|x_k) p(x_k|y_{1:k_p}) dx_k, \tag{36}$$

where the probability distribution of the state, conditional to measurements $y_{1:k_p}$, is approximated by: $p(x_k|y_{1:k_p}) \approx \sum_{i=1}^{N_p} W_k^{(i)} \delta_{x_k^{(i)}}(x_k)$.

It is noteworthy that in the case of systems endowed by critical variables that characterize their health (typically monotonically increasing, or decreasing, functions of the fault severity), the probability measure previously shown in Eqs. (35) and (36) has been misinterpreted as a *Cumulative Mass Function* (CMF). The latter, possibly because when all critical variables are monotonically increasing (or decreasing) functions of time, the probability measure will also be quasi-monotonic; increasing from zero and reaching its maximum value at one. This interpretation therefore is limited to cases of strictly degenerative systems. However in the general case, as the probability measure for ToF is defined at discrete time instants, this concept must be reinterpreted as a *Probability Mass Function* (PMF).

In this paper, we propose a correction for the calculation of $\mathcal{P}(ToF = k)$, $k > k_p$ and where k_p is the current time instant, which satisfies the conditions for a probability measure. Furthermore, the proposal extends the application to general systems, including all possible regenerative phenomena.

3.1. Mathematical formalization

A failure event can be treated as a non-stationary Bernoulli stochastic process; i.e., a Bernoulli process in which probabilities vary as the time evolves. Denoting healthy and faulty systems (at the k-th time instant) by $\mathcal{F}_k$ and $\mathcal{H}_k$, respectively, we can define a probability space $(\Omega_{fail}, \mathcal{B}_{fail}, \mathcal{P}_{fail})$, where

- $\Omega_{fail} = \{(\mathcal{H}_{k_p}, \mathcal{H}_{k_p+1}, \ldots, \mathcal{H}_{k-1}, \mathcal{F}_k) | k \in \mathbb{N}, k > k_p\}$ is the sample space that determines all possible sequences where the system remains healthy until the time instant "k", when it actually fails,
- $\mathcal{B}_{fail} \subseteq 2^{\Omega_{fail}}$ is a $\sigma - algebra$,
- $\mathcal{P}_{fail}$ is a function that assigns a probability measure to every possible event in the $\sigma - algebra$ $\mathcal{B}_{fail}$.

Given the above, it is possible to characterize the true probability of failure at the k-th time instant, $\mathcal{P}(\mathcal{F}_k)$. Indeed, denoting $\mathcal{H}_{k_p:k} = (\mathcal{H}_{k_p}, \mathcal{H}_{k_p+1}, \ldots, \mathcal{H}_k)$, and according to the definition of conditional probability, it follows that:

$$\mathcal{P}(\mathcal{F}_k) = \frac{\mathcal{P}(\mathcal{F}_k, \mathcal{H}_{k_p:k-1})}{\mathcal{P}(\mathcal{H}_{k_p:k-1}|\mathcal{F}_k)}, \quad \forall\ k > k_p \tag{37}$$

since $\mathcal{P}(\mathcal{H}_{k_p:k-1}|\mathcal{F}_k)$ corresponds to the probability of staying healthy until time $k-1$, given that the failure occurred at time k, it is important to note that $\mathcal{P}(\mathcal{H}_{k_p:k-1}|\mathcal{F}_k) = 1$ (it is assumed that the system can only fail once).

Applying the definition of joint probability, Eq. (37) can be rewritten as

$$\mathcal{P}(\mathcal{F}_k) = \mathcal{P}(\mathcal{F}_k|\mathcal{H}_{k_p:k-1}) \mathcal{P}(\mathcal{H}_{k_p:k-1}), \quad \forall\ k > k_p \tag{38}$$

It is important to note that:

- $\mathcal{P}(\mathcal{F}_k|\mathcal{H}_{k_p:k-1})$ corresponds to the failure probability measure that has been used in the literature so far, equivalent to the

expression described in Eq. (36); i.e.,

$$\mathcal{P}(\mathcal{F}_k|\mathcal{H}_{k_p:k-1}) = \int_{\Re^{n_x}} p(failure|x_k)p(x_k|y_{1:k_p})dx_k. \tag{39}$$

Prognostic algorithms currently used in the context of online monitoring characterize $p(x_k|y_{1:k_p})$ by the propagation of uncertainty to future moments without considering any constraint other than the dynamics of the state. Indeed, when propagating the uncertainty one step ahead in the future, those algorithms implicitly assume that the system was healthy during the previous time instant.

It is noteworthy that $\mathcal{P}(\mathcal{F}_k|\mathcal{H}_{k_p:k-1})$ can be interpreted as the expectation of $p(failure|x_k)$ with respect the probability measure $p(x_k|y_{1:k_p})$.

- $\mathcal{P}(\mathcal{H}_{k_p:k-1})$ is the probability that the system is healthy until the $(k-1)$-th time instant. In consequence,

$$\begin{aligned}\mathcal{P}(\mathcal{H}_{k_p:k-1}) &= \mathcal{P}(\mathcal{H}_{k-1}|\mathcal{H}_{k_p:k-2})\mathcal{P}(\mathcal{H}_{k_p:k-2})\\ &= \mathcal{P}(\mathcal{H}_{k-1}|\mathcal{H}_{k_p:k-2})\mathcal{P}(\mathcal{H}_{k-2}|\mathcal{H}_{k_p:k-3})\mathcal{P}(\mathcal{H}_{k_p:k-3})\\ &\vdots\\ &= \prod_{j=k_p+1}^{k-1}\mathcal{P}(\mathcal{H}_j|\mathcal{H}_{k_p:j-1})\end{aligned} \tag{40}$$

Then, as $\mathcal{P}(\mathcal{H}_j|\mathcal{H}_{k_p:j-1}) = 1 - \mathcal{P}(\mathcal{F}_j|\mathcal{H}_{k_p:j-1})$ and the failure is modelled through a Bernoulli stochastic process, it follows that:

$$\mathcal{P}(\mathcal{H}_{k_p:k-1}) = \prod_{j=k_p+1}^{k-1}\left(1 - \mathcal{P}(\mathcal{F}_j|\mathcal{H}_{k_p:j-1})\right) \tag{41}$$

$$\mathcal{P}(\mathcal{H}_{k_p:k-1}) = \prod_{j=k_p+1}^{k-1}\left(1 - \int_{\Re^{n_x}} p(failure|x_j)p(x_j|y_{1:k_p})dx_j\right). \tag{42}$$

The failure probability described in Eq. (38) is defined as the product of $\mathcal{P}(\mathcal{F}_k|\mathcal{H}_{k_p:k-1})$ and $\mathcal{P}(\mathcal{H}_{k_p:k-1})$, where the first term corresponds to the likelihood of failure at k-th time (assuming that the system was healthy for all previous moments). The second term indicates the probability that the system was actually healthy until the $(k-1)$-th time instant.

For completeness, we show here that the probability measure $\mathcal{P}(\mathcal{F}_k)$, given in Eq. (38), effectively holds all properties for a probability measure, as a function of time. As a first step, we demonstrate the following lemma.

Lemma 1. *If* $\mathcal{P}(\mathcal{H}_{k_p:k}) = \prod_{j=k_p+1}^{k}\left(1 - \mathcal{P}(\mathcal{F}_j|\mathcal{H}_{k_p:j-1})\right)$, $k > k_p$, *then*

$$\sum_{k=k_p+1}^{n}\mathcal{P}(\mathcal{F}_k|\mathcal{H}_{k_p:k-1})\mathcal{P}(\mathcal{H}_{k_p:k-1}) = 1 - \mathcal{P}(\mathcal{H}_{k_p:n}) \tag{43}$$

Proof.

$$\begin{aligned}&\mathcal{P}(\mathcal{H}_{k_p:k}) = \prod_{j=k_p+1}^{k}\left(1 - \mathcal{P}(\mathcal{F}_j|\mathcal{H}_{k_p:j-1})\right)\\ \Leftrightarrow\quad &\mathcal{P}(\mathcal{H}_{k_p:k}) = \left(1 - \mathcal{P}(\mathcal{F}_k|\mathcal{H}_{k_p:k-1})\right)\mathcal{P}(\mathcal{H}_{k_p:k-1})\\ \Leftrightarrow\quad &\mathcal{P}(\mathcal{F}_k|\mathcal{H}_{k_p:k-1})\mathcal{P}(\mathcal{H}_{k_p:k-1}) = \mathcal{P}(\mathcal{H}_{k_p:k-1}) - \mathcal{P}(\mathcal{H}_{k_p:k})\\ \Rightarrow\quad &\sum_{k=k_p+1}^{n}\mathcal{P}(\mathcal{F}_k|\mathcal{H}_{k_p:k-1})\mathcal{P}(\mathcal{H}_{k_p:k-1}) = \sum_{k=k_p+1}^{n}\mathcal{P}(\mathcal{H}_{k_p:k-1}) - \mathcal{P}(\mathcal{H}_{k_p:k})\\ \Leftrightarrow\quad &\sum_{k=k_p+1}^{n}\mathcal{P}(\mathcal{F}_k|\mathcal{H}_{k_p:k-1})\mathcal{P}(\mathcal{H}_{k_p:k-1}) = \mathcal{P}(\mathcal{H}_{k_p}) - \mathcal{P}(\mathcal{H}_{k_p:n})\end{aligned}$$

$\mathcal{P}(\mathcal{H}_{k_p}) = 1$, because it is known beforehand that the system is healthy at the time when the prognostic algorithm is executed. Thus, it follows that

$$\Rightarrow \sum_{k=k_p+1}^{n}\mathcal{P}(\mathcal{F}_k|\mathcal{H}_{k_p:k-1})\mathcal{P}(\mathcal{H}_{k_p:k-1}) = 1 - \mathcal{P}(\mathcal{H}_{k_p:n}).$$

□

Now we show that the probability measure $\mathcal{P}(\mathcal{F}_k)$ is well defined.

Theorem 2. *Considering the definition of $\mathcal{P}(\mathcal{F}_k)$ shown in Eq. (38), it holds that*

1. $0 \leq \mathcal{P}(\mathcal{F}_k) \leq 1, \forall\, k \in \mathbb{Z}$
2. $\mathcal{P}(\mathcal{F}_k) = 0, \forall\, k \leq k_p$
3. $\{\mathcal{P}(\mathcal{F}_k)\}_{k\in\mathbb{Z}} \in \ell^1$. *Moreover,* $\sum_{k=-\infty}^{\infty} \mathcal{P}(\mathcal{F}_k) = 1$ *either if it fails in finite time or if the support of the PDF of the system states have non-empty intersection with the hazard zone to infinity.*

Proof.

1. It follows from the Law of Total Probability.
2. If follows from the fact that failure prognostic algorithms solely provide useful information when utilized to predict Time-of-Failure (and thus, it is assumed that the system has not failed yet). If a failure had already occurred in the system, then real-time failure prognosis would be an ill defined problem.
3. Provided that

$$\begin{aligned}\sum_{k=-\infty}^{\infty} |\mathcal{P}(\mathcal{F}_k)| &\overset{(Theorem\ 2.1)}{=} \sum_{k=-\infty}^{\infty} \mathcal{P}(\mathcal{F}_k)\\ &\overset{(Theorem\ 2.2)}{=} \sum_{k=k_p+1}^{\infty} \mathcal{P}(\mathcal{F}_k)\\ &= \sum_{k=k_p+1}^{\infty} \mathcal{P}(\mathcal{F}_k|\mathcal{H}_{k_p:k-1})\mathcal{P}(\mathcal{H}_{k_p:k-1})\\ &= \lim_{n\to\infty} \sum_{k=k_p+1}^{n} \mathcal{P}(\mathcal{F}_k|\mathcal{H}_{k_p:k-1})\mathcal{P}(\mathcal{H}_{k_p:k-1})\\ &\overset{(Lemma\ 1)}{=} 1 - \lim_{n\to\infty} \mathcal{P}(\mathcal{H}_{k_p:n}),\end{aligned}$$

   then $\{\mathcal{P}(\mathcal{F}_k)\}_{k\in\mathbb{Z}} \in \ell^1$ as $\mathcal{P}(\mathcal{H}_{k_p:n}) \leq 1, \forall\, n > k_p$. Finally, considering the existence of $n \in \mathbb{Z}$ big enough to ensure that the support of $p(x_k|y_{1:k_p})$ intersects the support of $p(failure|x_k)$ for a sufficiently extended time period, it follows that:

$$\lim_{n\to\infty} \mathcal{P}(\mathcal{H}_{k_p:n}) = 0.$$

   The above equation holds trivially when the system RUL is bounded in time, a fact that can be safely assumed for all practical purposes.□

In the case of degenerative systems; i.e., those which health decreases monotonically in time (without presenting regeneration phenomena), it is expected to undergo system failure in a finite amount of time. Even in cases when the system could remain healthy for a extended lapse, for example when it exhibits regenerating phenomena, the probability of failure after a large amount of time should be close to zero. Indeed,

$$\begin{aligned}\lim_{k\to\infty} \mathcal{P}(\mathcal{F}_k) &= \lim_{k\to\infty} \mathcal{P}(\mathcal{F}_k|\mathcal{H}_{k_p:k-1})\mathcal{P}(\mathcal{H}_{k_p:k-1}) = \lim_{k\to\infty} \mathcal{P}(\mathcal{F}_k|\mathcal{H}_{k_p:k-1}) \prod_{j=k_p+1}^{k-1} \mathcal{P}(\mathcal{H}_j|\mathcal{H}_{k_p:j-1})\\ &= \lim_{k\to\infty} \underbrace{\mathcal{P}(\mathcal{F}_k|\mathcal{H}_{k_p:k-1})}_{\leq 1} \prod_{j=k_p+1}^{k-1} \left(1 - \mathcal{P}(\mathcal{F}_j|\mathcal{H}_{k_p:j-1})\right) \leq \lim_{k\to\infty} \prod_{j=k_p+1}^{k-1} \left(1 - \mathcal{P}(\mathcal{F}_j|\mathcal{H}_{k_p:j-1})\right)\\ &\leq \lim_{k\to\infty} \left(\underbrace{\max_{k_p<j<k} \left(1 - \mathcal{P}(\mathcal{F}_j|\mathcal{H}_{k_p:j-1})\right)}_{<1,\ \text{as it does not fail in finite time}} \right)^{k-k_p+1} = 0\end{aligned}$$

4. Proposed prognostic algorithm: propagation of uncertainty via sigma-points

Real-time evaluation of (38) requires proper characterization of the manner in which the uncertainty associated with the state vector propagates in time. In this regard, prognostic algorithms based on PFs, such as the one developed in [13], have repeatedly been used as the method of choice in different applications [41] including damage growth prediction [42], failure analysis of analog electronic circuits [43], fault-tolerant component design [44], failure analysis of electrical machines [45], and battery State-of-Charge and State-of-Health prognosis [34–36,46], among others [47]. However, this class of algorithm is not without problems. Firstly, it does not provide a characterization of exogenous inputs that may affect long-term predictions. Second, PF-based prognostic approaches typically use a fixed number of (sometimes equally weighted) particles randomly placed in the space-state, and it is difficult to ensure that those particles will suffice to properly characterize the tails of future state PDFs as the prediction horizon grows. These aspects motivate the development of new prognostic algorithms that could overcome these limitations.

This paper proposes a novel methodology for probabilistic characterization of the uncertainty associated with the temporal evolution of nonlinear dynamic systems, within the framework defined by failure prognosis problems. The method assumes a state-space representation of the system given by:

$$x_{k+1} = f(x_k, u_k, \omega_k) \tag{44}$$

$$y_k = g(x_k, u_k, \eta_k), \tag{45}$$

where $x_k \in \Re^{n_x}$ is the state vector, $u_k \in \Re^m$ is an exogenous input, $y_k \in \Re^p$ is the output, $\omega_k \in \Re^n$ and $\eta_k \in \Re^q$ are random variables respectively corresponding to process and measurement noise sources, and $f(\cdot)$ and $g(\cdot)$ are real-valued nonlinear functions. Requirements to be considered in the design of the proposed prognostic method include that:

(i) It must be applicable to nonlinear dynamical systems.
(ii) It should be computationally tractable for real-time applications.
(iii) It must offer approximations invariant to actual state PDFs.
(iv) It must consider the probabilistic characterization of exogenous inputs.

Since PFs are the *de facto* state-of-art for statistical inference in nonlinear systems within the PHM community, our proposed algorithm addresses the first requirement by using the weighted samples provided by PFs as inputs of the prognosis module. As the propagation of the uncertainty in time could indeed generate complex PDFs, both the second and third requirements represent difficult challenges. Last but not least, the fourth condition allows to include different possible scenarios for the future operation of the system.

Prognostic algorithms must be able to characterize future evolution of uncertainty associated with the state vector, solely using information collected from system outputs up to the present time. If we assume exogenous inputs to be known, or at least probabilistically characterized, the problem of uncertainty propagation should in theory be solved via Monte Carlo simulation and systematic application of the state transition Eq. (44). However, real-time computation of the predicted state PDF requires a significant computational effort when dealing with nonlinear systems and non-Gaussian process noise. To solve this issue, our approach proposes an approximation of the predicted PDF using sequential propagation of sigma-points and the computation of GMMs. The procedure is now described.

To motivate intuition, and for illustrative purposes, consider a system with additive Gaussian noise

$$x_{k+1} = f(x_k, u_k) + \omega_k \tag{46}$$

$$y_k = g(x_k, u_k, \eta_k) \tag{47}$$

Assume that a PF-based estimation stage has processed information up to time k_p, the prognosis time; thus providing an empirical posterior distribution for system states via a set of weighted samples (particles) $\{x_{k_p}^{(i)}, W_{k_p}^{(i)}\}_{i=1}^{N_p}$ denoted by

$$p(x_{k_p}|y_{1:k_p}) \approx \sum_{i=1}^{N_p} W_{k_p}^{(i)} \delta_{x_{k_p}^{(i)}}(x_{k_p}). \tag{48}$$

Let us also assume that the process noise ω_k PDF is approximated by the GMM:

$$\omega_k \sim \sum_{l=1}^{N_c} \gamma_k^{(l)} \mathcal{N}(\mu_k^{(l)}, \Sigma_k^{(l)}), \tag{49}$$

with $\gamma_k^{(l)} > 0$, $\mu_k^{(l)}$ and $\Sigma_k^{(l)}$ the l-th weight, mean and covariance matrix respectively, $\sum_{l=1}^{N_c} \gamma_k^{(l)} = 1$. Then, by applying Eqs. (44)–(48), the one-step predicted state PDF can be approximated as

$$p(x_{k_p+1}|y_{1:k_p}) \approx \left(\sum_{i=1}^{N_p} W_{k_p+1}^{(i)} \delta_{f(x_{k_p}^{(i)}, u_{k_p})}(x_{k_p+1}) \right) \otimes \omega_{k_p}. \tag{50}$$

Thus, $p(x_{k_p+1}|y_{1:k_p})$ is approximated by the convolution (denoted by $\otimes$) between an empirical PMF (defined by particles $\{f(x_{k_p}^{(i)}, u_{k_p}), W_{k_p+1}^{(i)}\}_{i=1}^{N_p}$) and the continuous PDF of ω_k. Note that the state transition model is used to reallocate particle positions in the state-space and $W_{k_p+1}^{(i)} = W_{k_p}^{(i)}$, as the increment of uncertainty related to one-step predictions is characterized by the convolution with the process noise.

Assuming that the process noise is distributed as GMM, Eq. (50) can be rewritten as:

$$p(x_{k_p+1}|y_{1:k_p}) \approx \sum_{i=1}^{N_p} W_{k_p+1}^{(i)} \sum_{l=1}^{N_c} \gamma_{k_p}^{(l)} \mathcal{N}(x_{k_p+1};\, \mu_{k_p}^{(l)} + f(x_{k_p}^{(i)}, u_{k_p}),\, \Sigma_{k_p}^{(l)}) \tag{51}$$

Eq. (51) provides a continuous representation of the state vector PDF at time instant $k_p + 1$. Unfortunately, this result cannot be extended beyond that particular instant, since u_{k_p+j} is a random variable $\forall\, j \geq 1$ and $f(\cdot)$ is a nonlinear function. In fact, it cannot even be assured that the predicted PDF at time $k_p + 2$ will be a GMM in general. Furthermore, in most cases, predicted PDFs beyond time $k_p + 2$ will generate computationally intractable distributions for real-time applications, as there is no analytic expression for them, making it difficult to perform statistical inference.

Now, assuming that u_{k_p+j} is a random variable $\forall\, j \geq 1$, it can be stated that, $\forall\, k > k_p$,

$$p(x_k|y_{1:k_p}) = \int_{\mathcal{U}_{k_p+1:k-1}} p(u_{k_p+1:k-1}) p(x_k|u_{k_p+1:k-1}, y_{1:k_p}) du_{k_p+1:k-1}. \tag{52}$$

Eq. (52) can be sequentially approximated if the sequence $\{u_{k_p+j}\}_{\forall j \geq 1}$ is assumed to be the realization of a Markov Chain, since the integration over the space of future system inputs is transformed into a summation over a finite amount of values (refer to Eq. (60) for one specific example).

However, it is still true that predicted state PDFs will not be a GMM in general, since $f(\cdot)$ is a nonlinear function. An alternative to overcome this problem, inspired by Monte Carlo (MC) approaches, would be to represent $\mathcal{N}(x_{k_p+1};\, \mu_{k_p}^{(l)} + f(x_{k_p}^{(i)}, u_{k_p}), \Sigma_{k_p}^{(l)})$ as a set of weighted samples. Nonetheless, this option will inevitably exponentially increase the number of samples as long as the prognosis procedure continues.

The proposed uncertainty characterization (prognosis) algorithm (Steps 1–4), based on *Unscented Transform Approximation Clustering* (UTAC) [18], aims at overcoming these difficulties by suggesting a procedure that can be implemented in real-time. It helps to generate approximated versions of the predicted state PDF at future time instants $k > k_p$, based on the fact that the filtering stage has been solved via a PF implementation (i.e., an empirical PF-based PMF, see Eq. (4), is the input to the prognosis stage). It is assumed that the estimation stage finishes at time instant k_p, the time at which the prognostic algorithm is executed.

●*Step 1: State transition via sigma-points*

The process noise $\omega_k \sim \sum_{l=1}^{N_c} \gamma_k^{(l)} \mathcal{N}(\mu_k^{(l)}, \Sigma_k^{(l)})$ is characterized by a weighted sum of Delta-dirac functions representing each component of the GMM by a set of sigma-points. In fact, indexing each component of the GMM by l and the elements of each set of sigma-pints by j, we have:

$$\sum_{l=1}^{N_c} \gamma_k^{(l)} \mathcal{N}(x_k;\, \mu_k^{(l)}, \Sigma_k^{(l)}) \approx \sum_{l=1}^{N_c} \sum_{j=0}^{2n_x} \gamma_k^{(l)} \mathcal{W}_k^{(l,j)} \delta_{\mathcal{X}_k^{(l,j)}}(x_k), \tag{53}$$

where $\{\mathcal{X}_k^{(l,j)}, \mathcal{W}_k^{(l,j)}\}$ is a sigma-point set, with $l = 1, \ldots, N_c$ and $j = 0, \ldots, 2n_x$. It is now possible to represent the process noise by a set of deterministically fixed weighted samples, $\{\mathcal{X}_k^{(l,j)}, \gamma_k^{(l)} \mathcal{W}_k^{(l,j)}\}$.

The state transition equation is then applied to each one of the N_k particles that are related to the k-th prognostic time instant, yielding the prior PDF

$$\hat{p}(x_k|u_{k-1}, y_{1:k_p}) = \sum_{i^*=1}^{N_k N_c (2n_x+1)} \tilde{W}_k^{(i^*)} \delta_{\tilde{x}_k^{(i^*)}}(x_k), \tag{54}$$

where $p(x_k|u_{k-1}, y_{1:k_p}) \approx \hat{p}(x_k|u_{k-1}, y_{1:k_p})$, and

$$\tilde{x}_k^{(i^*)} = f(\tilde{x}_{k-1}^{(i)}, u_{k-1}, \mathcal{X}_{k-1}^{(l,j)}), \tag{55}$$

$$\tilde{W}_k^{(i^*)} = W_{k-1}^{(i)} \gamma_{k-1}^{(l)} \mathcal{W}_{k-1}^{(l,j)}, \tag{56}$$

with $i^* = i^*(i, j, l)$, and $\tilde{x}_{k_p+1}^{(i^*)} = f(x_{k_p}^{(i)}, u_{k_p}, \mathcal{X}_{k_p}^{(l,j)})$. In other words, each i^* is assigned to a unique combination of i, j and l. Hence, as $i = 1, \ldots, N_k$, then $i^* = 1, \ldots, N_k \cdot N_c \cdot (2n_x + 1)$. In addition, it is possible to state that

$$p(x_k|y_{1:k_p}) \approx \int_{u_{k-1}} p(u_{k-1})\hat{p}(x_k|u_{k-1}, y_{1:k_p})du_{k-1}, \tag{57}$$

because the integration with respect to $u_{k_p+1:k-2}$ is implicitly contained in previous iterations of the algorithm, when executed in a sequential manner.

Note that if the sequence $\{u_{k_p+j}\}_{\forall j\geq 1}$ is assumed as the realization of a Markov Chain of N_u states, then the probability transition kernel would be of the form:

$$K_u = \begin{bmatrix} p_{1,1} & p_{1,2} & \cdots & p_{1,N_u} \\ p_{2,1} & p_{2,2} & \cdots & p_{2,N_u} \\ \vdots & \vdots & \ddots & \vdots \\ p_{N_u,1} & p_{N_u,2} & \cdots & p_{N_u,N_u} \end{bmatrix}, \tag{58}$$

and then the state probabilities can be updated as:

$$\begin{bmatrix} p(u_{k+1}^{(1)}) & p(u_{k+1}^{(2)}) & \cdots & p(u_{k+1}^{(N_u)}) \end{bmatrix} = K_u \cdot \begin{bmatrix} p(u_k^{(1)}) & p(u_k^{(2)}) & \cdots & p(u_k^{(N_u)}) \end{bmatrix}. \tag{59}$$

Thus, finally, Eq. (57) can be written as:

$$p(x_k|y_{1:k_p}) \approx \sum_{m=1}^{N_u} \sum_{i^*=1}^{N_k N_c(2n_x+1)} p(u_{k-1}^{(m)})\tilde{W}_k^{(i^*)}\delta_{\tilde{x}_k^{(i^*)}}(x_k). \tag{60}$$

●*Step 2: Fitting a GMM*

Considering that the empirical representation computed in Step 1 incorporates uncertainty by incrementing the number of particles, we apply the WEM algorithm (Section 2.4) to fit a GMM of M_k components, denoted by GMM_k:

$$GMM_k(x_k) = \sum_{j=1}^{M_k} \alpha_k^{(j)} \mathcal{N}(x_k;\, \mu_k^{(j)},\, \Sigma_k^{(j)}). \tag{61}$$

If $k = k_p + 1$, we use the k-Means algorithm (or another suitable option) to obtain an initial condition for the parametric estimates. Otherwise, an alternative is to use GMM_{k-1} and apply $f(\cdot)$ to the mean vector of each kernel.

●*Step 3: Simplification using sigma-points*

Sigma-points are used once again in this step for representing $GMM_k(x_k)$. The idea behind Step 2 is that we can vary (reduce) the size of the incremented particle population by modifying the value of M_k, according to the following relationship:

$$N_{k+1} = M_k \cdot (2n_x + 1) \tag{62}$$

●*Step 4: Repetition*

As Steps 1–3 define an invariant (with a controlled number of particles), these steps can be repeated indefinitely, propagating uncertainty until the whole prediction horizon is covered.

4.1. Comments on requirements of the proposed prognostic algorithm

Having explained the proposed uncertainty propagation (prognostic) algorithm, it is important to note how we ensure that the design requirements are met, as well as to specify certain conditions for the convergence of the WEM algorithm.

The first and fourth requirements are satisfied by the proposed algorithm because (i) it uses empirical distributions that are supplied by PFs during the estimation stage, and (ii) it allows for the characterization of exogenous inputs at each nonlinear state transition. The second requirement is achieved satisfactorily as Step 2 provides flexibility in terms of the simplification of the prior PDF at each time instant. Regarding the third requirement, it must be noted that the intensive use of sigma-points preserves the first two moments of the distribution, characterizing at least the mean and covariance of the original distribution more precisely.

The most important feature offered by the proposed algorithm is related to the manner in which we fulfill both the second and third requirements. The algorithm is elegant in terms of how it (deterministically) allocates samples in the state-space. A finite number of particles is scattered to approximate a PDF whose support grows as you move forward in the future. In addition, these samples are correctly positioned and appropriately weighted under the concept of sigma-points.

Step 3 saves computational resources because it provides a bounded number of particles from which it is possible to apply statistical inference. A reasonable alternative is to sample from the GMM set in Step 2 but, as the probability mass is spread on the state-space, it would be necessary to continuously increase the number of samples as a function of the remaining prediction horizon; a fact that would also affect the algorithm efficiency.

4.2. Convergence of the WEM algorithm

The proposed approach relies heavily on the convergence of the WEM algorithm in Step 2. Since the WEM algorithm is iterative and seeks a local extremum for the optimization problem, it may converge to incorrect solutions (or simply it may not converge at all) if at least one of the following conditions is met:

1. The initial values for the GMM parameters are remarkably different from the true ones.
2. The structure of the GMM model is over-parametrized.
3. The number of samples is insufficient in terms of the number of parameters to be estimated.

The usage of weighted samples (in Step 1, all sigma-points inherit the weight of the original particle, a fact that is explicitly used to compute the probability distribution (57)) allows the use of fewer particles in the characterization of the predicted state PDF. From a *frequentist* standpoint, weights can be interpreted as a measure of the probability mass associated with each sample, while in conventional sampling approaches require several samples to represent "probability". In other words, sample weights contain valuable information and allow faster convergence of the algorithm.

Although this paper does not include a formal analysis on convergence issues associated with implementations of the WEM algorithm, we would like to address certain considerations that may be taken into account to avoid convergence problems. Particularly, it is highly likely that these problems could occur in scenarios where it is intended to build a GMM (Step 2) using an insufficient number of samples from the previous step (Step 1). A measure to overcome this problem is to increase the number of sigma-points generated at Step 3.

The general procedure for the computation of sigma-points (see Section 2.3) uses the empirical covariance matrix of the state vector, locating sigma-points in directions defined by the matrix eigenvectors. Recalling that the equations that determine the positions of the sigma-points are given by:

$$\begin{aligned}
&\mathcal{X}^{(0)} = \bar{x} && i = 0,\\
&\mathcal{X}^{(i)} = \bar{x} + \left(\sqrt{(n_x+\kappa)\cdot P_x}\right)_i && i = 1,\ \ldots,\ n_x,\\
&\mathcal{X}^{(i)} = \bar{x} - \left(\sqrt{(n_x+\kappa)\cdot P_x}\right)_i && i = n_x+1,\ \ldots,\ 2n_x,
\end{aligned}$$

and also considering that the covariance matrix of the state matrix can be decomposed into $P_x = S_x S_x^T$, it follows that [48]:

$$P_x = S_x S_x^T = S_x I S_x^T = S_x C C^T S_x^T = S_x^r S_x^{rT}, \tag{63}$$

where I is the identity matrix and C is a rotation matrix, both of dimension $n_x \times n_x$. The rotation matrix C allows for the allocation of sigma-points in different directions of the eigenvectors of the covariance matrix P_x maintaining the properties of the UT and therefore achieving a greater number of sigma-points (whose weights should be properly defined).

In a case where n_x=2, the rotation matrix $C = C(\theta)$ would be

$$C(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \tag{64}$$

4.2.1. Increasing the number of sigma-points

Since the WEM algorithm may not converge due to the scarcity of weighted samples, increasing the number of sigma-points may help to solve this problem by augmenting the density of samples in the state-space. This task can be addressed by considering sigma-points in directions that differ from the ones classically defined by eigenvectors.

From Section 2.3, we can infer that if $x \in \Re^{n_x}$, it is only possible to generate $2n_x + 1$ sigma-points. However, we can increase this quantity to $2n_x n_r + 1$ by including rotations.

Indeed, given that a Gaussian distribution can be written as:

$$\mathcal{N}(x;\, \bar{x},\, P_x) = \frac{1}{n_r} \sum_{r=0}^{n_r-1} \mathcal{N}(x;\, \bar{x},\, P_x) \tag{65}$$

$$\mathcal{N}(x;\, \bar{x},\, P_x) = \frac{1}{n_r} \sum_{r=0}^{n_r-1} \mathcal{N}(x;\, \bar{x},\, S_x S_x^T), \tag{66}$$

and considering the r-th n_x-dimensional rotation square matrix denoted as $C\left(\frac{\pi}{2n_r}r\right)$, then we can rewrite the expressions as:

$$\mathcal{N}(x;\,\bar{x},\,P_x) = \frac{1}{n_r}\sum_{r=0}^{n_r-1}\mathcal{N}(x;\,\bar{x},\,S_xS_x^T) \tag{67}$$

$$\mathcal{N}(x;\,\bar{x},\,P_x) = \frac{1}{n_r}\sum_{r=0}^{n_r-1}\mathcal{N}\left(x;\,\bar{x},\,S_xC\left(\frac{\pi}{2n_r}r\right)C\left(\frac{\pi}{2n_r}r\right)^TS_x^T\right) \tag{68}$$

$$\mathcal{N}(x;\,\bar{x},\,P_x) = \frac{1}{n_r}\sum_{r=0}^{n_r-1}\mathcal{N}(x;\,\bar{x},\,S_x^rS_x^{rT}), \tag{69}$$

with $S_x^r = S_xC(\frac{\pi}{2n_r}r)$. Taking sigma-points on each of the Gaussian distributions, with rotated covariance matrices, we define a general way for augmenting the number of sigma-points:

$$\begin{aligned}
&\mathcal{X}^{(0)} = \bar{x} && i = 0\\
&\mathcal{X}^{(i)} = \bar{x} + \left(\sqrt{(n_x+\kappa)\cdot P_x}\right)_i && i = 1,\,\ldots,\,n_x\\
&\mathcal{X}^{(i)} = \bar{x} - \left(\sqrt{(n_x+\kappa)\cdot P_x}\right)_i && i = n_x+1,\,\ldots,\,2n_x\\
&\mathcal{X}^{(i)} = \bar{x} + \left(\sqrt{(n_x+\kappa)\cdot P_x}\cdot C\left(\frac{\pi}{2n_r}\right)\right)_i && i = 2n_x+1,\,\ldots,\,3n_x\\
&\mathcal{X}^{(i)} = \bar{x} - \left(\sqrt{(n_x+\kappa)\cdot P_x}\cdot C\left(\frac{\pi}{2n_r}\right)\right)_i && i = 3n_x+1,\,\ldots,\,4n_x\\
&\vdots && \vdots\\
&\mathcal{X}^{(i)} = \bar{x} + \left(\sqrt{(n_x+\kappa)\cdot P_x}\cdot C\left(\frac{\pi}{2n_r}(n_r-1)\right)\right)_i && i = 2(n_x-2)n_r+1,\,\ldots,\,2(n_x-1)n_r\\
&\mathcal{X}^{(i)} = \bar{x} - \left(\sqrt{(n_x+\kappa)\cdot P_x}\cdot C\left(\frac{\pi}{2n_r}(n_r-1)\right)\right)_i && i = 2(n_x-1)n_r+1,\,\ldots,\,2n_xn_r
\end{aligned}$$

where

$$\begin{aligned}
&\mathcal{W}^{(0)} = \frac{\kappa}{n_x+\kappa} && i = 0\\
&\mathcal{W}^{(i)} = \frac{1}{2n_r(n_x+\kappa)} && i = 1,\,\ldots,\,2n_xn_r
\end{aligned}$$

4.2.2. General n_x-dimensional rotation matrix

Let us assume that $n_x \geq 2$. According to [49], rotations in an n_x-dimensional space can be performed by rotating a 2-dimensional plane on an (n_x-2)-dimensional subspace. Since in Section 4.2.1 we require to rotate the eigenvectors of S_x, we may apply this property to an orthonormal basis built with its eigenvectors $v_1, v_2, \ldots, v_{n_x} \in \Re^{n_x}$. Thus, without lost of generality, we can define:

$$A = \left[v_1\ \ v_2\ \ \ldots\ \ v_{n_x-2}\right],\ B = \left[v_{n_x-1}\ \ v_{n_x}\right],$$

where the matrix A contains (n_x-2) eigenvectors of S_x that define the subspace on which we will rotate the plane defined by the columns of B. Therefore, from [49] we can define the rotation matrix:

$$C(B,\,\theta) = I_{n_x} + BB^T(\cos(\theta)-1) + BJ_2B^T\sin(\theta),$$

with I_{n_x} the n_x-dimensional identity matrix and

$$J_2 = \begin{bmatrix}0 & -1\\ 1 & 0\end{bmatrix}.$$

Note that for n_x=2, and considering $B = I_2$, we get:

$$C(B,\,\theta) = I_2 + I_2(\cos(\theta)-1) + I_2J_2I_2^T\sin(\theta) = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} + \begin{bmatrix}\cos(\theta)-1 & 0\\ 0 & \cos(\theta)-1\end{bmatrix} + \begin{bmatrix}0 & -\sin(\theta)\\ \sin(\theta) & 0\end{bmatrix} = \begin{bmatrix}\cos(\theta) & -\sin(\theta)\\ \sin(\theta) & \cos(\theta)\end{bmatrix},$$

which corresponds to the same expression shown in Eq. (64).

Although we are limited to rotate a plane defined by two arbitrary eigenvectors of S_x, it is possible to generalize the algorithm if we divide the probability mass equally between the sigma-points that are obtained from all possible combinations of rotation pairs.

5. Performance analysis of the proposed prognostic algorithm: Lithium-Ion battery end-of-discharge

This section analyzes the performance of the proposed prognostic algorithm in a case study that is focused on the problem of Lithium-Ion (Li-Ion) battery State-of-Charge (SoC) prognosis, which is equivalent to the problem of End-of-Discharge (EoD) estimation. This case study assumes the implementation of a PF algorithm, for SoC estimation purposes (filtering stage), which follows the recommendations suggested in [36] in terms of the number of particles utilized, as well as other parameters. State PDF estimates that are generated during the filtering stage are fed into the prognostic module as initial conditions. The proposed prognostic algorithm is evaluated for three different parameter configurations, in terms of the quality of statistics that are computed from EoD PMF estimates.

Both filtering and prognostic stages use a state-space model to represent the evolution in time of the Li-Ion battery voltage as a function of the SoC, the battery internal impedance, and the discharge current (exogenous system input). The objective in this case study is to prognosticate the moment in which the battery energy is depleted (failure event). The solution for this problem can be obtained offline by computing a "ground truth" failure PMF. This PMF is calculated using Monte Carlo simulations for future trajectories of the state vector (including the future SoC), which are conditioned on several realizations of a probabilistic representation of the future battery operating profiles. The true challenge is to solve this prognostic problem in real-time, using the proposed approach. The performance of the online prognostic algorithm will be measured in terms of the quality of the resulting empirical ToF PMF. Such a PMF is built using the new ToF probability measure that was introduced in Section 3, where ToF is equivalent to EoD time. Indeed, in the context of SoC estimation and prognostic problems, a lack of energy represents a failure mode of the system.

For most of the battery operating range, the relationship between SoC and the *Open Circuit Voltage* (OCV) curve can be characterized by an affine function; see "*zone* 2" in Fig. 1. Nonetheless, we have adopted the model proposed in [36], which also accounts for the nonlinear behavior present in "*zone* 1" and "*zone* 3".

State transition model:

$$x_1(k+1) = x_1(k) + \omega_1(k) \tag{70}$$

$$x_2(k+1) = x_2(k) - \frac{\hat{v}(x_1(k), x_2(k), u(k))\cdot u(k)\cdot T_s}{E_{crit}} + \omega_2(k) \tag{71}$$

Measurement model:

$$y(k) = \hat{v}(x_1(k), x_2(k), u(k)) + \eta(k), \tag{72}$$

with

$$\hat{v}(x_1(k), x_2(k), u(k)) = v_L + (v_0 - v_L)\cdot e^{\gamma\cdot(x_2(k)-1)} + \alpha\cdot v_L\cdot(x_2(k)-1) + (1-\alpha)\cdot v_L\cdot\left(e^{-\beta} - e^{-\beta\cdot\sqrt{x_2(k)}}\right) - u(k)\cdot x_1(k) \tag{73}$$

In this model, the input to the system $u(k) = i(k)[A]$ is defined as the discharge current, while $y(k) = v(k)[V]$ is the resulting voltage difference in the terminals of the battery and the system output. The states $x_1(k)$ corresponds to the module of the internal impedance (unknown model parameter); whereas $x_2(k)$ is the battery SoC measured with respect to E_{crit}, the expected total energy delivered by the battery. Process noises $\omega_1(k)$ and $\omega_2(k)$, and the measurement noise $\eta(k)$ assume a zero mean Gaussian distribution. Finally, $T_s[s]$ is the sample time and v_0, v_L, α, β and γ are model parameters to be estimated offline (see [36] for details of this procedure).

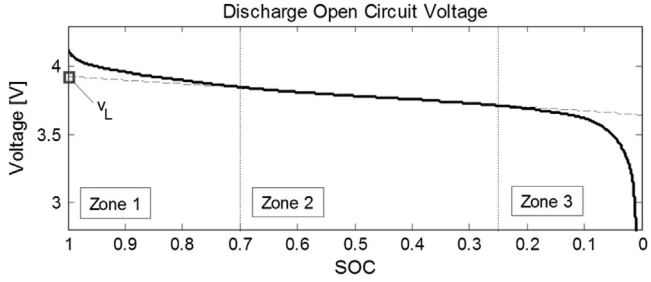

Fig. 1. OCV curve of a Li-Ion cell (black line) and the projection of its linear operational range (dashed gray line) as a function of SoC, [50,36].

5.1. Characterization of future discharge current profiles using Markov chains

Battery SoC and EoD prognosis requires the incorporation of knowledge about the future battery usage profile, in terms of its discharge current $i(k)$. For this purpose we can statistically characterize this consumption profile by assuming that past discharge current measurements correspond to a realization of a two-state Markov chain [36]. In this statistical characterization, one of the states of the Markov chain represents a high-energy consumption profile, while the second state is related to low-energy consumption profiles.

The implementation of this statistical characterization for the system inputs requires the estimation of transition probabilities, and the definition of maximum and minimum discharge currents. For example, if it is assumed that the battery is energizing electro-mechanical systems and providing relatively constant power, the average value of the current will tend to increase as the SoC decreases (because the battery voltage also decreases). This fact forces the incorporation of adaptation schemes that could learn (from acquired data) the most representative values for high and low energy consumption states. For this purpose, battery discharge data is segmented at regular time intervals. Each interval contains a fixed number of samples N_w. This segmentation generates m time intervals such that $m = \lfloor L/N_w \rfloor$, where L is the number of measurements available at the moment. For simplicity, if the ratio between the number of measurements and N_w is not an integer, then the first window can include more data samples. A low-pass filter is applied to the discharge current data to discard outliers and anomalous peaks, obtaining a filtered signal $i'(k)$ as a result. Then, for each j-th interval ($j = 1, \ldots, m$), we compute the minimum and maximum discharge current values, $i_{low}^{(j)} = \min\{i'(k)\}$, and $i_{high}^{(j)} = \max\{i'(k)\}$, where $k = 1, \ldots, N_w$ is a time index valid within the j-th window. Next, on each interval, current measurements are quantized into two possible values defined by $i_{low}^{(j)}$, and $i_{high}^{(j)}$. These values define the low-energy, and high-energy consumption states of the Markov Chain that characterizes the j-th interval. Discharge current data satisfying $i' > (i_{high}^{(j)} + i_{low}^{(j)})/2$ are quantized as $i_{high}^{(j)}$; otherwise, they are quantized as $i_{low}^{(j)}$. For each interval, it is possible to compute transition probabilities p_{ij} between low-energy and high-energy consumption states. These transition probabilities are estimated through maximum likelihood. For more details about this procedure, refer to [36].

5.2. Performance measures

In order to study the performance of the proposed algorithm, an assessment is made based on the following measures: expectation and $JITP_{\alpha\%}$ [35]. The first measure is well-known, whereas the last one is now explained in more detail.

$JITP_{\alpha\%}$ intends to characterize the risk associated to decisions based on the outcome of prognostic algorithms. Indeed, this measure assumes that the risk is a function of the tails of the underlying failure PDF. In this regard, the concept of *Just In Time Point* (JITP) is defined as

$$JITP_{\alpha\%} = \underset{k}{\operatorname{argmin}}(\mathcal{P}(ToF \leq k) \geq \alpha\%). \tag{74}$$

The JITP generates the notion of a threshold for the probability of failure. Also, it allows to define the index α_{crit} as a measure of risk aversion:

$$\alpha_{crit} = \underset{\alpha}{\operatorname{argmax}}(JITP_{\alpha\%}(k_p) \leq ToF)_{\forall k_p \in [1, ToF]}. \tag{75}$$

This value is defined as the maximum $\alpha \in [0, 100]$ guaranteeing that $JITP_{\alpha\%}(k_p)$ is smaller than the true ToF, $\forall\, k_p \in [1, ToF]$; where k_p is the time instant where prognosis begins.

5.3. State estimation stage: parameter setup for particle filtering algorithms

The outcome of the estimation stage is critical in terms of the performance that could be expected from probability-based prognostic algorithms, since the filtering algorithm defines the initial condition that will be used to compute long-term predictions. For this reason, it is paramount to define proper procedures for parameter setup during the filtering stage. In this work, we follow the procedure proposed in [36] to setup the parameters of our particle filter implementation, given that the state equations that govern model dynamics are basically identical. Measurement noise can be directly characterized from sensor measurements, particularly considering that most of the model parameters that are related to the observation Eq. (72) are estimated offline from the analysis of complete battery discharge curves [36]. Then, we only need to study the setup of the following parameters:

- Number of particles.
- Hyperparameters associated with the definition of process noise.

Process noise hyperparameters represent epistemic sources of uncertainty, and must be set so that the support of the prior distribution $p(x_k|x_{k-1})$ also contains the posterior distribution $p(x_k|y_{1:k})$. The number of particles is related to convergence

properties of the particle filter algorithm [6].

The number of particles can be set following the procedure proposed in [36], which is also focused on the problem of battery SoC estimation and prognosis. According to those results, a number between 50 and 100 particles is sufficient to provide a reasonable characterization of the underlying PDFs. Since the initial condition of the battery SoC is unknown, it may be recommendable to implement outer feedback correction loops (see Section 2.2) to generate time-varying hyperparameters for the process noise. In those cases, it is recommended to gradually reduce the variance of the process noise as the Bayesian processor acquires more measurements [22].

5.4. Experimental results

The proposed algorithm is applied to the problem of Li-Ion battery EoD prognosis, establishing a comparison between the solution provided by the online algorithm and the one yielded by offline Monte Carlo simulations. For the implementation of the PF algorithm, some functions of the Matlab Toolkit ReBeL-0.2.7 developed by Rudolph van der Merwe and Eric A. Wan are used. The values of the model parameters described by Eqs. (70)–(72) are the following: $\alpha=0.15$, $\beta=12$, $\gamma=6.6061$, $v_0=4$, $v_L=3.8126$, and $E_{crit}=19{,}865$. Online measurements are provided by experimental voltage and current discharge data of a Li-Ion 18,650 cell. The data set consists of 3638 successive observations with a sampling time of $T_s = 1[s]$. The data acquisition experiment is designed to force a discharge current profile that is equivalent to the realization of a homogeneous Markov chain of two states, which represent random changes between high and low current levels; see *Set* #2 of Section 3.2 in [51]. The states of the Markov chain are $u_0 = 1[A]$ and $u_1 = 3[A]$, and the transition kernel is:

$$K = \begin{bmatrix} 0.5709 & 0.4291 \\ 0.5577 & 0.4423 \end{bmatrix}. \tag{76}$$

As there are two possible input values, then

$$\hat{p}(x_k, u_k|y_{1:k_p}) \approx \sum_{i=0}^{1} p(u_i)\hat{p}(x_k|u_i, y_{1:k_p}), \quad \forall\, k \geq k_p, \tag{77}$$

where each PDF $\hat{p}(x_k, u_i|y_{1:k_p})$, with $i=0,1$, is computed separately applying Step 1, and later merged in Step 2. At every moment, the probability associated with the value of each input is updated using the transition kernel shown in Eq. (76).

Moreover, the *hazard zone* [40,25] is defined in this case as:

$$p(failure|x_k) = \begin{cases} 1, & SoC(k) \leq 40\% \\ 0, & \sim \end{cases} \tag{78}$$

To evaluate the performance of the prognostic algorithm based on a combination of sigma-points and GMMs, we study three scenarios. On each one of these scenarios, a constant number of particles and GMMs components (in Step 2) are used. Also, we assume constant values for $\kappa > 0$ (see Section 2.3), both in Step 1 and in Step 3 of the proposed algorithm.

Let the value of κ (required for the calculation of sigma-points) be denoted as κ_1 in Step 1 and as κ_3 in Step 3, respectively. After studying the impact of variations on the value of κ_1 on a series of experiments, we have found that values smaller or equal to 1 generate results which do not differ significantly. For this reason, we have chosen $\kappa_1 = 1$ for all cases. This is mainly caused by the fact that sigma-points in Step 1 are solely used to adjust the GMM at Step 2. However, the algorithm is very sensitive to the value of κ_3, since this parameter determines the density (in the state-space) of the particle population that is used as an input of the prognostic algorithm (to predict the state PDF at the next time step). If samples (particles) are located far away from the support of the true state PDF, much of the approximation is lost. As a result, the value of κ_3 must be small. The results are highly affected when $\kappa_3 > 0.15$, so we choose to define $\kappa_3 = 0.12$ for all simulations.

As it has been mentioned before, three different parameter configurations (for the proposed prognostic algorithm) are tested against the "ground truth" Monte Carlo solution. According to the notation used in Section 4, N_k represents the size of the particle population at the k-th time instant, and M_k is the number of components for the GMM fitted at the k-th time instant:

- Case 1: $N_k=45$ and $M_k=5$.
- Case 2: $N_k=90$ and $M_k=10$.
- Case 3: $N_k=135$ and $M_k=15$.

It is important to mention that the N_k – the number of particles – is fixed both during estimation and prognosis stages for all three cases. Also, it is noteworthy that, according to Eq. (62),

$\frac{N_k}{M_k} = 9$ (number of sigma–points in Step 3),

although according to Section 2.3, the number of sigma-points should be $2n_x + 1 = 5$ (for $n_x = 2$). The increment in the number of sigma-points is based on the fact that the assumption of five sigma-points in Step 3 lead to convergence problems; see Section 4.2. Therefore, in Step 3, sigma-points are calculated as (see Sections 2.3 and 4.2.1 as a reference):

$$\mathcal{X}^{(0)} = \bar{x} \quad i = 0 \tag{79}$$

$$\mathcal{X}^{(i)} = \bar{x} + \left(\sqrt{(n_x + \kappa)\cdot P_x}\right)_i \quad i = 1, \ldots, n_x \tag{80}$$

$$\mathcal{X}^{(i)} = \bar{x} - \left(\sqrt{(n_x + \kappa)\cdot P_x}\right)_i \quad i = n_x + 1, \ldots, 2n_x \tag{81}$$

$$\mathcal{X}^{(i)} = \bar{x} + \left(\sqrt{(n_x + \kappa)\cdot P_x}\cdot C\left(\frac{\pi}{4}\right)\right)_i \quad i = 2n_x + 1, \ldots, 3n_x \tag{82}$$

$$\mathcal{X}^{(i)} = \bar{x} - \left(\sqrt{(n_x + \kappa)\cdot P_x}\cdot C\left(\frac{\pi}{4}\right)\right)_i \quad i = 3n_x + 1, \ldots, 4n_x \tag{83}$$

where

$$\mathcal{W}^{(0)} = \frac{\kappa}{n_x + \kappa} \quad i = 0 \tag{84}$$

$$\mathcal{W}^{(i)} = \frac{1}{4(n_x + \kappa)} \quad i = 1, \ldots, 4n_x. \tag{85}$$

Considering n_r=2, the number of sigma-points becomes $2n_x n_r + 1 = 9$. Since an increment on the number of sigma-points solely seeks to avoid convergence problems (at the cost of increasing computational complexity) we have chosen to consider n_r=2 in this particular implementation of the code. Using $n_r > 2$ would only incur in an unnecessary use of computational resources.

Results obtained after applying our proposed prognostic algorithm for this case study are shown below. It is important to note that, given the probabilistic nature of the PF-based estimation stage, the analysis of each case (i.e., each parameter configuration) considers 30 realizations for both filtering and prognostic stages. In this manner, we can incorporate the impact of different initial conditions for the prognostic stage. In addition, as the quality of this initial condition for prognosis improves as the number of particles increases, we have used 135 particles during the PF-based estimation stage, independently of the parameter configuration utilized during the prognosis stage. In the case of the "ground truth" failure PMF generated by Monte Carlo simulations, we also considered 135 particles to characterize the joint PDF of the state vector during the estimation stage. Monte Carlo simulation is then performed by propagating each particle in time, using the concept of "random walks" (defined by the state model) until reaching an EoD condition. This procedure is repeated 1000 times per each one of the 30 realizations considered during the filtering stage (i.e., a total of 30,000 simulations). The prognostic algorithm is always executed at the 1000th second of operation in the battery discharge experiment.

Before we present a brief analysis of the obtained results, it is important to emphasize a few facts. From Eq. (38), in which we propose a new definition for the computation of ToF PMFs in the context of online applications, it can be noted that the failure probability at any given time instant (in the future) depends on the failure probabilities computed for previous time instants. As the proposed prognostic algorithm becomes less accurate in terms of uncertainty propagation as time goes on (a problem that is shared by any suboptimal prognosis algorithm), we expect that the predicted ToF PMF will be better characterized on its left tail, rather than on the right tail. Now, in terms of the usefulness of prognostic results, our interest focuses on the characterization of the left tail (e.g., decisions based on failure probabilities greater than 40% are too risky in practice). Since our focus is on quantifying the capability of the algorithm for risk characterization, we have chosen to analyze the performance of our proposed prognostic algorithm in terms of the accuracy and precision exhibited by statistics that are typically used for real-time decision-making processes: conditional expectations and Just-in-Time-Points (JITPs). The latter statistic is more interesting as it directly quantifies the risk in ranges that are actually useful for real-time health management (the user needs to take actions, but the question is how much failure probability is the user going to risk by postponing those actions?).

Considering all of the above, let us proceed with the analysis of obtained results (Tables 1-3). First, it is interesting to note that in all three cases we get slightly biased failure PMFs, although these biases (as well as the standard deviations of conditional expectations) are negligible with respect to the prediction horizon (976.2[s]). Particularly for the parameter configuration used in Case 2 (see Fig. 3), the bias represents only a (conservative) error of 1.56% when compared to this

horizon (Table 2). Even in Case 3, the most biased result, the systematic error only represents 2.78% of the prediction horizon (Table 3). When comparing the quality of the obtained results with respect to the "ground truth" failure PMF in terms of statistics such as the JITP, differences are also negligible when compared with the prediction horizon. This is, indeed, a significant achievement. Particularly if we consider that the proposed prognostic approach allows probabilistic characterization of external inputs with just one realization (contrary to the already consolidated PF-based prognostic algorithm proposed in [13], which requires several realizations of the Markov chain that characterizes system inputs).

A more detailed comparison of the results obtained for Case 2 in terms of JITP estimates, with respect to Cases 1 and 3 (Figs. 2 and 4, respectively), shows that average values of these statistics in Case 2 exhibit a delicate balance between accuracy and precision (Table 2). Indeed, estimates for JITP statistics in Case 2 have smaller standard deviations than Case 1, and their average values also represent more conservative estimates (a fact that is critical for decision-making processes in prognosis). Case 3 exhibits even more conservative biases, although JITP estimates are also less precise (a fact that is reflected in bigger standard deviations). Considering the importance behind JITP statistics as measures for the operational risk of the system, we conclude that Case 2 shows the best performance.

Case 1: $N_k=45$ *and* $M_k=5$, $\forall\ k \geq k_p$

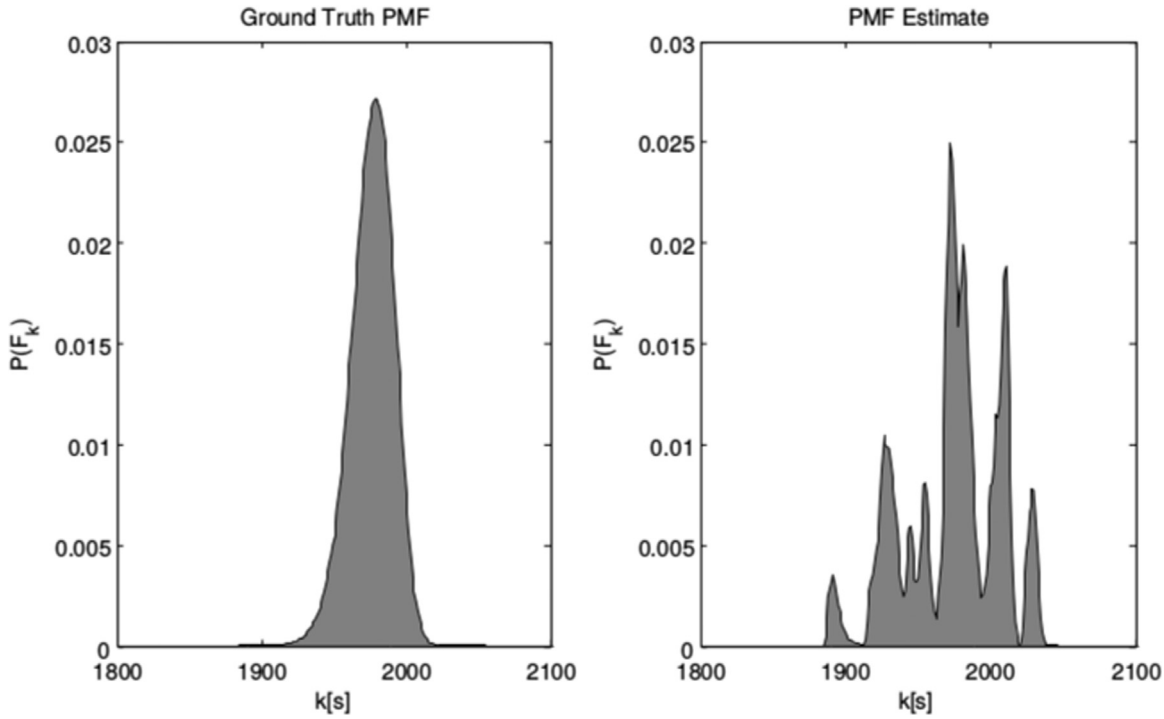

Fig. 2. Average EoD PMF generated by using 30 different initial conditions for the prognostic stage. The proposed prognostic algorithm uses 45 particles and a 5-kernel GMM. The Monte Carlo solution ("ground truth" Failure PMF) is computed using 135 particles during the filtering stage and 30,000 simulations for the prognostic stage. Monte Carlo solution is on the left, while the result of the novel prognostic algorithm based on sigma-points is presented on the right.

Table 1
Comparison between the value of statistics computed from the "ground truth" PMF (Monte Carlo) and the expectation of statistics computed from the results provided by the proposed prognostic algorithm (45 particles and a 5-kernel GMM).

Obtained results	Performance statistics			
	$\mathbb{E}\{\cdot\}$	$JITP_{5\%}$	$JITP_{10\%}$	$JITP_{15\%}$
Ground truth PMF (Monte Carlo)	1976.2	1950.0	1957.0	1961.0
Expectations (proposed algorithm)	1974.3	1969.3	1970.0	1970.7
Standard deviations (proposed algorithm)	33.3207	33.7601	33.8745	33.6554

Case 2: $N_k=90$ and $M_k=10$, $\forall\ k \geq k_p$.

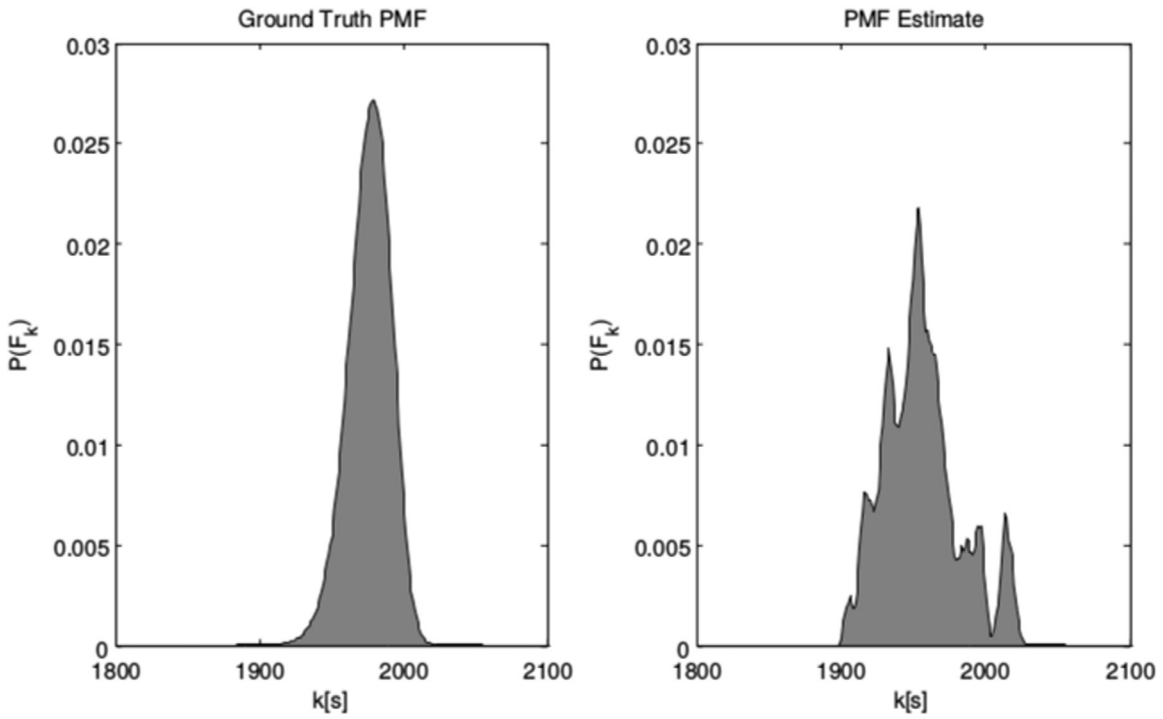

Fig. 3. Average EoD PMF generated by using 30 different initial conditions for the prognostic stage. The proposed prognostic algorithm uses 90 particles and a 10-kernel GMM. The Monte Carlo solution ("ground truth" Failure PMF) is computed using 135 particles during the filtering stage and 30,000 simulations for the prognostic stage. Monte Carlo solution is on the left, while the result of the novel prognostic algorithm based on sigma-points is presented on the right.

Table 2
Comparison between the value of statistics computed from the "ground truth" PMF (Monte Carlo) and the expectation of statistics computed from the results provided by the proposed prognostic algorithm (90 particles and a 10-kernel GMM).

	Performance statistics			
Obtained results	$\mathrm{E}\{\cdot\}$	$JITP_{5\%}$	$JITP_{10\%}$	$JITP_{15\%}$
Ground truth PMF (Monte Carlo)	1976.2	1950.0	1957.0	1961.0
Expectations (proposed algorithm)	1960.9	1953.6	1954.7	1955.6
Standard deviations (proposed algorithm)	27.9538	29.5070	29.3457	29.2393

Since the proposed prognostic algorithm assumes empirical distributions for the state vector (consisting of N_p particles), it is possible to state that the amount of kernels used to fit a GMM should not exceed N_p. Consider, for example, the case where we want to use a GMM of N_p kernels to characterize the uncertainty inherent to a population of N_p particles. We could (for example) choose to center each kernel on a specific particle, although in that case we would not be able to adjust the covariance matrices for each kernel because the resulting problem would be over parameterized (more parameters than data). The problem would only become worse if we were to use more than N_p kernels. On the other hand, the usage of just one kernel would be optimal only if the underlying distribution were to be Gaussian. These two extreme cases give intuition that there should be an optimal number of kernels M, such than $1 \leq M \leq N_p$. For this reason, it is not possible to state that results will necessarily improve by increasing the number of kernels in the GMM.

Case 3: $N_k = 135$ and $M_k = 15$, $\forall\ k \geq k_p$.

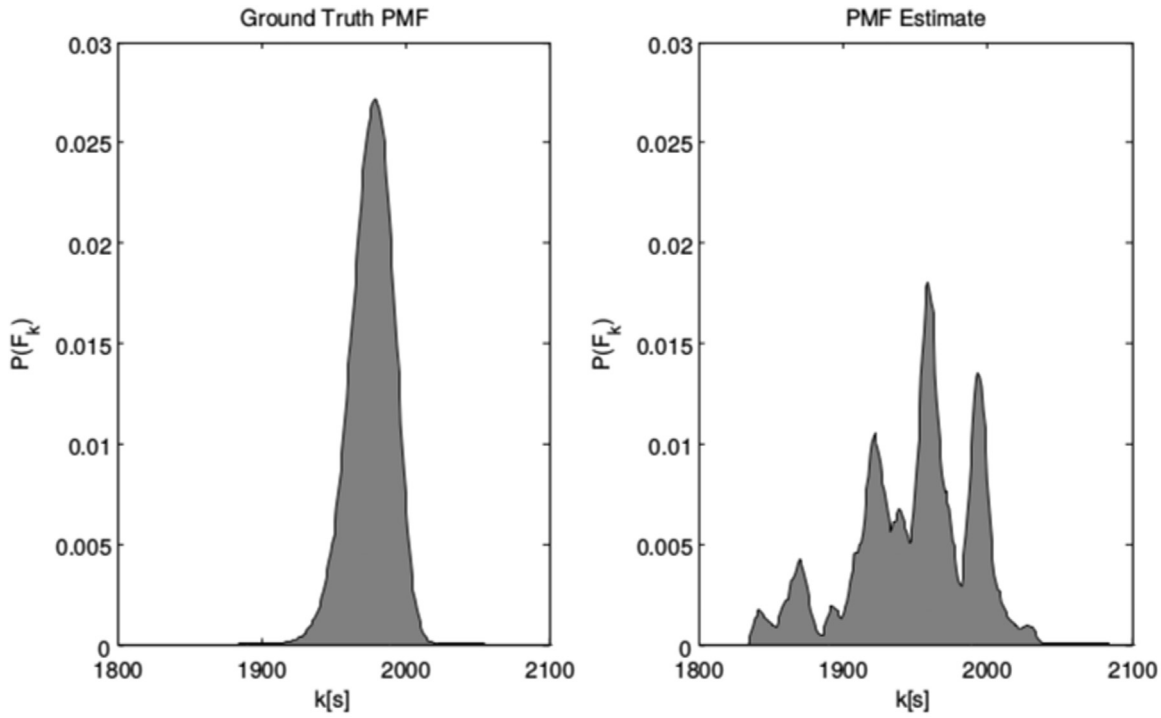

Fig. 4. Average EoD PMF generated by using 30 different initial conditions for the prognostic stage. The proposed prognostic algorithm uses 135 particles and a 15-kernel GMM. The Monte Carlo solution ("ground truth" Failure PMF) is computed using 135 particles during the filtering stage and 30,000 simulations for the prognostic stage. Monte Carlo solution is on the left, while the result of the novel prognostic algorithm based on sigma-points is presented on the right.

Table 3
Comparison between the value of statistics computed from the "ground truth" PMF (Monte Carlo) and the expectation of statistics computed from the results provided by the proposed prognostic algorithm (135 particles and a 15-kernel GMM).

	Performance statistics			
Obtained results	$\mathbb{E}\{\cdot\}$	$JITP_{5\%}$	$JITP_{10\%}$	$JITP_{15\%}$
Ground truth PMF (Monte Carlo)	1976.2	1950.0	1957.0	1961.0
Expectations (proposed algorithm)	1949.0	1939.0	1940.3	1941.5
Standard deviations (proposed algorithm)	40.4183	41.7583	41.8496	41.5732

Case 3 illustrate the effect of incorporating an excessive number of kernels in the GMM. Too many kernels provide a poor characterization of the ToF PMF, considering that you only have a limited amount of particles that can be used to adjust kernel parameters. Increasing the number of particles could help to adjust more parameters in the GMM, but it would also decrease the algorithm computational efficiency. Using too few kernels in the GMM (i.e., too few parameters) may be insufficient for proper characterization of the underlying state PDF, resulting in excessive simplification of the model and poor approximations, as it can be seen in Case 1 (Table 1). Case 2 represents a delicate balance between number of kernels in the GMM, number of particles and quality of results.

Although we present results of the proposed approach for very specific parameter configurations (Cases 1–3), mainly to show its potential for the computation of prognostic risk measures, it is noteworthy that the algorithm design parameters provide flexibility and applicability to a number of complex problems. Applicability, because our prognostic scheme is built assuming a state-space representation of the system and a probabilistic characterizations of exogenous inputs. Flexibility, because it is possible to easily adjust the manner in which the support of the true state PDF is explored by modifying few

algorithm parameters. The usage of sigma-points requires to define specific parameters for adequate allocation of samples in the state-space, whereas the (iterative) WEM algorithm needs adequate initial conditions and finishing thresholds. Furthermore, independently of how precisely could the state vector PDF be approximated during the estimation stage, the number of GMMs and weighted samples (obtained using sigma-points) plays a key role regarding the characterization of future uncertainty. Consider, for example, the performance of the algorithm in terms of the number of kernels used in the GMM. Results generated when using 5 or 15 kernels exhibited larger standard deviations for estimates of conditional expectations and JITPs, when compared to the case where 10 kernels were used. Even in Case 3, where 135 particles were used to obtain empirical distributions, performance of the algorithm degraded because convergence issues in the WEM algorithm, which was affected by an excess of kernels. Evidence indicates that there is an optimal number of kernels to be used, although the quality of the obtained results (even in the worst cases) is in the order of 2.5% of error.

A procedure for setting up all these design parameters must be developed considering the mathematical formalism that is related to the risk characterization provided by the ToF PMF definition presented in Section 3, which establishes a new paradigm in terms of fundamental precision limits for ToF prognosis. This task is, in the meantime, part of future work.

6. Conclusion and future work

This paper presents a major correction to the manner in which the ToF PMF is computed and also to its interpretation in the context of online monitoring. In this regard, the true probabilistic characterization for ToF is shown as a PMF which accounts for the probability of failure of systems in general, including of course nonlinear systems and non-Gaussian noise, or even systems presenting regeneration phenomena.

A novel algorithm for fault prognosis based on the use of sigma-points and GMMs has been also proposed. This new algorithm represents and improvement with respect to other prognostic approaches based on PFs, for it provides a better characterization of the tails of the underlying PDF of the system states and explicitly incorporates a probabilistic characterization of the system inputs. Contrary to other approaches that depend on regularization techniques to characterize the uncertainty associated with the predicted PDF, this method uses a combination of sigma-points (deterministically positioned in the state-space) and GMM-based approximations to propagate uncertainty and weights to properly characterize at least the first two moments of the distribution. Performance assessment of the proposed algorithm, for the case of Li-Ion cell EoD prognosis analyzes and compares the predicted failure time PMF with respect to the one obtained using intensive Monte Carlo simulation. Sensitivity analysis indicated that the quality of results provided by the proposed algorithm is affected mainly by initial conditions for the prognosis stage (as any other prognostic scheme), as well as the number of kernels used in GMM-based approximation (design parameter). Nevertheless, $JITP_{\alpha\%}$ statistics for Li-Ion cell EoD time show negligible errors when compared to the prediction horizon (in the order of 1% for 976 time steps in prediction). The latter allows to declare significant added value from our proposal in terms of its relevance for real-time risk assessment.

As future work we propose theoretical development of fundamental bounds that govern the estimation of the ToF PMF to measure performance of prognostic algorithms, and also the establishment of rules, either online (adaptive) or offline, for optimal parametrization of the proposed prognosis algorithm based on sigma-points.

Acknowledgment

This work has been partially supported by CONICYT FONDECYT Grant Nr. 1170044, CONICYT PIA Project ACT1405, and the Advanced Center for Electrical and Electronic Engineering, Basal Project FB0008.

References

[1] D. Dubois, Possibility theory and statistical reasoning, Comput. Stat. Data Anal. 51 (2006) 47–96.
[2] G. Shafer, A Mathematical Theory of Evidence, Princeton University Press, Princeton, NJ, 1976.
[3] L.A. Zadeh, Probability measures of fuzzy events, J. Math. Anal. Appl. 23 (2) (1968) 421–427.
[4] L.A. Zadeh, Fuzzy sets as a basis for a theory of possibility, Fuzzy Sets Syst. (1977) 9–34.
[5] D. Liu, V. Luo, Y. Peng, Uncertainty processing in prognostics and health management: an overview, in: 2012 IEEE Conference on Prognostics and System Health Management (PHM), 2012.
[6] A. Doucet, N. de Freitas, N. Gordon (Eds.), Sequential Monte Carlo Methods in Practice, Springer-Verlag, New York, 2001.
[7] U.D. Hanebeck, K. Briechle, A. Rauh, Progressive Bayes: a new framework for nonlinear state estimation, in: Proceedings of the SPIE AeroSense Symposium 2003, vol. 5099, Orlando, USA, 2003.
[8] A. Rauh, K. Briechle, U.D. Hanebeck, Nonlinear measurement update and prediction: prior density splitting mixture estimator, in: 2009 IEEE Control Applications, (CCA) & Intelligent Control, (ISIC), St. Petersburg, 2009, pp. 1421–1426.
[9] U.D. Hanebeck, M.F. Huber, V. Klumpp, Dirac mixture approximation of multivariate Gaussian densities, in: Proceedings of the 48th IEEE Conference on Decision and Control, 2009, Held Jointly with the 2009 28th Chinese Control Conference, CDC/CCC 2009, 2009, pp. 3851–3858.
[10] R.E. Kalman, A new approach to linear filtering and prediction problems, J. Basic Eng. 82 (1) (1960) 35–45.
[11] R. Faragher, Understanding the basis of the Kalman filter via a simple and intuitive derivation [lecture notes], IEEE Signal Process. Mag. 29 (5) (2012) 128–132.
[12] S. Engel, B. Gilmartin, K. Bongort, A. Hess, Prognostics, the real issues involved with predicting life remaining, in: Proceedings of 2000 IEEE Aerospace Conference, vol. 6, 2000, pp. 457–469.

[13] M. Orchard, G. Vachtsevanos, A particle-filtering approach for on-line fault diagnosis and failure prognosis, Trans. Inst. Meas. Control 31 (2009) 221–246.
[14] D.J.C. MacKay, Introduction to Monte Carlo methods, in: M.I. Jordan (Ed.), Learning in Graphical Models, Kluwer Academic Press, 1998, pp. 175–204.
[15] M. Orchard, G. Kacprzynski, K. Goebel, B. Saha, G. Vachtsevanos, Advances in uncertainty representation and management for particle filtering applied to prognostics, in: Proceedings of International Conference on Prognostics and Health Management, 2008, pp. 1–6.
[16] S.J. Julier, The scaled unscented transformation, in: Proceedings of American Control Conference (ACC 02), 2002, pp. 4555–4559.
[17] S.J. Julier, J.K. Uhlmann, Unscented filtering and nonlinear estimation, Proc. IEEE 92 (3) (2004) 401–422.
[18] J. Goldberger, H. Greenspan, J. Dreyfuss, Simplifying mixture models using the unscented transform, IEEE Trans. Pattern Anal. Mach. Intell. 30 (8) (2008) 1–7.
[19] A. Doucet, S.J. Godsill, C. Andrieu, On sequential Monte Carlo sampling methods for Bayesian filtering, Stat. Comput. 10 (3) (2000) 197–208.
[20] T. Li, S. Sun, T.P. Sattar, J.M. Corchado, Fight sample degeneracy and impoverishment in particle filters: a review of intelligent approaches, Expert Syst. Appl. 41 (8) (2014) 3944–3954.
[21] J. Liu, M. West, Combined parameter and state estimation in simulation-based filtering, in: A. Doucet, N. deFreitas, N. Gordon (Eds.), Sequential Monte Carlo Methods in Practice, Springer-Verlag, 2001, pp. 197–223.
[22] M. Orchard, F. Tobar, G. Vachtsevanos, Outer feedback correction loops in particle filtering-based prognostic algorithms: statistical performance comparison, Stud. Inform. Control 18 (4) (2009) 295–304.
[23] G.J. McLachlan, T. Krishnan, The EM Algorithm and Extensions, Wiley, Hoboken, NJ, 2008.
[24] R. van der Merwe, E. Wan, Gaussian mixture sigma-point particle filters for sequential probabilistic inference in dynamic state-space models, in: Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), vol. 6, 2003, pp. 701–704.
[25] L. Tang, M.E. Orchard, K. Goebel, G. Vachtsevanos, Novel metrics and methodologies for the verification and validation of prognostic algorithms, in: Proceedings of IEEE Aerospace Conference, 2011, pp. 5–12.
[26] S. Sankararaman, K. Goebel, Why is the remaining useful life prediction uncertain? in: 2013 Annual Conference of the Prognostics and Health Management Society, 2013.
[27] S. Sankararaman, Y. Ling, S. Mahadevan, Uncertainty quantification and model validation of fatigue crack growth prediction, Eng. Fract. Mech. 78 (7) (2011) 1487–1504.
[28] S. Sankararaman, Y. Ling, C. Shantz, S. Mahadevan, Uncertainty quantification in fatigue crack growth prognosis, Int. J. Progn. Health Manag. 2 (1) .
[29] C.R. Farrar, N.A. Lieven, Damage prognosis: the future of structural health monitoring, Philos. Trans. R. Soc. A: Math. Phys. Eng. Sci. 365 (1851) (2007) 623–632.
[30] A. Coppe, R.T. Haftka, N.H. Kim, F.-G. Yuan, Uncertainty reduction of damage growth properties using structural health monitoring, J. Aircr. 47 (6) (2010) 2030–2038.
[31] J. Gu, D. Barker, M. Pecht, Uncertainty assessment of prognostics of electronics subject to random vibration, in: AAAI Fall Symposium on Artificial Intelligence for Prognostics, 2007, pp. 50–57.
[32] H. Liao, W. Zhao, H. Guo, Predicting remaining useful life of an individual unit using proportional hazards model and logistic regression model, in: 2006 Reliability and Maintainability Symposium, RAMS'06, 2006, pp. 127–132.
[33] B. Saha, K. Goebel, Uncertainty management for diagnostics and prognostics of batteries using Bayesian techniques, in: 2008 IEEE Aerospace Conference, 2008, pp. 1–8.
[34] M. Orchard, M. Cerda, B. Olivares, J. Silva, Sequential Monte Carlo methods for discharge time prognosis in lithium-ion batteries, Int. J. Progn. Health Manag. 3 (2012) 1–12.
[35] B.E. Olivares, M.A. Cerda, M.E. Orchard, J.F. Silva, Particle-filtering-based prognosis framework for energy storage devices with a statistical characterization of state-of-health regeneration phenomena, IEEE Trans. Instrum. Meas. 62 (2) (2013) 364–376.
[36] D.A. Pola, H.F. Navarrete, M.E. Orchard, R.S. Rabié, M.A. Cerda, B.E. Olivares, J.F. Silva, P.A. Espinoza, A. Pérez, Particle-filtering-based discharge time prognosis for lithium-ion batteries with a statistical characterization of use profiles, IEEE Trans. Reliab. 64 (2) (2015) 710–721.
[37] M. Daigle, K. Goebel, Model-based prognostics under limited sensing, in: 2010 IEEE Aerospace Conference, 2010, pp. 1–12.
[38] M. Daigle, A. Saxena, K. Goebel, An efficient deterministic approach to model-based prediction uncertainty estimation, in: 2012 Annual Conference of the Prognostics and Health Management Society, 2012, pp. 326–335.
[39] S. Sankararaman, M. Daigle, A. Saxena, K. Goebel, Analytical algorithms to quantify the uncertainty in remaining useful life prediction, 2013 IEEE Aerospace Conference, 2013, pp. 1–11.
[40] A. Saxena, J. Celaya, B. Saha, S. Saha, K. Goebel, Evaluating prognostics performance for algorithms incorporating uncertainty estimates, in: Proceedings of 2010 IEEE Aerospace Conference, 2010, pp. 1–11.
[41] A. Dawn, J.-H. Choi, N.H. Kim, Prognostics 101: a tutorial for particle filter-based prognostics algorithm using matlab, Reliab. Eng. Syst. Saf. (2013) 161–169.
[42] E. Zio, G. Peloni, Particle filtering prognostic estimation of the remaining useful life of nonlinear components, Reliab. Eng. Syst. Saf. (2010) 403–409.
[43] A.S. Sarathi, B. Long, M. Pecht, Prognostics method for analog electronic circuits, in: 2012 Annual Conference of the Prognostics and Health Management Society, 2012, pp. 1–7.
[44] Y. Hu, P. Baraldi, F. Di Maio, E. Zio, A particle filtering and kernel smoothing-based approach for new design component prognostics, Reliab. Eng. Syst. Saf. (2014) 19–31.
[45] M. Yu, D. Wang, A. Ukil, V. Vaiyapuri, N. Sivakumar, Model-based failure prediction for electric machines using particle filter, in: 13th International Conference on Control, Automation, Robotics & Vision, 2014, pp. 1811–1816.
[46] D. Acuña, M. Orchard, J. Silva, A. Pérez, Multiple-imputation-particle-filtering for uncertainty characterization in battery state-of-charge estimation problems with missing measurement data: performance analysis and impact on prognostic algorithms, Int. J. Progn. Health Manag. 6 (008) (2015) 1–12.
[47] C. Chen, G. Vachtsevanos, M. Orchard, Machine remaining useful life prediction: an integrated adaptive neuro-fuzzy and high-order particle filtering approach, Mech. Syst. Signal Process. 28 (2012) 597–607.
[48] J. Duník, O. Straka, M. Šimandl, Sigma-point set rotation in unscented Kalman filter: analysis and adaptation, in: Proceedings of the 19th IFAC World Congress, 2014, pp. 5951–5956.
[49] D. Mortari, On the rigid rotation concept in *n*-dimensional spaces, J. Astronaut. Sci. 49 (3) (2001) 401–420.
[50] D. Pola, An improved prognosis strategy with temperature-dependent state space models for the analysis of the state-of-health and state-of-charge in lithium-ion batteries (M.Sc. thesis), Department of Electrical Engineering, University of Chile, 2014.
[51] H. Navarrete, Caracterización Estadística del Perfil de Uso de Baterías para el Pronóstico del Estado-de-Carga, Memoria de Título, Department of Electrical Engineering, University of Chile, 2014.