Auxiliary Particle Filter for Prognostics and Health Management

Hang Xiao, Jamie B. Coble, and J. Wesley Hines

Department of Nuclear Engineering, University of Tennessee

863 Neyland Drive, Knoxville, TN, 37996, USA

hxiao3@vols.utk.edu
jamie@utk.edu
jhines2@utk.edu

ABSTRACT

Accurately predicting the remaining useful life (RUL) of a system is a crucial factor in prognostics and health management (PHM). This paper introduces an auxiliary particle filter (APF) model, which has the advantages of dynamically updating the model parameters and being optimized in computational speed for prognosis applications in real engineering problems. The development of particle filter (PF) in the recent decade focused on increasing the PF model's complexity to solve more difficult problems. However, the added complexity negatively impacts the computational speed. The number of particles is commonly reduced to compensate for this increased computational burden, but this significantly reduces the accuracy of PF's posterior distribution. The developed APF model can estimate unknown states and model parameters at the same time with a large number of particles. This algorithm was demonstrated with a dataset from an electric motor accelerated aging experiment. The results show that this model can quickly and accurately predict the RUL and is robust to measurement noise.

1. INTRODUCTION

Prognostics and health management (PHM) is a process to assess, predict, and maintain the health condition of components, systems, and structures. It facilitates system health degradation models using monitoring data. PHM started to gain attention from both industry and academia in the 1970s (Pecht, 1991). It has proven to be an efficient way to reduce operational costs, increase system availability, and decrease system risks (Z. Zhang et al., 2018).

Remaining Useful Life (RUL) is defined as the reasonable time remaining on a component or a system that still performs at the desired specification (Si et al., 2011). RUL estimation is a critical factor in PHM. Accurately predicting the RUL prevents corrective maintenance, reduces system downtime, and transitions the maintenance strategy from time-based maintenance to condition-based maintenance. This can directly improve operational efficiency and reduce lifecycle costs (Jardine et al., 2006).

There are a large variety of simulations and modeling tools available for PHM purposes. Depending on the data available and knowledge of the predicted system, different methods can be deployed. The prognostic algorithms fall into two categories: experience-based approaches and data-driven approaches. Experience-based methods require system physics to be well-understood. Fuzzy systems (Alamaniotis et al., 2014) or expert systems (Q. Zhang et al., 2015) compare the similarity between an observed situation and predefined failure trends. Those experience-based models correlate expert knowledge and engineering experience with the measurements to estimate RUL. The model is usually straightforward to interpret. Experience-based approaches can yield accurate and reliable predictions if the system and its degradation are well understood. However, in most systems, especially complex systems, fault modes, and physical degradation processes are not well-understood, limiting the usage of experience-based approaches.

Data-driven approaches do not necessarily require a physical understanding of the system, making them more flexible for RUL estimation. Regression-based methods are commonly used, especially in industry, because they are easy to interpret. However, these kinds of methods require a monotonic assumption or stepwise approximation to find RUL (Park & Bae, 2010). This model may generate a conservative estimation if the degradation process is not always monotonic (Si et al., 2011). Deep learning methods (Li et al., 2018) use neural networks to solve problems with a massive dataset. Neural network-based models require a significant amount of data to fine-tune the model. Lacking proper data is a universal phenomenon in the research of PHM. It is hard to guarantee the robustness and accuracy of

https://doi.org/10.36001/IJPHM.2023.v14i2.3485

neural network-based models without sufficient data. Stochastic process models, such as Wiener processes (Tseng & Peng, 2004), Gamma processes, and inverse Gaussian processes (Wang & Xu, 2010) have straightforward mathematical calculations and the physical meanings are easy to understand. However, the numeric computation requires a long run time, limiting the usage for online prognostics. Markovian-based models assume that the future degradation state depends only on the current state. However, these models rely on a strong assumption that each state is conditionally independent. Filtering-based methods (Orchard & Vachtsevanos, 2009), such as Kalman Filter and Particle Filter (PF) solve state-space models for the unobserved degradation process with observed measurement data.

In recent decades, PF has drawn a lot of attention because it can be applied in a broad range of engineering problems (Baraldi et al., 2015; G. Kim et al., 2018; Zio & Peloni, 2011). This Bayesian-based framework is designed for nonlinear and non-gaussian state space model filtering (Gordon et al., 1993). As PFs become more common for prognostic modeling, many modifications are being made for different applications. The typical model simulates or approximates the model parameters before the model initialization step and subsequently does not change the predicted model parameter. This method relies on a strong assumption that the model parameter is static. In real engineering problems, equipment operating conditions, such as environmental temperatures, moisture, and atmospheric pressure, may differ every day. The changing operational conditions might lead to a variable model parameter. Even though the model parameter is static in some cases, the prediction results heavily depend on how accurately the parameter is estimated beforehand, which increases the model uncertainty (Jouin et al., 2016).

The augmented PF (Hu et al., 2015) considers the model parameters as state vectors and uses another state transition function to estimate the model parameters. This method dynamically estimates the model parameter and is applicable to engineering problems with variable model parameters. However, there is no way to quantify the accuracy of the model parameters' state transition functions.

In this paper, an Auxiliary Particle Filter Prognostic Model (APFPM) is proposed to estimate states and parameters simultaneously and apply it in prognostics. This method also treats model parameters as states. First, it performs a classical PF, and then it samples the model parameters in an auxiliary step. In the auxiliary step, model parameters are sampled from a distribution, and the posterior is updated. The weight is then calculated based on the new posterior.

One drawback of PF is it requires more computational resources than other filters. Based on the classical PF model, a significant number of evolved PF models have appeared in the last 20 years. A risk sensitive PF-based prognostics model (Orchard et al., 2010) incorporates a cost function in the importance distribution to generate more particles in the regions with higher cost (possibility). It best fits systems with an anticipated sudden change in the operating conditions. Rabiei et al. (2018) also optimized the PF algorithm with relative entropy; Yu (2017) proposed a prognostic model based on logistic regression and PF; Daroogheh et al. (2017) used neural networks to train the PF model. However, the recent efforts to modify PFs continue to increase the complexity of the model, which makes it more challenging to execute without further increasing the computational cost.

The number of particles is a critical PF parameter. The rate of convergence of the approximate posterior probability distribution is inversely proportional to the square root of particle numbers (Bain & Crisan, 2009). In other words, the more particles, the more accurately the filter approximates the posterior distribution (Elvira et al., 2017). However, increasing the number of particles increases the computational burden significantly. When the model is complex, it is common to sacrifice the number of particles to manage the computational load. Daroogheh et al. (2017) only uses 150 particles for simulation, while Yu (2017) uses 500 particles. Recent applications of PFs in prognostics introduced more degrees of freedom into the model, which significantly slow down the simulation. Researchers have decreased the number of particles to compensate for the extra complexity, but it also reduced the simulation accuracy.

The most time-consuming part of PF is resampling. This paper compares different resampling methods and selects the systematic resampling for APFPM. It significantly reduces the time complexity compared with other common resampling methods, which makes APFPM capable of estimating the RUL using more than 100,000 particles.

This model was demonstrated with data from an accelerated aging experiment on three-phase electrical motors. Induction motors are widely applied in the industry due to their robustness, reliability, and low maintenance requirements (Djeddi et al., 2007). Bearing failure is the most common failure mode for the electric motor, and it happens due to excessive load, high temperature, lousy lubrication, etc. (Gnanaprakasam & Chitra, 2014). A series of accelerated run-to-failure aging experiments was conducted to simulate degradation and extremely noisy measurement data were collected by low-precision sensors (Sharp, 2012; Barbieri et al., 2015).

The paper is organized as follows. Section 2 introduces the methodology of APFPM used for RUL estimation. Section 3 describes the experimental dataset and prognostic parameter generation process. Section 4 presents the results of applying APF to estimate RUL of the experimental dataset and compares it with naïve PF. Section 5 summarizes this work and highlights areas of future work.

2. METHODOLOGY

Hidden Markov Models (HMM), or state-space models, are widely used for RUL estimation. A more detailed review of HMM can be found in (Cappé, 2005). They can be characterized by the following framework:

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

$$z_k = h(x_k, v_k) \tag{2}$$

where x_k and z_k are hidden degradation state and observed state at time k, respectively; ω and v are random process noise and random measurement noise, respectively; $f(\cdot)$ is the physical state model of the degradation process; and $h(\cdot)$ is the measurement equation.

Given the initial distribution $p(x_0)$, the degradation state x_k can be characterized by a transition probability $p(x_k|x_{k-1})$. As shown in Eq. (2), noisy measurements z_k are assumed to be conditioned on the current degradation state x_k. Thus, the objective of solving this HMM is to address the posterior distribution $p(x_k|z_{1:k})$, denoting that $z_{1:k} \triangleq \{z_1, \cdots, z_k\}$ represent the measurements up to time k.

Two sequential steps can solve this problem: the prediction step and the update step. In the prediction step, the prior state pdf $p(x_k|z_{1:k})$ can be estimated using the Chapman-Kolmogorov equation:

$$\begin{aligned} &p(x_k|z_{1:k-1}) \\ &= \int p(x_k|x_{k-1}) \cdot p(x_{k-1}|z_{1:k-1}) dx_{k-1} \end{aligned} \tag{3}$$

The update step uses Bayes Rule to calculate the posterior state pdf $p(x_k|z_{1:k})$; the recursive factorization of the posterior state pdf is given as:

$$\begin{aligned} &p(x_k|z_{1:k}) \propto p(z_k|x_k) \cdot \\ &\int p(x_k|x_{k-1}) \cdot p(x_{k-1}|z_{1:k-1}) dx_{k-1} \end{aligned} \tag{4}$$

However, the integrals in Eq. (3) and Eq. (4) are difficult to solve analytically. Filtering techniques provide an alternative approach to approximate their solutions. A PF is a Monte Carlo method with the capability to solve nonlinear and non-Gaussian systems.

2.1. State Estimation of Particle Filter

A PF uses a set of N particles $\mathrm{x}_k^j, j = 1, \cdots, N$ to approximate the posterior state pdf $p(x_k|z_{1:k})$ with weight w_k^j, where $\sum_{j=1}^{N} w_k^j = 1$ (Andrieu et al., 2001):

$$p(x_k|z_{1:k}) \approx \sum_{j=1}^{N} w_k^j \cdot \delta(x_k - x_k^j) \tag{5}$$

Particles and their corresponding weights are obtained with importance sampling (Andrieu et al., 2001; H. Kim et al., 2000). Samples x_k^j are drawn from an importance density $q(x_k|z_{1:k})$ and the weights are defined as:

$$w_k^j \propto \frac{p(x_k^j|z_{1:k})}{q(x_k^j|z_{1:k})} \tag{6}$$

The probability $p(x_k^j|z_{1:k})$ is estimated with Bayes Rule Eq. (7) and the importance density $q(x_k^j|z_{1:k})$ is presented in Eq. (8):

$$\begin{aligned} &p(x_k^j|z_{1:k}) \\ &= p(z_k|x_k^j) \cdot p(x_k^j|x_{k-1}^j) \cdot p(x_{k-1}^j|z_{1:k-1}) \end{aligned} \tag{7}$$

$$q(x_k^j|z_{1:k}) = q(x_k^j|x_{k-1}^j, z_{1:k}) \cdot q(x_{k-1}^j|z_{1:k}) \tag{8}$$

Combining Eq. (7) and Eq. (8), the weight can be updated from Eq. (6) to Eq. (9) in a recursive format:

$$w_k^j \propto w_{k-1}^j \frac{p(z_k|x_k^j) \cdot p(x_k^j|x_{k-1}^j)}{q(x_k^j|x_{k-1}^j, z_{1:k})} \tag{9}$$

The simplest form is to use the prior distribution $p(x_k^j|x_{k-1}^j)$ as the importance density, yielding:

$$w_k^j \propto w_{k-1}^j p(z_k|x_k^j) \tag{10}$$

However, a more reliable importance density was proposed in (Pitt & Shephard, 1999):

$$\begin{aligned} &q(x_k^j|x_{k-1}^j, z_{1:k}) \\ &\propto p(z_k|\mu_k^j) \cdot p(x_k^j|x_{k-1}^j) w_{k-1}^j \end{aligned} \tag{11}$$

where the mean of $p(x_k^j|x_{k-1}^j)$ is μ_k^j. To derive the importance density, the particle index *i* serves as an auxiliary variable (Chen et al., 2005). This method allows new particles to generate based on the current measurement and particles from the previous timestep. The weight is then estimated with this auxiliary method as

$$w_k^j \propto \frac{p(z_k|x_k^i)}{p(z_k|\mu_k^j)} \tag{12}$$

2.2. Parameter Estimation of Particle Filter

Traditional PFs define state transition equations based on a physical model. The model parameters are either known or can be regressed with historical data. Thus, state vectors and model parameters can be initialized to train the model. Maximum Likelihood (Wicker et al., 2008), Expectation-Maximization (Zhao et al., 2013) and a few other regression-based methods can be used to estimate the model parameters based on a batch of historical data. These methods cannot dynamically update the model parameter, causing the parameter to be re-estimated every time a new measurement becomes available. Previous calculations are not used in new estimations.

However, traditional PF methods rely on a strong assumption that the model parameters are static. Equipment operational conditions, such as environmental temperatures, moisture, and atmospheric pressure, may differ every day in real engineering problems. The changing operational conditions might lead to a variable model parameter. Even if the model parameter is static in some cases, the prediction results still heavily depend on how accurately the parameter is estimated beforehand, which can increase the model uncertainty (Jouin et al., 2016).

In some cases, not only is it necessary to estimate degradation states but it is also necessary to calculate the unknown parameters. Eq. (1) and Eq. (2) become Eq. (13) and Eq. (14) if both states and parameters are to be estimated:

$$x_k = f(x_{k-1}, \omega_k, \theta_{k-1}) \tag{13}$$

$$z_k = h(x_k, v_k, \theta_k) \tag{14}$$

given θ as the model parameter. The joint posterior distribution can then be written as Eq. (15) following Bayes' theorem:

$$\begin{aligned} &p(x_k, \theta_k | z_{1:k}) \\ &\propto p(z_k | x_k, \theta_k) \cdot p(\theta_k | z_{1:k-1}) \\ &\cdot p(x_{k-1} | \theta_k, z_{1:k-1}) \end{aligned} \tag{15}$$

There are two different approaches to define $p(\theta_k|z_{1:k-1})$ in Eq. (15). One way is to incorporate the Markov Chain Monte Carlo (MCMC) method into the PF to estimate the state and provide likelihood information for MCMC to approximate parameters (Andrieu et al., 2010). The joint MCMC and PF approach adds significant computational complexity to the model, which is not suitable for many industrial applications.

The other approach is to treat unknown parameters as states and use APF to estimate both states and parameters (J. Liu & West, 2001; Storvik, 2002). An et al. (2012) proposed to keep the distribution of model parameters unchanged during the prediction step to simulate the evolution of the model parameter. It is simple to implement, but the drawback of this method is that it is not applicable for variable parameter estimation. A proper way to sample the model parameter is required.

A Gaussian random walk method (Chen et al., 2005) is proposed that adapts parameters into new data:

$$\theta_k = \theta_{k-1} + \xi_k \tag{16}$$

where $\xi_k \sim G(0, W_k)$ is drawn from a Gaussian distribution with zero mean and W_k covariance matrix. Thus, $p(\theta_k|z_{1:k-1})$ in Eq. (15) can be approximated in the framework of PF as:

$$\begin{aligned} &p(\theta_k | z_{1:k-1}) \\ &= \sum_{j=1}^{N} w_{k-1}^j \cdot G(\theta_k | \theta_{k-1}^j, W_k) \end{aligned} \tag{17}$$

We can define V_{k-1} as covariance matrices of all weighted particles. The covariance in Eq. (17) is thus $V_{k-1} + W_k$, which increases over time. This results in more diffused particles.

A kernel smoothing method (J. Liu & West, 2001) uses the shrinkage rule to force the particles to be closer to their mean. The Kernel location m_{k-1}^j is specified as Eq. (18) with smoothing factor $h \in (0,1)$

$$\begin{aligned} &m_{k-1}^j \\ &= \sqrt{1-h^2}\theta_{k-1}^j + \left(1 - \sqrt{1-h^2}\right)\bar{\theta}_{k-1} \end{aligned} \tag{18}$$

where $\bar{\theta}_{k-1}$ is the parameter mean at time k-1. Thus, $p(\theta_k|z_{1:k-1})$ in (15) is approximated as

$$\begin{aligned} &p(\theta_k | z_{1:k-1}) \\ &= \sum_{j=1}^{N} w_{k-1}^j \cdot G(\theta_k | m_{k-1}^j, h^2 V_{k-1}) \end{aligned} \tag{19}$$

The mixture probability in Eq. (19) has a mean of $\bar{\theta}_{k-1}$ and a covariance matrix V_{k-1}. This kernel smoothing method prevents the covariance from increasing and is ideal for parameter estimation. The weight is estimated as

$$w_k^j \propto \frac{p(z_k | x_k^i, \theta_k^j)}{p(z_k | \mu_k^j, m_{k-1}^j)} \tag{20}$$

2.3. Sequential Importance Resampling (SIR)

Particle degeneracy is the main problem encountered for the naïve PF model. After a few iterations, more and more weights tend to be zero, effectively leaving only a few particles. Degeneracy occurs when some particles lose track of the actual degradation state, which results in increased weight variance and a more skewed importance weight distribution (Andrieu et al., 2001).

Sequential Importance Resampling (SIR) is the solution to particle degeneracy (Doucet et al., 2000). A resampling algorithm samples a new set of particles x_k^m for $m = 1, \cdots, N$ to replace x_k^j. After the particles are resampled, the particle weights are set to 1/N.

There are various resampling algorithms. A good algorithm should select a representative population of particles with high probability and give enough consideration to low probability particles so that nonlinear behavior can be detected as well. Another essential metric to select a resampling algorithm is that the algorithm should be computationally efficient.

Multinomial resampling (Doucet et al., 2000) is a secure algorithm that is commonly used in many PF tutorials. It computes the cumulative distribution function (CDF) of the weights. A random number between 0 and 1 is generated uniformly. For each random number, this algorithm uses a binary search in the CDF of weights to find where the corresponding weight is. The corresponding particle's array with the selected weights is chosen to form the new array of particles, x_k^m. This process has to be repeated *N* times until all spaces in x_k^m are filled.

Stratified resampling (Andrieu et al., 2001; Kitagawa, 1996) divides the weight CDF into *N* spaces, and then selects one particle from each space randomly. This method ensures that particles with significant weight are appropriately sampled, and all chosen particles are separated by $\left[0, \frac{2}{N}\right]$.

Systematic resampling (Kitagawa, 1996) is very similar to stratified resampling. It divides the weight CDF into *N* spaces with a random offset and ensures that all selected particles are apart precisely by $\frac{1}{N}$.

Residual resampling (J. S. Liu & Chen, 1998) calculates a particles' weight index using the weight of each particle divided by the normalized weight *N-1*. For particles with a weighted index greater than *1*, they are sampled *t* times, in which *t* equals the integer part of the weight index. It is not guaranteed to select *N* particles; the remaining spaces of x_k^m are filled with other resampling methods mentioned above.

The most significant disadvantage of PFs is that it is computationally expensive. Resampling is the most time-consuming part of PF algorithms. Time complexity is usually introduced in computer science to describe an algorithm's computational complexity by counting the number of elementary operations performed. It utilizes the big O notation to represent the worst-case scenario computational complexity. This notation was adopted to calculate the time complexity of each resampling algorithm.

Multinomial resampling requires generating *N* random numbers and performs a binary search in space for *N* times; therefore, it has $O(n \log n)$ time complexity per iteration. Since the sampling is entirely random, it cannot guarantee that large weights are adequately sampled. Stratified and systematic resampling have a similar sampling distribution in that all parts of particle spaces are sampled, and large weights are sampled more often. However, stratified resampling also requires the generation of *N* random numbers and performs a binary search for *N* times, which makes it have the same time complexity as multinomial resampling. Systematic resampling only needs to generate one random number and compares *N* times only at each space boundary. It has a time complexity of $O(n)$, which is much faster than either the multinomial or stratified resampling method. The only drawback is that systematic resampling samples particles uniformly, which is not as good as stratified resampling in ensuring more resampling for higher weights. Residual resampling sacrifices some stochasticity to ensure that all large weights are sampled. Experiments showed that half of the particles are determined by weight index (Hol et al., 2006), and the other half of the particles have to be determined using other resampling methods. Residual resampling does not distribute samples across the entire CDF evenly and cannot guarantee sampling for reasonably large particles. It is hard to tell the time complexity of residual resampling. It depends on which resampling algorithm is used to deal with the remaining spaces after heavier weights are resampled.

Most PF-based prognostic model researchers use a multinomial resampling method, which is the slowest method. Yu (2017) used residual resampling, and Dong et al. (2014) proposed a resampling procedure based on support vector regression (SVR). The SVR method usually requires $O(N^3)$ time complexity, which is much slower than most of the resampling methods, but optimization may be used to speed up the SVR resampling process. It is extremely important to use fast resampling method to boost the computational speed.

In most cases, it is not always necessary to resample at each iteration. It is only required to resample occasionally, when the weight variance exceeds a certain threshold. Effective sample size N_{eff} (Doucet et al., 2000; Kong & Liu, 1994) is used in Eq. (21) to measure the weight variance. When the effective sample size is below a predefined threshold, resampling is performed.

$$N_{eff} = \frac{1}{\sum_{j=1}^{N} w_k^j} \qquad (21)$$

Resampling Method	Time Complexity for Single Time Resampling	Particle Sampling Distribution
Multinomial Resampling	$O(n \log n)$	No guarantee either large particles or reasonably large particles are properly sampled
Residual Resampling	Residual part $O(n)$, the rest part depends on implemented method	Large particles are well sampled but reasonably large particles may not be sampled properly
Stratified Resampling	$O(n \log n)$	Both large and reasonably large particles are well sampled
Systematic Resampling	$O(n)$	Even distribution on weight space, similar distribution as stratified resampling,

		but not as good as stratified resampling in sampling large particles

Table 1. Resampling method summaries

2.4. Prognosis Using Particle Filter

Prognostics describes the long-term degradation of the system. PFs can predict the future degradation trajectory and the corresponding particle distribution for each time step. This application goes beyond the traditional filtering method's horizon because there are no new measurements for Bayesian updating. The predicted posterior PDF $p(x_{t+l}, \theta_{t+l}|z_{1:t})$ at time $t+l$ is calculated from the last updated posterior PDF $p(x_t, \theta_t|z_{1:t})$ and propagated using the state equation until time $t+l$. The computation of $p(x_{t+l}, \theta_{t+l}|z_{1:t})$ is shown (Doucet et al., 2000):

$$\begin{aligned} &p(x_{t+l}, \theta_{t+l}|z_{1:t}) \\ &= \int \cdots \int \prod_{j=t+1}^{t+l} p(x_j, \theta_j|x_{j-1}, \theta_{j-1}) \\ &\cdot p(x_t, \theta_t|z_{1:t}) \prod_{j=t}^{t+l-1} dx_j \end{aligned} \tag{22}$$

Orchard et al. (2008) provided the method of computing the above equation, but it is too computationally expensive to solve the above equation numerically. Doucet et al. (2000) proposed to neglect the error generated by particles from time t to $t+l$. Therefore, the posterior PDF can be estimated at time $t+l$ using

$$\begin{aligned} &p(x_{t+l}, \theta_{t+l}|z_{1:t}) \\ &\approx \sum_{j=1}^{N} w_t^j \cdot \delta(x_{t+l} - x_{t+l}^j) \end{aligned} \tag{23}$$

Each particle state x_{t+l}^j is calculated by applying the degradation state equation from x_{t+l-1}^j.

At time *t*, when the last measurement data z_t is processed by the model, the PF model converts from training to the prognosis step. The parameter θ_t^j becomes unchanged in the prognosis step, and the degradation model propagates until all particles exceed a predefined threshold. The time of failure (TOF) can be determined when the mean of the weighted particles μ_k^j exceeds the threshold, while a 95% confidence interval is recorded when 2.5 percentile and 97.5 percentile of particles exceeds the threshold. The predicted RUL distribution can be calculated as in Eq. (24) (Zio & Peloni, 2011):

$$p(RUL) \approx \sum_{j=1}^{N} w_{TOF}^j \delta(RUL - RUL_{TOF}^j) \tag{24}$$

2.5. Auxiliary Particle Filter Prognostics Model Framework

The Auxiliary Particle Filter Prognostics Model (APFPM) framework allows prognostics with an unknown physical model and disparate data sources. The assumptions of APFPM require the following information to be available as model input:

(1) A degradation state transition model to estimate the evolution of states. PF-based models use a physical model to estimate state posteriors. Dynamic updating of states and model parameters adds a lot of flexibility to the model. However, a rough estimate of the degradation state transition model from either visual inspection or regression is required.

(2) A measurement transition model that maps the observed measurements to the degradation state *x*. This step can be done in data extraction and manipulation. Raw data are collected from sensors and need to be processed and fused into a prognostic parameter, also known as a health index. The prognostic parameter simplifies the measurement transition model and can be considered to be a degradation state.

(3) A failure threshold that defines the boundary between the acceptable working state and the failure state. When the degradation prediction crosses the failure threshold, the system is considered to be failed. The RUL is estimated by subtracting current time from the TOF.

(4) All historical measurements and their corresponding measurement time from *0* to time *t*. Measurements are process data that can be accessed with the condition monitoring system. For online prognostics, it includes all historical measurements and new measurements. Measurements are mapped into degradation states via measurement transition models.

The APFPM has three major steps: initialization step, model training step, and prognosis step. In the initialization step, the initial particles and model parameters are sampled, and constants are determined based on engineering judgment. It usually requires the initial particles and model parameters to be sampled uniformly unless they have other known distributions. An uncertainty test is required to determine the variance of state transition and measurement noise level.

The model training step occurs when new measurements are available. Eq. (11) is adopted as the importance density function and the mean of $p(x_k^j|x_{k-1}^j, \theta_{k-1}^j)$ is calculated as μ_k^j and the kernel location of parameters m_{k-1}^j as Eq. (20). Then, $p(z_k|\mu_k^j, m_{k-1}^j)$ can be estimated.

As mentioned in Section 2.2, in the auxiliary step, an auxiliary integer variable i is sampled from the range of [1, *N*], such that

$$\Pr{}_k^{i=j} = w_k^i \cdot p(z_k | \mu_k^i, m_{k-1}^i) \tag{25}$$

New model parameters θ_k^i are then sampled from the i^{th} normal component kernel density using a Gaussian distribution $G\left(\cdot \,|m_{k-1}^j, h^2 V_{k-1}\right)$.

Particle states x_k^i can then be sampled. It is important to note that x_k^i is sampled from the state transition function using x_{k-1}^i and θ_k^i instead of x_{k-1}^i and θ_{k-1}^i, because the final weight, calculated with Eq. (20), is required to test the accuracy of both state and model parameters. Since the new parameter is sampled in the auxiliary step, the weight needs to verify that those parameters are sampled correctly.

Now $p(z_k | x_k^i, \theta_k^j)$ can be estimated to further calculate particle weight w_k^i using Eq. (20). Particle weights are normalized and then be used to calculate the effective number of particles, as in Eq. (21). If the effective number of particles falls below a pre-defined threshold, SIR is performed, and a systematic resampling method is used.

In the prognosis step, the posterior distribution is calculated with Eq. (23). For simplicity, it is assumed that the error generated by particles from time t to $t + l$ is negligible. There is another important assumption in the prognosis step. It is also assumed that the model parameter is changing in the model training step, but the model parameter cannot be updated without a new measurement. The model parameter used in the prognosis step is fixed to the last estimated model parameter. Thus, in the prognosis step, it is assumed that the model parameter does not change substantially from the last estimated model parameter so that the RUL does not deviate significantly from the true value. The pseudo-code of APFPM is here:

```
1. Initialization Step
Sample $x_0^j$ and $\theta_0^j$ uniformly from a predefined range
$k = 1$
2. Model Training Step
2.1 While $k \leq T$, T is the time that the last measurement data is obtained.
For j=1:N
    Calculate $\mu_k^j = E(x_k^j | x_{k-1}^j, \theta_{k-1}^j)$
    Calculate $m_{k-1}^j$ using (18)
    Calculate $p(z_k | \mu_k^j, m_{k-1}^j)$
End
2.2 Normalize weight $w_k^j$
2.3 Determine if $N < N_{eff}$, if so, perform systematic resampling (step 2.4), if not, go to step 2.5
2.4 Auxiliary SIR
For i=1:N
    Sample an auxiliary integer variable $i \in \{1:N\}$ such that
    $\Pr{}_k^{i=j} = w_k^i \cdot p(z_k | \mu_k^i, m_{k-1}^i)$
    Sample new $\theta_k^i$ from ith normal component of kernel density using Gaussian distribution.
        $\theta_k^i \sim G\left(\cdot \,|m_{k-1}^j, h^2 V_{k-1}\right)$
    Sample $x_k^i \sim p\left(\cdot \,|x_{k-1}^j, \theta_k^i\right)$
    Calculate $p(z_k | x_k^i, \theta_k^j)$
    Assign weight $w_k^i$ using (20)
End
2.5 Normalize $w_k^{j=i}$
$k = k + 1$
End
3. Prognosis Step
While $\exists x_k^j > p$, where $p$ is the threshold of failure
        Propagate $x_k^j \sim p\left(\cdot \,|x_{k-1}^j, \theta_T^j\right)$
        $k = k + 1$
End
```

3. MODEL APPLICATION ON EXPERIMENTAL DATA

3.1. Experimental Data Set Description

Accelerated degradation experiments were performed with a series of five horsepower (HP), three-phase, and 220-volt squirrel cage induction motors (Sharp, 2012). Those motors were baked in an oven at 160°C for three days. The motors were then placed in a condensation chamber in a sealed container for a total of 48 hours at 100% humidity. Then those motors were placed back in the oven for a 24-hour heating process to get rid of any residual moisture, also at 160°C. Following the second heating process, the motors were cooled and run for an hour with data collected from an array of sensors. A single cycle took slightly over one week, and the motor underwent thermal and moisture induced degradation. The accelerated degradation testing was adapted from the studies of Upadhyaya et al. (1997) and as suggested by the relevant IEEE Standard ('IEEE Standard Test Procedure for Evaluation of Systems of Insulating Materials for Random-Wound AC Electric Machinery', 1974). The detailed experiment plan can be found in (Barbieri et al., 2015). The list of collected dataset variables is shown in Table 2.

Number	Variable Description
1	Motor Current x (Amperes)
2	Motor Current y (Amperes)
3	Motor Current z (Amperes)
4	Motor Voltage x (Volts)
5	Motor Voltage y (Volts)
6	Motor Voltage z (Volts)
7	X Direction Accelerometer (g)
8	Y Direction Accelerometer (g)
9	Industrial Microphone (g)
10	Tachometer (RPM)
11	Speed (m/s)
12	Output Current (Amperes)
13	Output Voltage (Volts)
14	Temperature (°C)

Table 2. Raw Data Variables from Motor Testbed

The data were collected at 10,240Hz for two seconds every 15 minutes in a one-hour collection period. Thus, four data files were obtained in one experimental cycle. In the analysis, data from five different motors were used. Each motor was run-to-failure, and the number of tests (degradation cycles) for each motor is reported in Table 2. In the analysis, there are a few data files that are incredibly faulty due to data collection anomalies. Those outliers were excluded from future data analysis. The number of cycles in the cleaned test to failure data is also shown in Table 3. Each test represents a two-second snapshot of data collection.

3.2. Feature Extraction and Prognostic Parameter Generation

Both time and frequency domain features were extracted from the original data. Time-domain features were extracted for all variables. These features are: variance, kurtosis, mean, skewness, rooted mean square (RMS), and the ratio of peak magnitude to RMS. Variance and kurtosis of motor voltage z and kurtosis of motor voltage x were excluded because they have a few huge peaks, which affected the performance of later Principal Component Analysis (PCA) analysis.

Frequency domain features were extracted for X and Y direction vibrational data. Given the possible bearing fault band, as shown in Table 4, peak tracking for these three frequency ranges were extracted (Barbieri et al., 2015). RMS of both X and Y directions on these three faulty frequency ranges were also calculated. Other frequency domain features, including peak to peak, shape factor, impulse factor, and margin factor, were also extracted. There are in total 97 different features extracted from the data set; Table 5 gives a complete list of extracted features.

Motor	Tests to Failure	Cleaned Tests to Failure
1	108	102
2	103	98
3	110	101
4	109	109
5	107	102

Table 3. Tests to failure data for each motor

Bearing Fault	Fault Frequency (Hz)
Ball pass frequency of inner race	325
Ball pass frequency of outer race	215
General ball pass frequency	283

Table 4. Bearing Fault and Corresponding Frequency

Dimensionality reduction techniques reduce the number of variables and thus the complexity of the problem. Two-step dimensionality reduction methods were utilized in this research. The first step was to perform a feature Importance Ranking Test (IRT). Variables with high ranks can be retained. Feature importance is ranked by monotonicity based on the algorithm shown in Eq. (26) because monotonicity is the most intuitive feature for degradation trend.

$$Importance\ Ranking = \sum_{l=1}^{M-1} \frac{sgn[x(l+1)-x(l)]}{M-1} \tag{26}$$

where M is the number of measurements in a variable. The result of importance ranking is shown in Fig. 1. Variables with IRT values greater than 0.1 were retained for future analysis. The number of variables was reduced from 97 to 26.

Feature	Number of features applied
Variance	13 (exclude variable 6 in Table 2)
Kurtosis	12 (exclude variable 4 and 6 in Table 2)
Mean	14
Skewness	14
RMS	14
Peak Magnitude to RMS	14
Peak Tracking of Faulty Frequency Range	6
RMS of Faulty Frequency Range	2
Peak to Peak	2
Shape Factor	2
Impulse Factor	2

Margin Factor	2

Table 5. List of features extracted for analysis

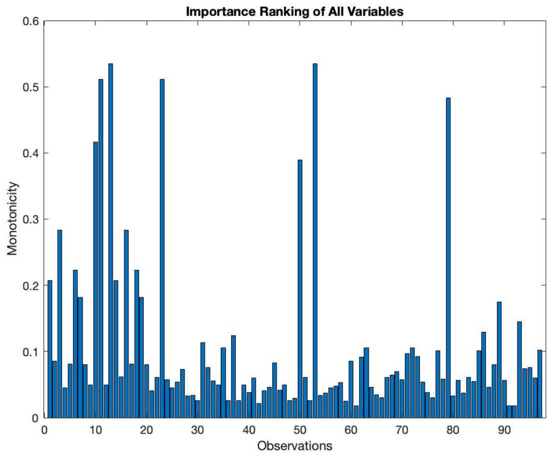

Figure 1. Importance Ranking of all variables extracted from raw data

The second step used PCA to reduce the dimensionality further. PCA is a linear technique that maps variables to a low-dimensional space. It is a useful procedure to reduce the dimensionality of collinear data without losing much information. The first Principal Component (PC) contains the maximum variance of the data set, and the next PC always has less variance than the preceding PC. Remaining variables from IRT were mapped to form PCs, and 18 PCs are retained that contained 90% of the variance.

A Genetic Algorithm (GA) is a stochastic optimization method that has successfully been applied to generate prognostic parameters (Coble & Hines, 2009). A GA was used to construct an optimal prognostic parameter through the maximization of three prognostic metrics: monotonicity, prognosability, and trendability and prognostic parameter metrics score is calculated (Coble, 2010). First, a Savitzky-Golay filter (Savitzky & Golay, 1964) was used to smooth the data. Finally, the output from GA was normalized between 0 and 1 to form the prognostic parameter. Fig. 2 shows the prognostic parameter of 5 motors. The diagram of prognostic parameter generation steps can be found in Fig. 3.

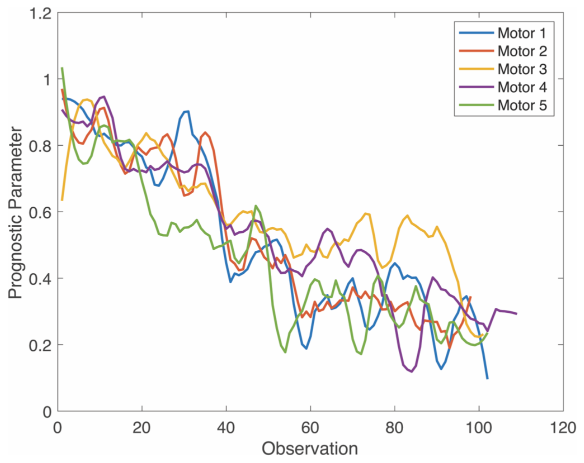

Figure 2. Prognostic Parameter of 5 Motors

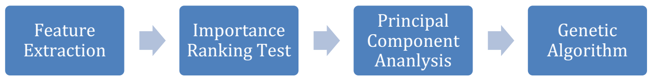

Figure 3. Steps of Prognostic Parameter Generation from Raw Data

From the generated prognostic parameter, the degradation process starts from the beginning of the lifecycle. Different motors degrade at different rates from each other, but in general, they follow a similar degradation path. The prognostic parameter is not very smooth and linear, which makes a PF an ideal method to model this data.

Monotonicity is the most direct and relevant metric for degradation of induction motors. It is expected that these components degrade over time and have no self-healing. The IRT selection phase eliminated a majority of variables that do not have a strong monotonic trend over time. From another aspect, if IRT were skipped and PCA was applied to the entire 97 variables, the degradation-irrelevant variance would be taken into consideration when mapping those variables into PC spaces. It would not only affect the useful variance in the generated PCs but also force the model to contain more PCs to capture (potentially irrelevant) information. Table 6 shows a comparison of prognostic parameter metric scores with a few different data processing methods using the same motor data. It shows that combining IRT, PCA, and GA can generate the best prognostic parameter for the current data set from the choices considered.

Data Processing Method	**Prognostic Parameter Metrics Score**
Only GA	1.35
Only IRT and GA	2.37
Only PCA and GA	1.65
IRT, PCA, and GA	2.76

Table 6. Comparison of Prognostic Parameter Metrics Score with Different Data Processing Methods

4. PARTICLE FILTER MODEL FOR PROGNOSIS

There is no well-recognized physical model to simulate a complex system like motors. In this research an exponential model is selected to model the degradation:

$$y = a\exp\left(-bt\right) \tag{27}$$

where a and b are unknown model parameters. According to Eq. (13), the degradation model Eq. (27) can be rewritten in the following recursive format:

$$x_k = \exp\left(-b_k \cdot \Delta t\right) \cdot x_{k-1} \tag{28}$$

In the prognostic parameter generation step, all parameters are normalized to have a starting point of 1, indicating the system is completely healthy. By assuming an arbitrary uncertainty, the state x_0 is uniformly sampled from [0.85, 1.15]. By roughly calculating the model parameter, it is around 0.013, that is in the range of [0.01, 0.02]. Then the parameter b_0 is uniformly sampled from [0.01, 0.02]. Model measurement noise standard deviation σ was set to be 0.1. Process noise ω_k was already incorporated in the uncertainty of the model parameter so that it can be ignored. The smoothing factor h was set to 0.2, as suggested by (J. Liu & West, 2001). The likelihood function of measurement, shown in Eq. (29), assumes that the measurement noise v_k in Eq. (14) is normally distributed

$$L\left(z_k \middle| \mu_k^j, m_{k-1}^j\right) = \frac{1}{\sqrt{2\pi}\sigma} exp\left(-\frac{\left(z_k - \mu_k^j\right)^2}{2\sigma^2}\right) \tag{29}$$

The number of particles was chosen to be 100,000, compared with other research that use 30 to 5,000 particles. Computational speed for multinomial, residual, and systematic resampling method was tested using our dataset. It showed that systematic resampling is around 500 times faster than multinomial. Systematic resampling is also significantly quicker than residual resampling, which is why a systematic resampling algorithm was used in our model.

Models were trained for each of the five motors separately using a leave one out method. When one is trained, the failure data from the remaining four motors are used to calculate the failure threshold. The failure threshold was determined at the 50th percentile among the four remaining motors' prognostic parameter value at failure.

The mean and standard deviation of the weighted particles were also calculated for each iteration to determine the simulated degradation trajectory. A 95% confidence interval was also calculated.

Fig. 4 shows the predicted degradation compared with the actual prognostic parameter. The model was trained until the 75th cycle. The mean of the particles in the training step, representing the degradation trajectory, is shown with the red line. The tail of the simulated curve, shown by the green line, is the mean of the particles in the prognosis step, representing the predicted degradation trajectory. The time of failure is achieved when the mean of the particles crossed the threshold, as shown by the black dashed line equal to ~0.24. The blue dashed lines show the 95% confidence intervals.

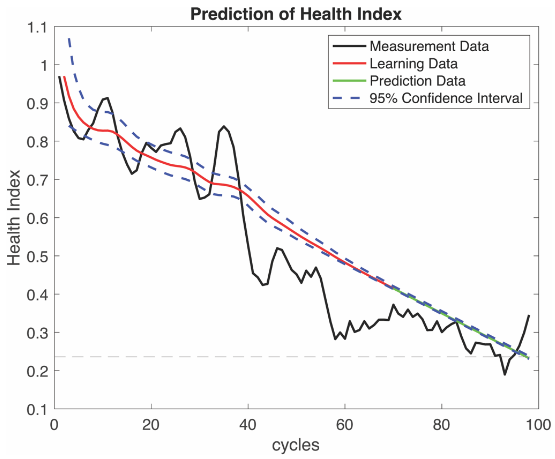

Figure 4. Motor 2 prognostic parameter with accelerated degradation using APFPM

Although the prognostic parameter is noisy and non-monotonic, the simulated APF trajectory is much smoother. The particles are sparse at the beginning of the cycle, indicated by the broad range of 95% confidence interval. They are more closely clustered later on. This means that the resampling algorithm selects a representative population of particles with high probability.

Relative accuracy (RA) is a metric to estimate the accuracy of RUL. It is a measure of the error in prediction relative to the actual value at a specific time, as shown in Eq. (30)

$$\mathrm{RA} = 1 - \frac{\left|R_p - R_t\right|}{R_t} \tag{30}$$

where R_p is the predicted RUL, and R_t is the true RUL. The prediction is more accurate when RA is close to *1*.

Precision Index (PI) is another metric that can be applied to estimate the precision of RUL, which measures the relative width of prediction interval. PI can be defined with

$$\mathrm{PI} = \frac{\left|R_{p,0.975} - R_{p,0.025}\right|}{R_p} \tag{31}$$

where $R_{p,0.975}$ and $R_{p,0.025}$ are the upper and lower bound of predicted RUL's 95% confidence interval. The prediction is more precious when PI is close to *0*.

In practice, it is not as important to acquire accurate RUL in early life as it is towards the end of life. α-λ performance (Saxena et al., 2010) is a binary metric that evaluates whether the accuracy at a specific time falls within a defined bound. By setting up a bound α equals to 20%, it can be determined if the RUL predictions fall within the bound at λ-fraction of the entire life. It requires the predictions to stay within a cone of accuracy, which is roughly loose at the beginning of life but more stringent as the system approaches the end of life.

A series of simulations were performed using different numbers of measurements to train the model. Fig. 5 shows the α-λ performance of the RUL estimation of Motor 2 as an example case. The horizontal axis "lifetime of learning" indicates the percentage of measurement data in the entire life cycle that was used for APF training. RA and PI of all five motors through their fractional lifetime can be found in Fig. 6 and Fig. 7, respectively.

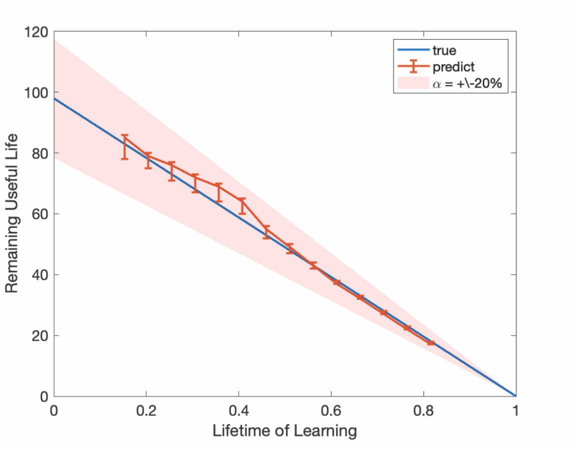

Figure 5. α-λ performance of RUL estimation of Motor 2 using APFPM

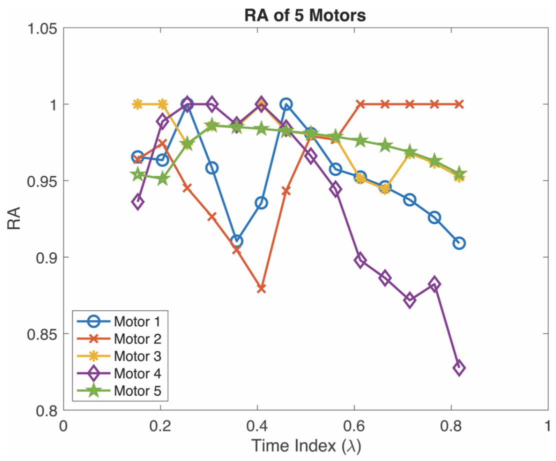

Figure 6. RA of all five motors

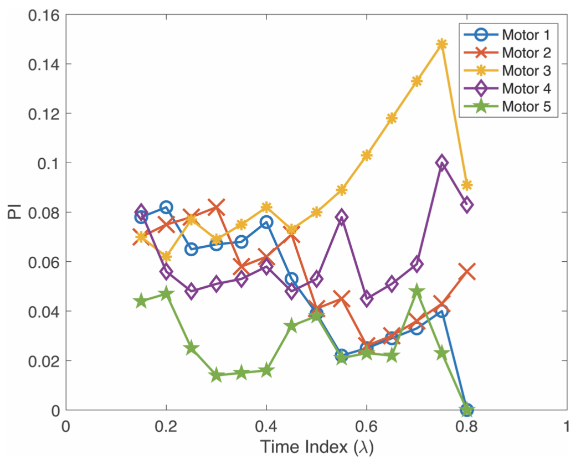

Figure 7. PI of all five motors

From Fig. 5-7, the model has an accurate prediction on Motor 2 at the end of life. When the prognostic parameter gets noisy in the middle of life, the RA drops slightly but comes back to around one as more data are observed to reinforce the degradation trend. Motor 2 has a similar PI performance; it slightly increases with noisy measurements but it decreases as more measurements are available. Motor 1 has a similar prognostic parameter trend with Motor 2. Both prognostic parameters have a significant amount of noise added to the measurement, which causes a drop in RA and roughly high PI in the mid-life cycle prediction.

The prognostic parameter for Motor 3 started to flat out in the middle of life cycle and showed a prolonged degradation until around cycle 90, followed by a dramatic decrease in value (Fig. 2). In the model training step, measurement after 0.80-time index, which is approximately 80 cycles, has not been used to train the APF. Thus, the model could not catch this change. Besides, this Bayesian model is not designed to handle a sudden and significant change in states, which caused the model to have a bigger span of particle distribution as indicated by high PI value.

APFPM is benchmarked with naïve PF-based prognostic model. In the benchmark, resampling is set to occur in each iteration, instead of waiting until the effective number of particles drop by a half. This approach ensures that both models resample at similar times. The naïve PF model is also initialized the same way as APFPM on Motor 2, without worrying about the auxiliary part and kernel smoothing technique. The naïve PF model utilized the multinomial resampling method, as it is the most commonly seen method in the literature. Both models were run fourteen times with different breakpoints to estimate RULs on a 2015 Macbook Pro with 16GB memories and 2.2GHz Quad-core Intel Core i7 processers. APFPM model finishes the calculation in 19.628 seconds, while the naïve PF model took 471.036 seconds to finish. Since 100,000 particles are used to train the model, APFPM is 24 times faster than naïve PF, which is close to their theoretical difference: 16 times, calculated by their time complexity difference $\frac{O(n \log n)}{O(n)}$.

Fig. 8 shows the α-λ performance of RUL estimation of Motor 2 using naïve PF model. By comparing it with Fig. 5, it is clear that the naïve PF model has a bigger particle distribution span than APFPM in each data point. Fig. 9 and Fig. 10 shows the particle and parameter distribution of naïve PF model at cycle 20, 50 and 70, respectively. In comparison with APFPM, Fig. 11 and Fig. 12 shows the particle and parameter distribution of APFPM at cycle 20, 50, and 70, respectively. As the particle and parameter distribution span is extremely narrow at later stage, separate plots of particle and parameter distribution of APFPM at cycle 70 is shown in Fig. 13 and Fig. 14, respectively. A comparison between prognostic parameter with simulated degradation of naïve PF model is shown in Fig. 15 using the first 70 measurements as training data.

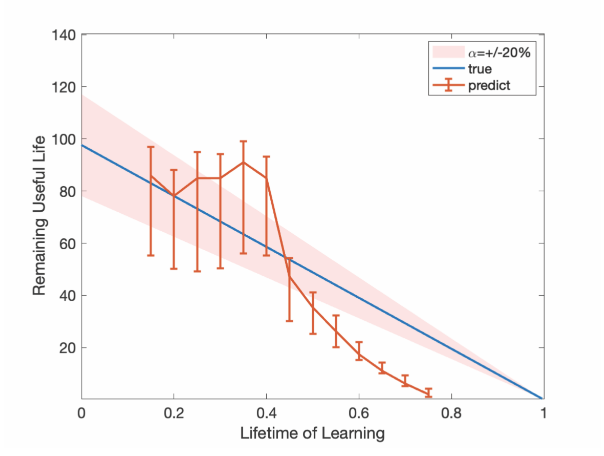

Figure 8. α-λ performance of RUL estimation of Motor 2 using PF

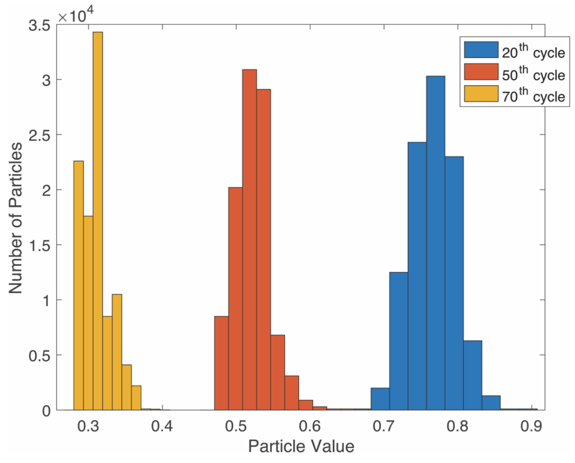

Figure 9. Particle distribution of naïve PF model at 20th, 50th, and 70th cycle

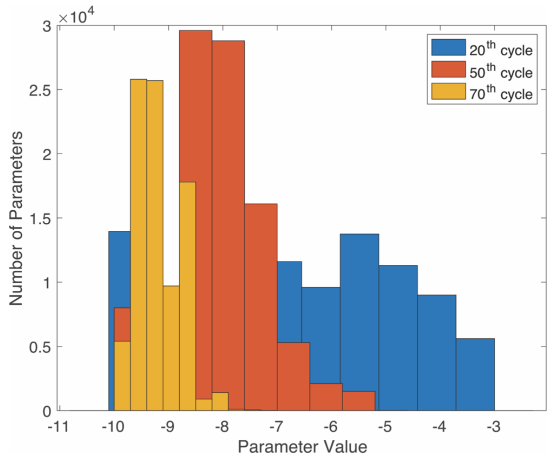

Figure 10. Parameter distribution of naïve PF model at 20th, 50th, and 70th cycle

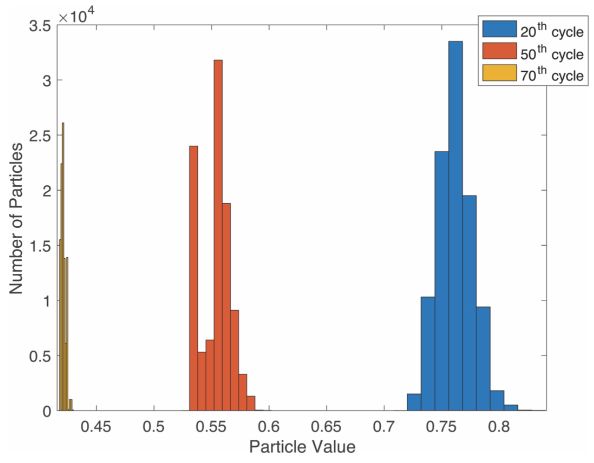

Figure 11. Particle distribution of APFPM model at 20th, 50th, and 70th cycle

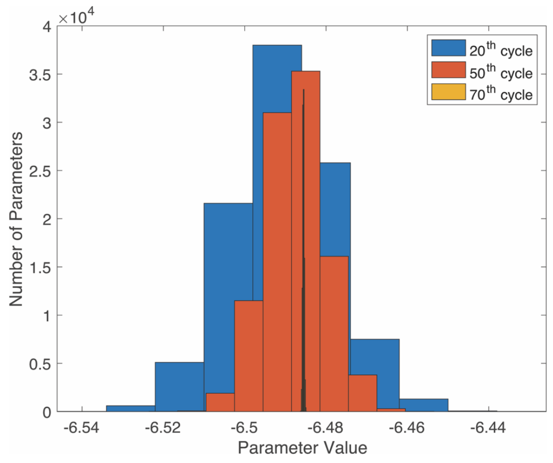

Figure 12. Parameter distribution of APFPM model at 20th, 50th, and 70th cycle

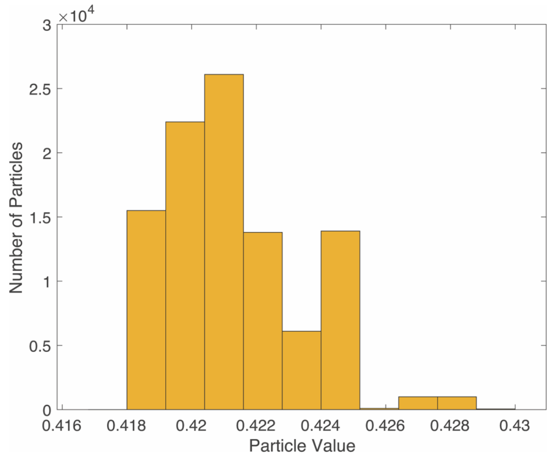

Figure 13. Particle distribution of APFPM at 70th cycle

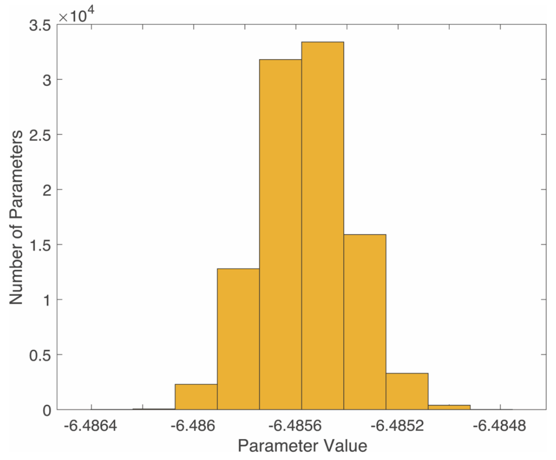

Figure 14. Parameter distribution of APFPM at 70th cycle

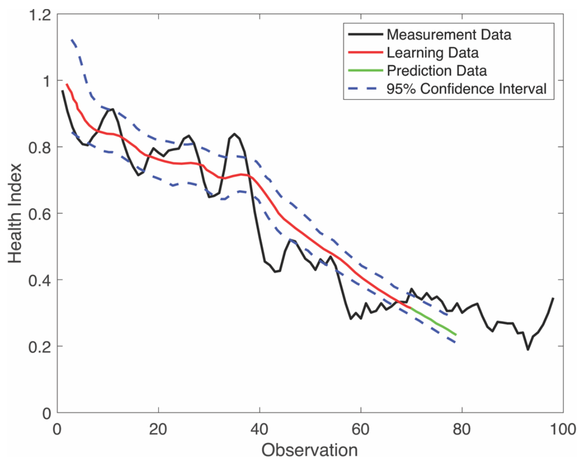

Figure 15. Motor 2 prognostic parameter with accelerated degradation using naïve PF

The particle distribution span decreases over time as more data are trained by both models. In naïve PF, the parameter is resampled when particles are resampled, based on $p(z_k|x_k^i)$. However, a high $p(z_k|x_k^i)$ does not guarantee a more accurate model parameter. Thus the selection of the model parameter is rather random, causing the distribution to be not ideally normal.

In comparison, the parameter distribution of APFPM is strictly normal and resampled using a kernel smoothing method to avoid deviation from the mean value. This algorithm guarantees the parameter and particle to be constrained in a tighter distribution. It is why the APFPM has a smaller 95% confidence interval of particles at each datapoint, seen by comparing the confidence intervals in Fig. 4 and Fig. 15.

The benchmark result also indicates that naïve PF model is more sensitive to a sudden change in the data, while APFPM is more robust to such changes. Following a rapid drop in the measurement data after the 35th cycle, the trend of naïve PF is dragged downward and predicts an early failure even with very good model initialization. This makes naïve PF model more sensitive to noise, and it is easy for naïve PF model to pick up measurement noise and cause overfitting. Conversely, APFPM requires a good initialization and it takes more iterations to correct itself if there is a change in the process. Thus APFPM is more suitable for situations that have noisier measurement data, smaller time steps, or the process is unlikely to change very often.

5. CONCLUSION

This paper introduced an auxiliary particle filter-based prognostic framework, APFPM. This APF model can estimate both unknown states and parameters at the same time using a large number of particles. This method is demonstrated using a dataset obtained from an accelerated motor degradation experiment.

Preprocessing of motor degradation data allows it to fuse raw experimental data into prognostic parameters. This paper introduced a data fusion method that combines IRT, PCA, and GA algorithms from a set of features extracted from the original data. It proves that the combination of these three algorithms generates prognostic parameter with highest score in monotonicity, prognosability, and trendability.

APFPM learns the degradation pattern using a prognostic parameter and further estimates the RUL by extrapolating the degradation curve until it crosses a predefined failure threshold. The results showed that this model could accurately predict the RUL.

The initialization step of APFPM requires particles and parameters to be set in a range. By learning the degradation curve, the particles and parameters can automatically adjust themselves to fit the curve using SIR. The kernel smoothing method is applied in sampling parameters to ensure the newly sampled parameters are close to their mean. It does not result in the increase of particle covariance.

Different resampling methods are compared based on the time complexity and the resampling performance of both particles with big weights and particles with reasonably big weights. The systematic resampling method is favored because of its low computational time complexity, which is extremely important in particle filter-based simulations. The systematic resampling method also has good performance on resampling particles with different weights.

A benchmarking between naïve PF-based prognostic model and APFPM is conducted and the use case of two models are discussed. Although APFPM requires an auxiliary step in each iteration, it is still significantly faster than naïve PF because systematic resampling method is deployed. The particles in APFPM are more concentrated in the distribution than naïve PF and the distribution of particle parameters is guaranteed to be normal. APFPM is not sensitive to noise in the prognostic parameter, which makes it more suitable for systems with more measurements and is unlikely to change much.

ACKNOWLEDGEMENT

This work was supported by a Nuclear Energy University Programs grant sponsored by the U.S. Department of Energy, Office of Nuclear Energy, award number DE-NE0008796.

REFERENCES

Alamaniotis, M., Grelle, A., & Tsoukalas, L. H. (2014). Regression to fuzziness method for estimation of remaining useful life in power plant components. Mechanical Systems and Signal Processing, 48(1–2), 188–198.

An, D., Choi, J. H., & Kim, N. H. (2012). *A comparison study of methods for parameter estimation in the physics-based prognostics*. 53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference 2012, 1–11.

Andrieu, C., Doucet, A., & Holenstein, R. (2010). *Particle Markov chain Monte Carlo methods. Journal of the Royal Statistical Society. Series B: Statistical Methodology*, 72(3), 269–342.

Andrieu, C., Doucet, A., & Punskaya, E. (2001). *Sequential Monte Carlo Methods for Optimal Filtering. In A. Doucet, N. de Freitas, & N. Gordon (Eds.), Sequential Monte Carlo Methods in Practice* (pp. 79–95). Springer New York

Bain, A., & Crisan, D. (2009). *Fundamentals of Stochastic Filtering*.

Baraldi, P., Mangili, F., & Zio, E. (2015). *A prognostics approach to nuclear component degradation modeling based on Gaussian Process Regression. Progress in Nuclear Energy*, 78, 141–154.

Barbieri, F., Hines, J. W., Sharp, M., & Venturini, M. (2015). *Sensor-based degradation prediction and prognostics for remaining useful life estimation: Validation on experimental data of electric motors*. International Journal of Prognostics and Health Management, 6(SP3), 1–20.

Cappé, Olivier. (2005). *Inference in hidden Markov models (Eric. Moulines & Tobias. Ryden, Eds.).* Springer.

Chen, T., Morris, J., & Martin, E. (2005). *Particle filters for state and parameter estimation in batch processes*. Journal of Process Control, 15(6), 665–673.

Coble, J. B. (2010). *Merging data sources to predict remaining useful life--an automated method to identify prognostic parameters*. Ph.D dissertation. The University of Tennessee.

Coble, J., & Wesley Hines, J. (2009). *Identifying optimal prognostic parameters from data: A genetic algorithms approach*. Annual Conference of the Prognostics and Health Management Society, PHM 2009, 1–11.

Daroogheh, N., Baniamerian, A., Meskin, N., Member, S., & Khorasani, K. (2017). *Prognosis and Health Monitoring of Nonlinear Systems Using a Hybrid Scheme Through Integration of PFs and Neural Networks*. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 47(8), 1990–2004.

Djeddi, M., Granjon, P., & Leprettre, B. (2007). *Bearing fault diagnosis in induction machine based on current analysis using high-resolution technique*. 2007 IEEE International Symposium on Diagnostics for Electric Machines, Power Electronics and Drives, SDEMPED, 23–28.

Dong, H., Jin, X., Lou, Y., & Wang, C. (2014*). Lithium-ion battery state of health monitoring and remaining useful life prediction based on support vector regression-particle filter*. Journal of Power Sources, 271, 114–123.

Doucet, A., Godsill, S., & Andrieu, C. (2000). *On sequential Monte Carlo sampling methods for Bayesian filtering*. Statistics and Computing, 197–208.

Elvira, V., Miguez, J., & Djurie, P. M. (2017). *Adapting the Number of Particles in Sequential Monte Carlo Methods Through an Online Scheme for Convergence Assessment*. IEEE Transactions on Signal Processing, 65(7), 1781–1794.

Gnanaprakasam, C. N., & Chitra, K. (2014). *Prognostic of electrical motor vibration signals: A hybrid technique*. Journal of Theoretical and Applied Information Technology, 63(2), 543–559.

Gordon, N. J., Salmond, D. J., & Smith, A. F. M. (1993). *Novel approach to nonlinear/non-gaussian Bayesian state estimation*. IEEE Proceedings, Part F: Radar and Signal Processing, 140(2), 107–113.

Hol, J. D., Schön, T., & Gustafsson, F. (2006). *On Resampling Algorithms for Particle Filters*. 2006 IEEE Nonlinear Statistical Signal Processing Workshop, 79–82.

Hu, Y., Baraldi, P., Di Maio, F., & Zio, E. (2015). *A particle filtering and kernel smoothing-based approach for new design component prognostics*. Reliability Engineering and System Safety, 134, 19–31.

Huynh, K. T., Castro, I. T., Barros, A., & Bérenguer, C. (2012). *Modeling age-based maintenance strategies with minimal repairs for systems subject to competing failure modes due to degradation and shocks*. European Journal of Operational Research, 218(1), 140–151.

IEEE Standard Test Procedure for Evaluation of Systems of Insulating Materials for Random-Wound AC Electric Machinery. (1974). In ANSI C50.32-1976 and IEEE Std 117-1974 (Reaffirmed 1984) (Revision of IEEE Std 117-1956) (pp. 1–24).

Jardine, A. K. S., Lin, D., & Banjevic, D. (2006). *A review on machinery diagnostics and prognostics implementing condition-based maintenance*. Mechanical Systems and Signal Processing, 20(7), 1483–1510.

Jouin, M., Gouriveau, R., Hissel, D., Péra, M. C., & Zerhouni, N. (2016). *Particle filter-based prognostics: Review, discussion and perspectives*. Mechanical Systems and Signal Processing, 72–73, 2–31.

Kim, G., Kim, H., Zio, E., & Heo, G. (2018). *Application of particle filtering for prognostics with measurement uncertainty in nuclear power plants*. Nuclear Engineering and Technology, 50(8), 1314–1323.

Kim, H., Robert, C. P., & Casella, G. (2000). *Monte Carlo Statistical Methods*. In Technometrics (Vol. 42, Issue 4).

Kitagawa, G. (1996). *Monte Carlo Filter and Smoother for Non-Gaussian Nonlinear State Space Models*. Journal of Computational and Graphical Statistics, 5(1), 1–25.

Kong, A., & Liu, J. S. (1994). *Sequential imputations and Bayesian missing data problems*. Journal of the American Statistical Association, 89(425), 278–288.

Li, X., Ding, Q., & Sun, J. Q. (2018). *Remaining useful life estimation in prognostics using deep convolution neural networks*. Reliability Engineering and System Safety, 172(December 2017), 1–11.

Liu, J. S., & Chen, R. (1998). *Sequential monte carlo methods for dynamic systems. Journal of the American Statistical Association*, 93(443), 1032–1044.

Liu, J., & West, M. (2001). *Combined Parameter and State Estimation in Simulation-Based Filtering. Sequential Monte Carlo Methods in Practice*, 197–223.

Orchard, M., & Vachtsevanos, G. J. (2009). *A particle-filtering approach for on-line fault diagnosis and failure prognosis*. 4, 221–246.

Orchard, M., Kacprzynski, G., Goebel, K., Saha, B., & Vachtsevanos, G. (2008). *Advances in uncertainty representation and management for particle filtering applied to prognostics.* 2008 International Conference on Prognostics and Health Management, 1–6.

Orchard, M., Tang, L., Saha, B., Goebel, K., & Vachtsevanos, G. (2010). *Risk-Sensitive Particle-Filtering-based Prognosis Framework for Estimation of Remaining Useful Life in Energy Storage Devices*. Studies in Informatics and Control, 19, 209–218.

Park, J. I., & Bae, S. J. (2010). *Direct Prediction Methods on Lifetime Distribution of Organic Light-Emitting Diodes From Accelerated Degradation Tests*. IEEE Transactions on Reliability, 59(1), 74–90.

Pecht, M. (1991). *Prognostics and Health Management of Electronics*. In New York: John Wiley (pp. 1–13).

Pitt, M. K., & Shephard, N. (1999). *Filtering via Simulation: Auxiliary Particle Filters*. Journal of the American Statistical Association, 94(446), 590–599.

Rabiei, E., Droguett, E. L., & Modarres, M. (2018). *Fully adaptive particle filtering algorithm for damage diagnosis and prognosis*. Entropy, 20(2).

Savitzky, Abraham., & Golay, M. J. E. (1964). *Smoothing and Differentiation of Data by Simplified Least Squares Procedures*. Analytical Chemistry, 36(8), 1627–1639.

Saxena, A., Celaya, J., Saha, B., Saha, S., & Goebel, K. (2010). *Metrics for offline evaluation of prognostic performance*. International Journal of Prognostics and Health Management, 1(1).

Sharp, M. E. (2012). *Prognostic Approaches Using Transient Monitoring Methods*. Ph.D Dissertation, The University of Tennessee.

Si, X. S., Wang, W., Hu, C. H., & Zhou, D. H. (2011). *Remaining useful life estimation - A review on the statistical data driven approaches*. European Journal of Operational Research, 213(1), 1–14.

Storvik, G. (2002). *Particle filters for state-space models with the presence of unknown static parameters*. IEEE Transactions on Signal Processing, 50(2), 281–289.

Tseng, S. T., & Peng, C. Y. (2004). *Optimal burn-in policy by using an integrated Wiener process*. IIE Transactions (Institute of Industrial Engineers), 36(12), 1161–1170.

Upadhyaya, B. R., Erbay, A. S., & McClanahan, J. P. (1997). *Accelerated Aging Studies of Induction Motors and Fault Diagnostics*. Maintenance and Reliability Center, The University of Tennessee, Knoxville.

Wang, X., & Xu, D. (2010). *An inverse gaussian process model for degradation data*. Technometrics, 52(2), 188–197.

Wicker, N., Muller, J., Kalathur, R. K. R., & Poch, O. (2008). *A maximum likelihood approximation method for Dirichlet's parameter estimation*. Computational Statistics & Data Analysis, 52(3), 1315–1322.

Yu, J. (2017). *Aircraft engine health prognostics based on logistic regression with penalization regularization and state-space-based degradation framework*. Aerospace Science and Technology, 68, 345–361.

Zhang, Q., Tse, P. W. T., Wan, X., & Xu, G. (2015). *Remaining useful life estimation for mechanical systems based on similarity of phase space trajectory*. Expert Systems with Applications, 42(5), 2353–2360.

Zhang, Z., Si, X., Hu, C., & Lei, Y. (2018). *Degradation data analysis and remaining useful life estimation: A review* *on Wiener-process-based methods*. European Journal of Operational Research, 271(3), 775–796.

Zhao, Z., Huang, B., & Liu, F. (2013). *Parameter estimation in batch process using EM algorithm with particle filter*. Computers & Chemical Engineering, 57, 159–172.

Zio, E., & Peloni, G. (2011). *Particle filtering prognostic estimation of the remaining useful life of nonlinear components*. Reliability Engineering and System Safety, 96(3), 403–409.