A Data-Driven Particle Filter Approach for System-Level Prediction of Remaining Useful Life

Abel Diaz-Gonzalez
Institute for Software Integrated Systems, Vanderbilt University, Nashville, TN, USA

Austin Coursey
Institute for Software Integrated Systems, Vanderbilt University, Nashville, TN, USA

Marcos Quinones-Grueiro
Institute for Software Integrated Systems, Vanderbilt University, Nashville, TN, USA

Gautam Biswas
Institute for Software Integrated Systems, Vanderbilt University, Nashville, TN, USA

Abstract

Accurate estimation of the remaining useful life (RUL) of industrial systems is a critical component of predictive maintenance strategies. This work presents a data-driven method for RUL prediction that also quantifies uncertainty, drawing inspiration from model-based particle filtering techniques. Instead of simulating system state transitions, we model degradation as a stochastic process governed by performance metrics and use a Bayesian particle filtering framework to infer its underlying parameters. Our approach bypasses traditional state-space modeling by directly estimating the end-of-life distribution from observed performance data. Key characteristics of the filter, such as propagation noise and observation correction strength, are adapted over time based on current observations and past predictive performance, enabling better capture of future uncertainty. We evaluate the proposed method using an unmanned aerial vehicle simulation dataset developed for system-level prognostics research, which includes high-fidelity degradation signals and ground-truth system performance metrics for validating predictive accuracy.

2012 ACM Subject Classification Applied computing → Aerospace; Computing methodologies → Artificial intelligence; Computing methodologies → Machine learning approaches

Keywords and phrases remaining useful life, particle filter methods, data-driven methods, system-level prognostics, performance metrics

Digital Object Identifier 10.4230/OASIcs.DX.2025.11

Funding *Abel Diaz-Gonzalez*: NASA University Leadership Initiative (ULI)

1 Introduction

Industrial systems inevitably degrade over time, posing a risk of failure and unexpected downtime. Accurate estimation of a system's remaining useful life (RUL) is essential for implementing predictive maintenance strategies that improve operational efficiency, reduce unplanned outages, and lower life-cycle costs [12].

Prognostic techniques for RUL estimation are typically classified into model-based, data-driven, and hybrid approaches [9]. Model-based techniques leverage physical principles to construct degradation models and can yield interpretable and reliable predictions when the underlying models are accurate and well-calibrated. However, they require extensive domain knowledge and are often impractical for systems with complex or poorly understood dynamics [6]. In contrast, data-driven methods, including deep neural networks, learn degradation patterns directly from sensor data and are well-suited for modern applications where large volumes of operational data are available [10, 5]. Despite their success, these methods often lack interpretability and provide limited tools for quantifying predictive uncertainty, which is critical for risk-aware decision-making. Hybrid approaches combine

elements of both paradigms to exploit the complementary advantages of physics-based modeling and data-driven learning. By incorporating domain knowledge into data-driven frameworks, hybrid methods can improve prediction accuracy, robustness, and extrapolation capability. However, they also inherit the limitations of both families of techniques, such as the modeling cost of physics-based approaches and the data requirements of data-driven methods, which makes their implementation challenging in practice.

In this paper, we introduce a novel prognostic framework that adapts the particle filter (PF) methodology to a data-driven setting. In traditional model-based prognostics, PFs are used to estimate the hidden state of a system over time by recursively applying a known or learned state transition model. However, in data-driven applications, these models must be inferred from data, and even small errors in the learned dynamics can compound over time, degrading predictive accuracy. To avoid this issue, our method does not rely on recursive state estimation. Instead, it directly models the distribution of the performance metrics that define the system's end of life (EOL). We assume that these performance values are available both online and in historical run-to-failure datasets. How they are obtained from raw system observations, whether through direct measurement or a separate modeling stage, is outside the scope of this work. Given these inputs, we represent degradation as a stochastic process indexed by performance level rather than time, allowing for flexible and interpretable modeling of failure progression. To demonstrate the method, we use an extension of the Gamma process proposed by [11], which models the distribution of time required to reach a given performance value.

The parameters of this process, which govern the system's degradation behavior, are treated as latent variables within a Bayesian framework and are inferred from the observed performance values using particle filtering. Our method builds on the probabilistic foundation of particle filtering but repurposes it to estimate the EOL distribution directly from past and current observations. This formulation captures both stochastic uncertainty, arising from inherent randomness in the degradation process, and epistemic uncertainty, due to limited knowledge about future evolution. As the system progresses, particle weights are updated through a modified correction step that remains faithful to the particle filtering framework but accounts for future uncertainty by adjusting the "strength" of the correction. Observations made early in the degradation timeline, when future outcomes are still highly uncertain, have a weaker influence on particle weights, while those made closer to failure carry more weight. Both the propagation noise, used to sample hypothetical values of the latent process parameters, and the correction strength are learned from historical run-to-failure data, enabling a data-driven and theoretically grounded approach to modeling uncertainty across the degradation timeline.

The method can be applied in parallel to multiple performance metrics that reflect different aspects of system degradation. For each metric, we estimate a dedicated EOL distribution based on a predefined threshold. The final RUL prediction for the system is then obtained using standard aggregation rules. In this work, we select the earliest predicted failure time across all metrics.

The main contributions of this work are as follows:

- A data-driven prognostic framework inspired by model-based filtering, which directly estimates RUL and its uncertainty from performance metrics without relying on system models or expert knowledge.
- A learning-based particle filtering scheme that infers degradation behavior and adapts uncertainty handling from historical run-to-failure data, offering improved interpretability over black-box methods.

We evaluate our approach using a custom dataset generated in Simulink, which provides ground-truth system-level performance metrics under run-to-failure conditions. By modeling degradation as a stochastic process indexed by performance level and treating its governing parameters as latent variables, our method uses particle filtering not for state tracking, but for sequential inference of these degradation parameters. The resulting EOL predictions are expressed as probabilistic mixtures over learned process trajectories, providing both interpretable and uncertainty-aware RUL estimates. This formulation offers a principled alternative to black-box models, with improved transparency and statistical grounding.

2 EOL Particle Filter

Performance metrics used to define the EOL process typically exhibit a monotonic trend [13]; performance degrades over time, and does not recover except following maintenance. We assume this monotonicity in our analysis and treat noise-induced fluctuations as minor deviations. We then linearly scale these monotonic metrics so that the performance level s ranges from 1 (fully healthy) to 0 (the EOL threshold).

Consider the stochastic process $\{T_s\}_{s\in[0,1]}$ that models the time at which the scaled performance metric reaches the level s. Notice that the distribution of the time to performance metric EOL is given by the random variable T_0. To model unit-specific degradation behaviour, we introduce latent degradation parameters X that govern the evolution of the stochastic process T_s. The parameters X capture intrinsic degradation characteristics that remain fixed for each unit over its lifetime. Different units may follow distinct degradation trajectories, and this unit-to-unit variability is encoded through variation across their corresponding latent parameters X. In practice, the true values of the parameters X are unknown and must be inferred from data. Once the true parameter vector is identified, i.e., $X = x^*$, the process $\{T_s(x^*)\}_{s\in[0,1]}$ defines a distribution over the times at which each performance level s is reached. The randomness inherent in $T_s(x^*)$ captures stochastic uncertainty in the degradation process.

Accurately estimating x^* typically requires a dense sequence of observations spanning the full performance range. However, such data is only available at the end of a unit's life, when predictions are no longer needed. For online predictions prior to failure, it is crucial to account for the uncertainty in the estimation of x^* due to missing future observations. To address this, we adopt a Bayesian framework in which the latent degradation parameters X are treated as latent random variables with a prior distribution that is updated over time as new observations are collected. At each time step k, we maintain a posterior approximation denoted X_k, allowing us to infer the likely values of x^* as additional measurements y_i become available. The stochasticity of X_k captures epistemic uncertainty arising from incomplete information. We employ particle filtering to perform this sequential Bayesian inference in real time as the system evolves.

2.1 Particle Filters

PFs are a class of Bayesian filters that use sequential Monte Carlo sampling to approximate the posterior distribution of latent variables with a set of weighted samples or "particles" [7, 14]. While classical filters such as the Kalman Filter are optimal under linear and Gaussian assumptions, PFs offer a more flexible and nonparametric alternative that performs well in nonlinear and non-Gaussian settings. In our work, we adapt the particle filtering framework to sequentially infer the latent degradation parameter X as more observations become available. This allows us to capture both epistemic and stochastic uncertainty in EOL predictions. Below we present the details of this degradation model and the inference procedure.

Bayesian filtering provides a principled framework for sequentially updating our knowledge about a sequence of latent variables as new indirect observations of these variables arrive. At time step k, the goal is to estimate the posterior distribution of the latent variable X_k given all observations $y_{0:k}$ up to current time k. This posterior is often referred to as the belief distribution and is computed in two recursive steps:

$$\begin{aligned} bel_k(x_k) &:= p(X_k = x_k \mid Y_{0:k-1} = y_{0:k-1}) \\ &= \int p(x_k \mid x_{k-1})\, \overline{bel}_{k-1}(x_{k-1})\, dx_{k-1} \quad \text{(prediction)} \end{aligned} \tag{1}$$

$$\overline{bel}_k(x_k) := p(X_k = x_k \mid Y_{0:k} = y_{0:k}) \propto p(y_k \mid x_k)\, bel_k(x_k) \quad \text{(correction)} \tag{2}$$

The belief distribution $\overline{bel}_k$ thus captures the current probabilistic estimate of X_k. Notice that the particle filtering framework depends only on the transition, sensor models:

$$\begin{aligned} p(x_k \mid x_{k-1}) &:= p(X_k = x_k \mid X_{k-1} = x_{k-1}) \quad \text{(transition model)}, \\ p(y_k \mid x_k) &:= p(Y_k = y_k \mid X_k = x_k) \quad \text{(sensor model)}. \end{aligned}$$

and the initial distribution of the latent variable $bel_0(x) = p(X_0 = x)$.

In our setting, the latent process parameters are static, so ideally we have $X_k = X$ for all $k \geq 1$. However, similar to standard particle filtering, we adopt a pseudo-transition model with Gaussian propagation noise to maintain particle diversity and mitigate degeneracy.

Each observation $y_k = (t_k, s_k)$ provides an indirect measurement of the underlying degradation parameters X through the stochastic process T_s. This leads to the following sensor and transition models:

$$\begin{aligned} p(y_k \mid x_k) &= P(T_{s_k} = t_k \mid X_k = x_k), \\ p(x_k \mid x_{k-1}) &= \mathcal{N}(x_{k-1}, \Sigma), \end{aligned} \tag{3}$$

where the transition model injects artificial Gaussian noise centered at the previous state. As usual in particle filtering, for simplification, we assume the covariance matrix Σ to be diagonal, which corresponds to injecting independent noise into each state component. Its diagonal entries control the strength of the perturbation, allowing the filter to explore broader regions of the parameter space when uncertainty is high, and to concentrate the posterior distribution as more informative data become available.

The algorithm then follows the standard bootstrap PF framework [7, 8], with a modified correction step tailored for parameter inference. This modification arises from the fact that, unlike in standard particle filtering, not all observations provide the same level of insight into the degradation parameters. Early in the system's life, observations tend to be less informative, as much of the uncertainty lies in the unobserved future. Consequently, particles that appear unlikely based on current data may still correspond to plausible future scenarios and should not be prematurely discarded. Conversely, near the EOL, observations become more informative and should exert greater influence on particle selection.

To account for this, we introduce a tempering parameter $\kappa > 0$, referred to as the observation correction strength, which modulates the influence of the sensor model. We temper the likelihood function by raising it to the power κ, yielding tempered importance weights:

$$\tilde{w}_k^{(i)} = P(T_{s_k} = t_k \mid X_k = x_k^{(i)})^{\kappa}. \tag{4}$$

This approach follows the power posterior formulation in Bayesian inference, where the likelihood is scaled to adjust its impact on the posterior. When $\kappa < 1$, the update is softened to preserve particle diversity under high uncertainty. When $\kappa > 1$, the correction becomes sharper and more selective. The standard PF corresponds to the special case $\kappa = 1$.

In our framework, both the observation correction strength κ and the variances in the diagonal matrix Σ are learned offline from historical data as a function of the observation. Offline data is also used to construct the initial distribution $p(X_0)$. Specifically, we obtain a parameter estimate $x_0^{(i)}$ for each training unit by maximizing the likelihood of its run-to-failure data, for $i = 1, 2, \ldots, n$. These estimates are then augmented with Gaussian noise to generate the desired number of particles N, which approximates the initial distribution $p(X_0)$ as:

$$p(X_0 = x) \approx \frac{1}{N} \sum_{i=1}^{N} \delta_{x_0^{(i)}}(x), \tag{5}$$

where δ_z is the Dirac measure centered at z. This corresponds to a standard PF initialization, where all particles are equally weighted. The variance of the Gaussian noise used for augmentation is treated as a hyperparameter of the algorithm.

Then the RUL prediction PF we propose can be written as:

1. *Initialization*, $k = 0$.
   - For $i = 1, \ldots, N$, we consider $x_0^{(i)}$ and set $k = 1$.
2. *Prediction step*
   - For $i = 1, \ldots, N$, sample $\tilde{x}_k^{(i)} \sim \mathcal{N}(x_{k-1}^{(i)}, \Sigma)$
3. Correction step
   - For $i = 1, \ldots, N$, evaluate the importance weights:

     $$\tilde{w}_k^{(i)} = p(y_k \mid \tilde{x}_k^{(i)})^{\kappa}.$$

   - Normalise the importance weights.

     $$w_k^{(i)} = \frac{\tilde{w}_k^{(i)}}{\sum_{j=1}^{N} \tilde{w}_k^{(j)}}.$$

4. *Selection step*
   - Resample with replacement N particles $\{x_k^{(i)}\}_{i=1}^{N}$ from the set $\{\tilde{x}_k^{(i)}\}_{i=1}^{N}$ according to the importance weights $\{w_k^{(i)}\}_{i=1}^{N}$.
5. Set $k \leftarrow k + 1$ and go to step 2.

2.2 Predictive distribution

At each time step k, particle filtering maintains an empirical approximation $\overline{bel}_k(x)$ of the posterior distribution over the latent degradation parameters. This distribution represents the current belief about the true degradation parameter X. In practice, PFs approximate this belief using a discrete posterior distribution composed of Dirac delta functions centered at the particles:

$$\overline{bel}_k(x) \approx \sum_{i=1}^{N} w_k^{(i)} \, \delta_{x_k^{(i)}}(x),$$

where $w_k^{(i)}$ is the importance weights associated with the particle $x_k^{(i)}$.

We use this empirical belief distribution to compute the predictive probability distribution of T_s, marginalizing over the posterior distribution of X_k:

$$\begin{aligned} P(T_s = t \mid y_{0:k}) &= \int P(t \mid x_k) P(x_k \mid y_{0:k})\, dx_k \\ &= \int P(t \mid x_k)\, \overline{bel}_k(x_k)\, dx_k \\ &\approx \sum_{i=1}^{N} w_k^{(i)}\, P(T_s(X_k) = t \mid X_k = x_k^{(i)}) \\ &= \sum_{i=1}^{N} w_k^{(i)}\, P(T_s(x_k^{(i)}) = t). \end{aligned} \tag{6}$$

The result is a weighted mixture distribution, where each component corresponds to the performance metric stochastic process where the degradation parameter is the particle, that is, $T_s(x_k^{(i)})$ for each particle $x_k^{(i)}$. This can be interpreted as:

- Each particle $x_k^{(i)}$ represents a distinct hypothesis about the unit's degradation behavior.
- Its weight $w_k^{(i)}$ reflects the current confidence in that hypothesis.

2.3 EOL prediction

In particular, the EOL predictive distribution at time step k can be approximated by

$$P(T_0 = t \mid y_{0:k}) \approx \sum_{i=1}^{N} w_k^{(i)}\, P(T_0(x_k^{(i)}) = t). \tag{7}$$

From the distribution of the stochastic process $\{T_s\}_{s\in[0,1]}$, we compute the EOL prediction at time k as the mean of the EOL predictive distribution:

$$\widehat{\mathrm{EOL}}_k := \mathbb{E}[T_0 \mid y_{0:k}] = \sum_{i=1}^{N} w_k^{(i)}\, \mathbb{E}[T_0(x_k^{(i)})]. \tag{8}$$

In general, the distribution $P(T_0 = t \mid y_{0:k})$ does not have a closed-form solution, but we can use any numerical method to approximate confidence intervals. To compute a confidence interval at level ℓ, we find the quantiles a and b such that

$$\begin{aligned} P(T_0 < a) &= \frac{1-\ell}{2}, \\ P(T_0 < b) &= \frac{1+\ell}{2}, \end{aligned}$$

so that $P(a < T_0 < b) = \ell$.

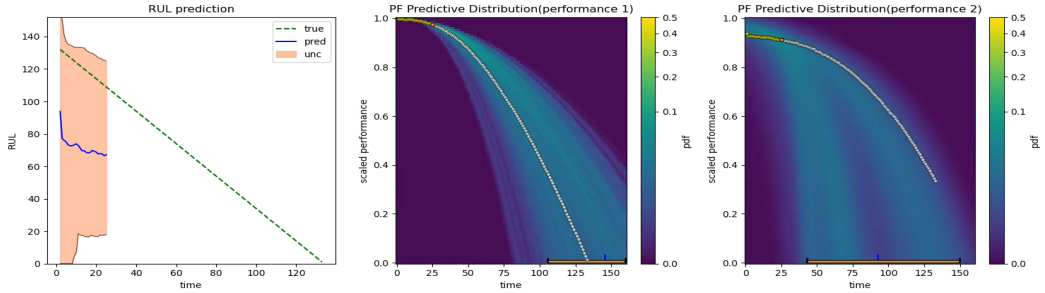
Finally, the EOL prediction and confidence interval at time step k of the system are computed by applying any aggregation of the corresponding performance predictions. In this work, we use the earliest among the estimated quantities (i.e., mean and quantiles) across all performance metrics as the final prediction and uncertainty bounds.

Figure 1 provides a visual overview of the proposed RUL prediction method at three different time points. Each row corresponds to a distinct stage in the system's lifecycle.

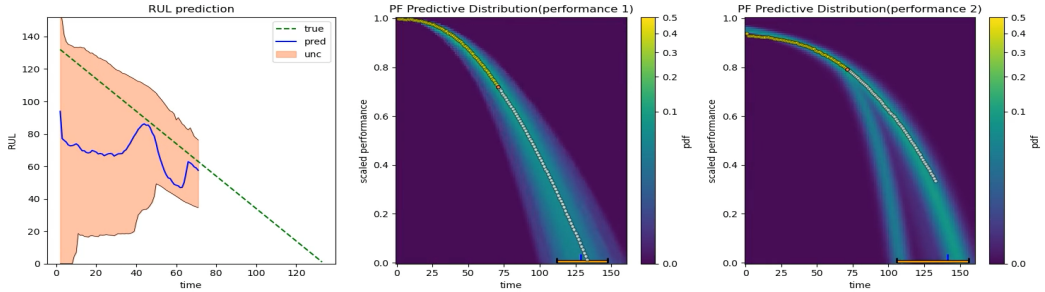
The left panel in each row displays the predicted RUL of the system, along with a 95% confidence interval computed from the empirical particle distribution. As the system approaches the end of its operational life, the uncertainty decreases and the predictions

become more confident. The green dashed line indicates the ground-truth RUL, while the solid blue line and shaded salmon region represent the predicted RUL and its confidence interval, respectively.

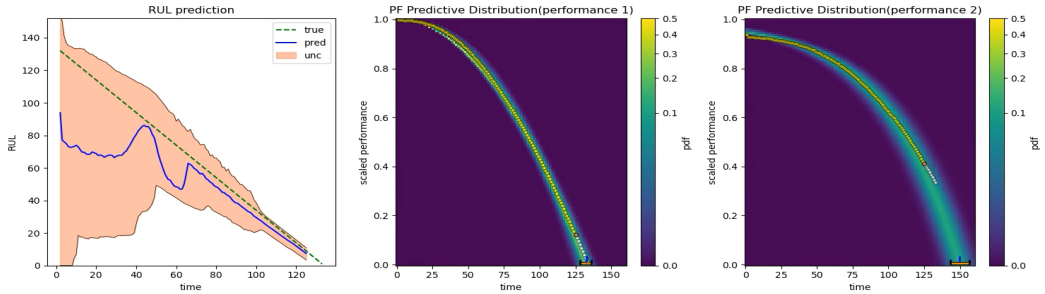
The center and right panels depict the predictive probability densities of the stochastic process $\{T_s\}_{s\in[0,1]}$, each corresponding to a different performance metric, and are based on their respective current PF estimate of the latent degradation parameters X. Yellow, orange, and white points indicate past, current, and future observations, respectively. The horizontal orange bars at the bottom mark the 95% confidence interval of the predicted end-of-life distributions at the current time, and the vertical blue line denotes the predicted RUL. Both are aggregated to produce the RUL prediction and confidence interval shown in the left panel. A brighter point (t, s) in the plot indicates a higher likelihood of reaching performance level s at time t. At the second time point, for example, a bimodal predictive distribution emerges, suggesting multiple plausible degradation trajectories consistent with the current particle estimates.



(a) Visualization at an early time step.



(b) Visualization at a mid-lifecycle time step.



(c) Visualization near EOL.

Figure 1 Evolution of RUL prediction and predictive probability densities over three stages of a system's life. Each row corresponds to a different time step during the run-to-failure simulation.

3 Performance Model

Following [11], we use the commonly encountered Gamma family of end-of-life distributions to illustrate our method and present the results. While that work also considers the Weibull distribution, we focus on the Gamma distribution for simplicity. Another key difference is that their model is applied to the health index that aggregates all performance metrics and units distributions, whereas we use it to capture the stochasticity of individual unit performance values separately.

We extend the parametric model presented in equation (11) of [11] to increase its flexibility. In the original formulation, all performance trajectories begin at the fully healthy state $s = 1$, and the initial variance of the process (at $t = 0$) is zero. To relax these constraints, we introduce two additional parameters: α, which decouples the initial performance value from $s = 1$, and μ, which allows the model to capture non-zero variance at early stages.

We define the performance degradation model as:

$$\mathbf{s}(t) = \alpha\left(1 - [b(t-\mu)]^{\rho}\right), \quad b \sim \text{InvGamma}(\beta, \lambda), \tag{9}$$

where $\mathbf{s}(t)$ represents the performance level at time t, and b is a random variable following an inverse Gamma distribution.

The parameters of the model are summarized as:

- $\alpha > 0$: controls the initial performance level,
- $\rho > 0$: controls the curvature or steepness of the degradation curve,
- $\mu \leq 0$: shifts the gamma function from zero to a negative value allowing the degradation to not start with zero variance,
- $\beta, \lambda > 0$: shape and rate (inverse scale) parameters of the inverse Gamma distribution.

From this performance model, and following a procedure similar to [11], the distribution of the time T_s to reach a given performance level s can be derived as follows:

$$\begin{aligned} P(T_s > t) &= P(\mathbf{s}(t) > s) = P\left(\alpha\left(1 - [b(t-\mu)]^{\rho}\right) > s\right) \\ &= P\left([b(t-\mu)]^{\rho} < 1 - \frac{s}{\alpha}\right) = P\left(b < \frac{(1-\frac{s}{\alpha})^{1/\rho}}{t-\mu}\right) \\ &= P\left(\frac{1}{b} > \frac{t-\mu}{(1-\frac{s}{\alpha})^{1/\rho}}\right). \end{aligned}$$

Since $b \sim \text{InvGamma}(\beta, \lambda)$, we have $\frac{1}{b} \sim \text{Gamma}(\beta, \lambda)$. Therefore,

$$P(T_s < t) = P\left(\frac{1}{b} < \frac{t-\mu}{(1-\frac{s}{\alpha})^{1/\rho}}\right) = \frac{1}{\Gamma(\beta)}\gamma\left(\beta, \lambda\frac{t-\mu}{(1-\frac{s}{\alpha})^{1/\rho}}\right),$$

where γ denotes the lower incomplete Gamma function. Hence, the stochastic process T_s representing the time to reach performance level s follows the shifted Gamma distribution:

$$T_s \sim \text{Gamma}\left(\beta, \lambda_s, \mu\right), \quad \text{where} \quad \lambda_s = \frac{\lambda}{(1-\frac{s}{\alpha})^{1/\rho}}.$$

In particular, the EOL distribution corresponds to $s = 0$, is given by:

$$T_0 \sim \text{Gamma}\left(\beta, \lambda, \mu\right).$$

Finally, we collect all the parameters of the degradation model into a single vector:

$$X = (\beta, \lambda, \rho, \mu, \alpha),$$

which defines the latent degradation parameters of the stochastic process. The PF is then used to infer the value of X that best explains the observed data.

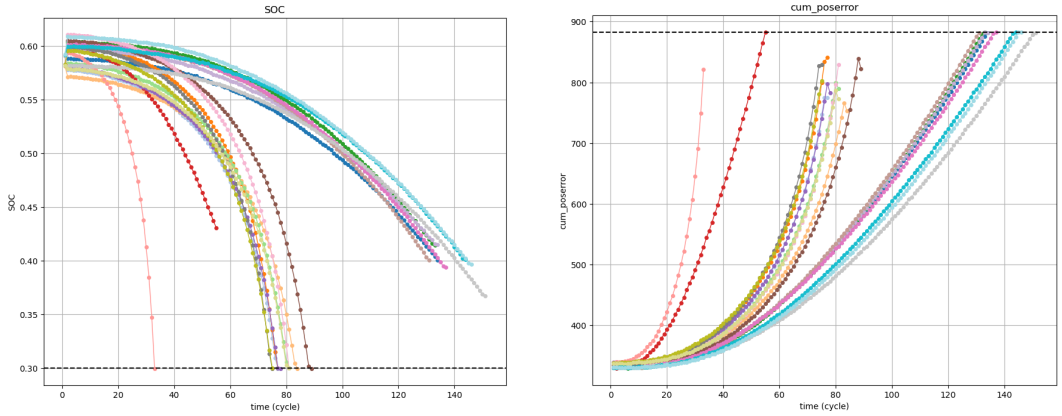
3.1 Dataset

To generate realistic degradation data for evaluating our proposed method, we used a MATLAB implementation of the octo-rotor simulation model introduced in [1]. This high-fidelity framework emulates a fault-tolerant unmanned aerial vehicle (UAV) system by modeling detailed dynamics, control mechanisms, and component degradation under realistic flight conditions. The simulator incorporates empirically validated degradation behaviors for key components such as motor internal resistance, battery capacity, and internal resistance, capturing both component aging and environmental interactions.

Using this platform and the original degradation profiles, we created a custom dataset aligned with our experimental goals. Each UAV is modeled with nine degrading components: eight motors (each with degrading internal resistance) and one battery (modeled with degrading charge capacity and internal resistance). We simulated run-to-failure trajectories for 20 UAVs, each performing repeated missions along a square 1km flight path at a fixed altitude of 50meters. The missions include full take-off and landing cycles and are conducted under a constant 10 m/s wind disturbance in the negative x-direction. This simulation environment provides a controlled yet complex test bed for validating prognostic methods under realistic operational and degradation conditions.

Two performance metrics were monitored throughout each UAV's run-to-failure simulation: a component-level metric, the battery state of charge (SOC), and a system-level metric, the cumulative position error over the flight trajectory. The position error was computed as the accumulated distance between the UAV's measured position and its projection onto the intended straight-line path between consecutive waypoints. After each flight, these metrics were recorded, and the simulation continued until either metric exceeded a predefined failure threshold. Specifically, the failure threshold for SOC was set to 0.3, while the threshold for cumulative position error was defined as the equivalent of 1 meter per second of expected flight time.

Figure 2 illustrates the evolution of both SOC and cumulative position error across the 20 simulated UAVs, with dashed lines indicating the failure thresholds for each metric.



(a) Battery SOC. The dashed black line marks the failure threshold at 0.3.

(b) Cumulative position error. The dashed black line indicates the failure threshold of 884 meters.

Figure 2 Evolution of system-level performance metrics for 20 UAVs throughout their run-to-failure simulations.

As a post-processing step, the final flight in which failure occurred was removed from each trajectory, and both performance metrics were normalized to the interval $[0, 1]$. The maximum healthy value observed across the dataset was mapped to 1, and the respective

failure threshold to 0. Among the 20 UAVs, failure was triggered by the cumulative position error in 9 cases, and by SOC depletion in 11 cases. We randomly selected 15 UAVs for training and 5 for testing.

4 Results

We applied our method described in Section 2 using the degradation model described in Section 3 to our generated dataset described in Section 3.1. Then we have our latent space $X = (\beta, \lambda, \rho, \mu, \alpha)$, defined by the degradation model (9).

4.1 Training

The first step of our method is to estimate the initial distribution of the EOL particle filter, denoted $P(X_0)$. To do this, we learn an individual parameter vector $x_0^{(i)}$ for each training unit $i = 1, 2, \ldots, n$ by maximizing the likelihood of its degradation data and using the performance degradation model defined in equation (9). In our dataset, the number of training units is $n = 15$. So the initial estimation of the latent distribution $P(X_0)$ is given by equation (5) for the learned values $x_0^{(i)}$, $i = 1, 2, \ldots, 15$.

Figure 3 shows the predictive distribution of the particle filter deduced in equation (6),

$$P(T_s(X_0) = t) \approx \frac{1}{N} \sum_{i=1}^{N} P(T_s(x_0^{(i)}) = t), \quad s \in [0, 1], \quad t > 0$$

for the initial step $k = 0$ and without considering the augmentation step (i.e. $N = n$). We include the augmentation step in the next subsection because the augmentation noise variance is a hyperparameter.

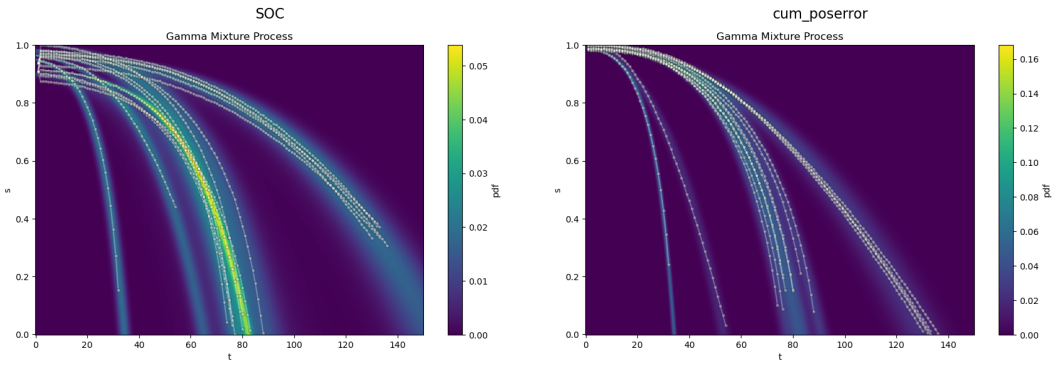


Figure 3 Initial prediction probability density functions for the degradation gamma mixture model learned from training data, shown for both performance metrics. These define the initial belief distribution of the particle filter, $bel_0(x) = p(X_0 = x)$. Circled points denote the training data. For illustration purposes, we have not included the augmentation step, so the number of particles N is equal to the number of training units n. At this stage, no observations have been received and the distribution is based solely on the particles learned from data.

4.2 Evaluation

To learn the correction strength and the variance of the PF propagation noise, we learned a function f that maps the current observation to these six parameters: κ (as defined in equation (4)) and the diagonal entries of Σ as defined in equation (3)) corresponding to the

noise of the degradation model parameters $(\beta, \lambda, \rho, \mu, \alpha)$. The function is linear, except for a softplus activation applied to the output to ensure positivity. Since the observation is two-dimensional, $y_k = (t_k, s_k)$, and we included bias terms, we obtained a total of $3 \times 6 = 18$ hyperparameters for learning this function. An additional hyperparameter is used for the variance of the augmentation noise employed to build the initial distribution, resulting in a total of 19 hyperparameters.

We used Optuna [2] for hyperparameter optimization, employing its implementation of the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [4], which is well-suited for efficiently exploring complex and non-convex search spaces. We applied leave-one-out cross-validation. In each fold, the initial distribution was estimated using 14 out of the 15 training units, and the hyperparameters were optimized by evaluating performance on the remaining unit. For the experiments, we set the number of particles at $n_{\text{particles}} = 3010$ (we added 10 to make the number of particles a multiple of the 14 training unit fold). We applied our EOL particle filter method independently to both performance metrics: SOC and cumulative position error and at each time step, we evaluated the predictive accuracy using the log-likelihood of future observations under the predictive distribution of the EOL particle filter, as defined in equation (6).

4.3 RUL Prediction

To assess performance, we used three standard prognostic metrics: one for RUL prediction quality (root mean square error, RMSE) and two for uncertainty estimation (prediction interval coverage probability, PICP, and prediction interval normalized width, PINAW). RMSE captures overall prediction accuracy; PICP measures how often the ground-truth RUL falls within the prediction interval; and PINAW captures how tight the interval is, normalized over the range of RUL values.

Figure 4 shows the predicted RUL trajectories along with 95% confidence intervals for both predictive and uncertainty variance across the 5 test UAVs. The experiment was repeated 10 times to assess statistical robustness. The model effectively captures degradation trends and represents both aleatoric and epistemic uncertainty. As the system approaches EOL, the uncertainty naturally decreases due to the accumulation of informative observations. Table 1 reports the numerical results over the full lifetime and the last 50 flights, highlighting improved accuracy in late-stage predictions.

Table 1 RUL prediction performance on the simulated dataset using the proposed EOL PF method.

Prediction window	RMSE	PICP	PINAW
Full life	20.37	94.72%	39.95%
Last 50 flights	6.47	100.00%	57.71%

5 Conclusions

We introduced a data-driven particle filtering framework for system-level prognostics that estimates remaining useful life and its uncertainty directly from performance observations. Unlike traditional model-based approaches, our method operates without requiring a physical model of the system. Instead, it learns a probabilistic degradation process offline from historical data and updates this belief online using a modified PF. This structure naturally captures both stochastic and epistemic uncertainty, while enhancing the interpretability

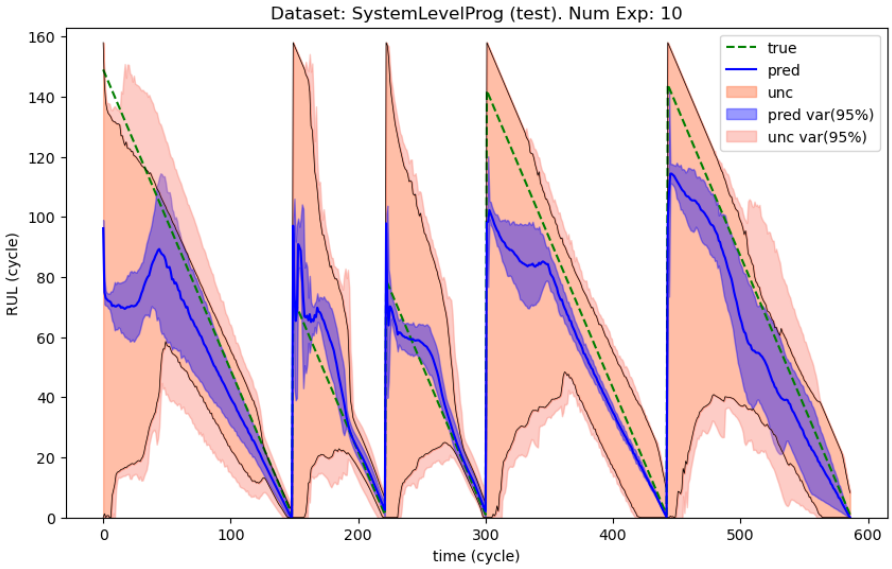


Figure 4 Predicted remaining useful life over flight cycles for the five test UAVs. The green dashed line shows the true RUL, the solid blue line is the mean predicted RUL, and the black line represents the mean of the 95% confidence interval across 10 repeated runs. The shaded regions show the 95% confidence intervals for the predictive variance (blue) and uncertainty variance (salmon), estimated using 1.95 standard deviations.

of purely data-driven models through performance-indexed degradation trajectories. We evaluated the approach on a high-fidelity UAV simulation environment and demonstrated that it produces accurate and uncertainty-aware predictions throughout the degradation lifecycle.

In future work, we plan to evaluate our method on the N-CMAPSS benchmark [3] and on real-world datasets to compare its performance against state-of-the-art approaches. Although N-CMAPSS lacks explicit performance metric trajectories, we are developing a strategy to infer them post hoc. We also plan to enhance the flexibility of the PF by learning the correction strength and propagation noise parameters by using neural networks. Due to the stochastic nature of PFs, this will likely require a differentiable approximation or a reparameterization-style trick to enable gradient-based training. Such extensions could improve both learning efficiency and expressiveness, allowing the model to better adapt to complex, nonlinear degradation behaviors.

References

1 Ibrahim Ahmed, Marcos Quinones-Grueiro, and Gautam Biswas. A high-fidelity simulation test-bed for fault-tolerant octo-rotor control using reinforcement learning. In *2022 IEEE/AIAA 41st Digital Avionics Systems Conference (DASC)*, pages 1–10. IEEE, 2022.

2 Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2623–2631, 2019. doi:10.1145/3292500.3330701.

3 Manuel Arias Chao, Chetan Kulkarni, Kai Goebel, and Olga Fink. Aircraft engine run-to-failure dataset under real flight conditions for prognostics and diagnostics. *Data*, 6(1):5, 2021. doi:10.3390/DATA6010005.

4 Anne Auger and Nikolaus Hansen. A restart cma evolution strategy with increasing population size. In *2005 IEEE congress on evolutionary computation*, volume 2, pages 1769–1776. IEEE, 2005. doi:10.1109/CEC.2005.1554902.

5 Yuanhong Chang, Fudong Li, Jinglong Chen, Yulang Liu, and Zipeng Li. Efficient temporal flow transformer accompanied with multi-head probsparse self-attention mechanism for remaining useful life prognostics. *Reliability Engineering & System Safety*, 226:108701, 2022. doi:10.1016/J.RESS.2022.108701.

6 Manuel Chao, Chetan Kulkarni, Kai Goebel, and Olga Fink. Fusing physics-based and deep learning models for prognostics. *Reliability Engineering & System Safety*, 217:107961, 2022. doi:10.1016/J.RESS.2021.107961.

7 Arnaud Doucet, Nando de Freitas, and Neil Gordon. *Sequential Monte Carlo Methods in Practice*. Springer, New York, 2001.

8 Neil J Gordon, David J Salmond, and Adrian FM Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *IEE Proceedings F (Radar and Signal Processing)*, 140(2):107–113, 1993.

9 Jian Guo, Zhaojun Li, and Meiyan Li. A review on prognostics methods for engineering systems. *IEEE Transactions on Reliability*, 69(3):1110–1129, 2019. doi:10.1109/TR.2019.2957965.

10 Cheng-Geng Huang, Hong-Zhong Huang, and Yan-Feng Li. A bidirectional lstm prognostics method under multiple operational conditions. *IEEE Transactions on Industrial Electronics*, 66(11):8792–8802, 2019. doi:10.1109/TIE.2019.2891463.

11 Dersin Pierre, Kristupas Bajarunas, and Manuel Arias-Chao. Analytical modeling of health indices for prognostics and health management. In *PHM Society European Conference*, volume 8, pages 11–11, 2024.

12 Darius V Roman, Ross W Dickie, David Flynn, and Valentin Robu. A review of the role of prognostics in predicting the remaining useful life of assets. In *27th European Safety and Reliability Conference 2017*, pages 897–904. CRC Press, 2017.

13 Abhinav Saxena, Jose Celaya, Bhaskar Saha, Sankalita Saha, and Kai Goebel. Metrics for offline evaluation of prognostic performance. *International Journal of Prognostics and health management*, 1(1):4–23, 2010.

14 Enrico Zio and Giovanni Peloni. Particle filtering prognostic estimation of the remaining useful life of nonlinear components. *Reliability Engineering & System Safety*, 96(3):403–409, 2011. doi:10.1016/J.RESS.2010.08.009.