A Two-Step Sub-Sampling Approach for a Computationally Efficient Particle Filter-Based Prognosis

A Two-Step Sub-Sampling Approach for a Computationally Efficient Particle Filter-Based Prognosis

Kevin Espinoza, Jorge García Bustos, Leonardo Baldo, *Graduate Student Member, IEEE*, Francisco Jaramillo-Montoya, David Acuña-Ureta, and Marcos Orchard

***Abstract*—Since their introduction, prognostic algorithms based on Particle Filtering (PF) have secured a leading position wherever reliable, grounded, and explainable predictions are required. If this success led to a plethora of research aimed at further enhancing accuracy and explainability, comparatively less attention has been paid to addressing the high computational costs associated with particle propagation. Against this backdrop, this article approaches the idea of sub-sampling the prediction steps, allowing the propagation of the system state directly between nonconsecutive time instants, thus obtaining an optimized tradeoff between accuracy and computational effort. In particular, driven by the idea that predictions should be more accurate towards the end of life, this research proposes a novel strategy based on sub-sampling with two different sampling frequencies. The two-phase sub-sampling scheme divides the prognostic horizon into two phases, identifying the ideal set of the three driving parameters: The frequency for the first phase, the switching time, and the higher frequency for the last phase. The computationally intensive tasks are delegated to an offline calibration phase, during which an XGBoost-based recommendation model is trained. In contrast, the online phase involves a single, rapid inference to recommend the optimal configuration, enabling the particle filter (PF) to operate with the selected two-rate propagation strategy while adhering to a user-defined computational time budget. On an experimental Li-ion battery-discharge case study, the framework reduces online computational time by up to 95% while keeping the relative error below 3% and achieving MAPE lower than 0.52% with respect to the standard PF. Comparative analysis demonstrates the framework's capacity to identify an optimal time-error trade-off, thus effectively merging those qualities that have made PF-based approaches the go-to solutions for explainable prognostic with the computational efficiency required for online implementation.**

***Index Terms*—Computational cost, genetic algorithm, lithium-ion batteries, particle filtering, sub-sampling, XGBoost.**

I. Introduction and Motivation

IN ENGINEERING, Prognostics and Health Management (PHM) can be defined as a discipline that aims to assess the current and future health of machines and their components by comprehensively analyzing the system operating conditions. By applying PHM concepts, the Remaining-Useful-Life (RUL) or the Time-of-Failure (ToF) can be estimated with the purpose of supporting decision-making processes [1]. In fact, with this information, preventive actions can be applied before the occurrence of a failure event to avoid expensive repairs or unscheduled downtime [2], [3], [4], [5]. PHM enables Original Equipment Manufacturers, operators, and technicians to anticipate failures, optimize maintenance strategies, and enhance overall reliability [6].

The primary aim of a PHM system is to provide a reliable estimate of the monitored health index in the future, together with an estimation of the uncertainty associated with the prediction [7]. Consequently, in recent years, significant research efforts have understandably focused on improving predictive accuracy. However, the other side of the coin cannot be overlooked: computational efficiency [8]. The two points of view must indeed go hand in hand to ensure a seamless and scalable deployment of PHM solutions. In fact, addressing the computational demands is equally crucial, as numerous applications require real-time execution in resource-constrained environments such as Commercial-Off-The-Shelf (COTS) platforms [9].

Among all possible PHM tools, Particle Filters (PFs) have established themselves as one of the most versatile, recognized, and adopted solutions to address prognostic problems [10], [11]. This prominence is principally attributed to their capability to handle nonlinear models, non-Gaussian uncertainty sources, and to accurately quantify the uncertainty associated with predictions [12]. However, their computational demand, inherently linked to the need for particle propagation, can become an obstacle for applications that require near-instantaneous responses [13]. In fact, PF-based approaches, as other standard prognostic algorithms, follow a Riemann-based sampling

Received 15 July 2025; revised 10 November 2025; accepted 14 January 2026. Date of publication 19 January 2026; date of current version 6 February 2026. This work was supported in part by ANID FONDECYT 1250036, Advanced Center for Electrical and Electronic Engineering, ANID Basal Project CIA250006. The work of David E. Acuña-Ureta was supported by ANID FONDECYT 11231148. The work of Jorge E. García Bustos has been supported by ANID-PFCHA/Doctorado Nacional/2022-21221213. Leonardo Baldo: This publication is part of the project PNRR-NGEU which has received funding from the MUR – DM 352/2022. Associate Editor: S. Taghipour. *(Corresponding author: Leonardo Baldo.)*

Kevin Espinoza, Jorge García Bustos, Francisco Jaramillo-Montoya, and Marcos Orchard are with the Department of Electrical Engineering, University of Chile, Santiago 1025000, Chile (e-mail: kevin.espinoza@ug.uchile.cl; jorge-garcia@ug.uchile.cl; francisco.jaramillo@uchile.cl; morchard@ing.uchile.cl).

Leonardo Baldo is with the Department of Mechanical and Aerospace Engineering, Politenico di Torino, 10129 Torino, Italy (e-mail: leonardo.baldo@polito.it).

David Acuña-Ureta is with the Department of Mechanical and Metallurgical Engineering, School of Engineering, Pontificia Universidad Católica de Chile, Santiago 7820436, Chile (e-mail: david.acuna@uc.cl).

Digital Object Identifier 10.1109/TR.2026.3655549

method [14], [15], which propagates the system state one instant at a time. In other words, starting at the present instant k_p, they sequentially calculate the states at the instants $k_p + i$, where $i = 1, 2, \ldots \infty$ is a positive integer that defines the progressing time step. In such Riemann-based sampling, state predictions occur at fixed time intervals, where the sampling period is the default system sampling time, often chosen based on worst-case scenarios rather than the actual dynamics of the system [16]. Although this methodology may ensure accurate estimations, it can incur a high computational cost, particularly when dealing with long prediction horizons. This is why the authors have selected PF strategies as the target for this study, as improving the efficiency of PF-based algorithms is crucial to facilitate their implementation in critical operational scenarios, ensuring the necessary accuracy in fault prediction without exceeding available hardware constraints. In scenarios such as these, accepting a moderate reduction in accuracy in exchange for decreased computation times can be the key factor that determines whether online implementation is feasible or not. Consequently, it is essential to identify an optimal tradeoff between accuracy and efficiency in prognostic algorithms.

To address this limitation, this article proposes a novel framework aimed at computationally efficient prognostic performance of PF-based approaches. The solution significantly reduces the computational burden of PF-based prognostic algorithms by adaptively adjusting the magnitude of the sampling frequency within a two-phase sub-sampling scheme. The concept of sub-sampling enables the direct propagation of the system state between nonconsecutive time instants. In other words, instead of calculating the intermediate states sequentially, the proposed algorithm uses linearization to compute the state at instant $k_p + J \cdot i$, where $i = 1, 2, \ldots \infty$ is a positive integer that defines ith step. This interval can be conceptualized in two equivalent ways: as a time-jump size J or as a sub-sampling frequency $F = f/J$, where f is the original sampling frequency. For clarity and flexibility in presentation, both terminologies are employed interchangeably throughout this article. It is important to note that the use of sub-sampling in prognostics inherently introduces a tradeoff between computational efficiency and estimation accuracy, which is directly dependent on the jump size J. Determining an optimal value for J is not straightforward and depends on several factors, such as the desired accuracy, the current operating conditions of the system, and the available computational resources.

It is common knowledge that, to provide actionable data, PHM predictions need to be more accurate around the ToF. In line with this objective, the authors decided not to subsample the predictions using a single frequency along the whole prediction horizon, but rather to utilize two distinct sampling frequencies, F_1 and F_2. By dividing the time horizon into two segments, a coarser sampling frequency is used during the initial phase of the prognostic horizon, where the focus is more on cost optimization. On the other hand, in the final segment, a finer sub-sampling approach is implemented to prioritize accuracy in the trade-off between precision and cost. As a result, the methodology divides the prediction horizon into two, finding the optimal combination of switching time T_{sw} and frequencies

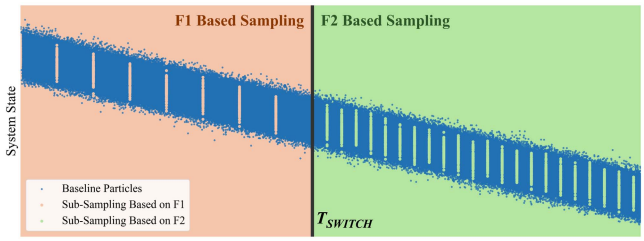


Fig. 1. Illustration of the roles of parameters F_1, F_2, and T_{sw} in the proposed prognostic strategy compared to the traditional baseline approach.

(F_1 and F_2) to meet user requirements in terms of maximum running time.

The search for the optimal parameter set is delegated to an offline-trained XGBoost-based recommendation model, which leverages a pre-generated parameter pool constructed via a Genetic Algorithm (GA). This approach ensures that the selected Prognostic Algorithm Parameters (PAPs), namely F_1, T_{sw}, and F_2, enable a prediction process that adheres to computational constraints while maintaining high prognostic accuracy. Although the sub-sampled PF significantly reduces computational cost compared to the standard PF, training the recommendation model remains computationally intensive. However, this burden is confined to an offline calibration phase and does not impact the online PF implementation. During calibration, the GA explores the parameter space, and XGBoost models are trained using data from resource-heavy standard PF runs. As a result, the online phase is streamlined to lightweight parameter selection and PF execution. Fig. 1 illustrates the influence of the three PAPs on the particles generated by the prognostic algorithm. The developed methodology has been designed to be applicable to a wide range of engineering challenges, as long as the process can be represented by a discrete-time state-space model that satisfies the Markov property, and the process noise can be represented by a Gaussian distribution. With the aim of testing and validating the proposed approach, the overall methodology has been applied to an experimental Li-Ion battery discharge dataset as a case study.

The main contributions of the article are as follows.

1) We propose a computationally efficient framework that can be seamlessly integrated into PF-based prognostic schemes without major changes in the PF methodology. Before asset operation, an offline baseline run is needed to create a parameter mapping and train a parameter recommendation scheme. At run time, a fast inference selects the configuration and the PF executes with substantially lower computational cost while maintaining comparable predictive accuracy.
2) The framework lets the user set the desired allowable maximum running time, then the recommendation scheme selects the optimal configuration of PAPs to obtain the most loyal results.
3) The recommendation scheme is powered by a set of three XGBoost models trained offline, on a pregenerated data set created with a GA that efficiently explores the solution space. Then, we use them online to pick a two-rate sub-sampling configuration; applying this configuration at run

time significantly reduces computational effort without degrading predictive accuracy.

4) The framework has been validated on an experimental battery-discharge case study, demonstrating the effectiveness of the proposed solution. After the offline step, the computational effort is reduced by up to 95% while keeping the relative error below 3% and MAPE below 0.52% with respect to the baseline PF.

A. Related Work

Motivated by the widespread adoption of PF-based algorithms in recent years, some research efforts have been directed toward minimizing computational complexity associated with failure prognostic algorithms. Although fewer in number than accuracy-focused publications, these investigations encompass a spectrum of approaches, from high-level algorithmic strategies to low-level architectural modifications.

In general, efforts to reduce the computational burden of PF prognostics can be divided into three main directions. The first is selective execution. A divergence–based trigger presented in [17] updates the filter only when the distance between the prior and posterior distributions exceeds a prescribed threshold, thus avoiding unnecessary calculations while maintaining accuracy when the threshold is well tuned. Building on the same idea, an Auxiliary PF described in [18] redistributes particles dynamically and estimates model parameters online; this improves robustness under nonstationary conditions, though it still relies on user-defined thresholds. The second direction capitalizes on hardware parallelism. Early work in [19] demonstrated that moving PF kernels to graphics processors enables near-real-time performance to be transformed into accurate real-time execution. In [13], a ten-fold acceleration for ballistic target tracking was reported after parallelizing state propagation and identifying that stage (not resampling) as the principal computational constraint. Resampling has since been redesigned for massively parallel hardware. The Uphill-Fast algorithm in [20] enhances numerical stability and aligns memory access, while the Megopolis scheme in [21] eliminates tuning parameters while maintaining estimation quality. A backtracking PF combined with geospatial constraints reached real-time performance on embedded platforms in [22]. Although parallelization yields significant gains, the benefit diminishes when the particle count drops below roughly 10^4, a typical ceiling in embedded prognostic systems. The third direction pursues algorithmic reformulation. Coupled Markov chains in [23] remove initialization bias and shorten Monte-Carlo simulations, while [24] reduces the search space for energy-optimal routing through Monte-Carlo Tree Search. In the prognostics domain, an uncertain-event likelihood proposed in [8] reframes the End-of-Discharge prediction, avoiding explicit threshold-crossing simulations. The uncertain PF introduced in [25] adapts the update and resampling rules to abrupt changes in degradation, thus speeding up convergence in battery and capacitor case studies. An improved Unscented PF within a Lebesgue-time framework, reported in [26], increases the accuracy of Li-ion cells under variable loads. A hybrid data-driven solution that combines dropout-based Long Short-Term Memory networks with LightGBM and physics-informed priors, presented in [27], off-loads part of the computational effort usually handled by the PF while retaining predictive performance.

Despite these advances, three limitations remain. First, there is no general mechanism for balancing computational effort against prognostic fidelity; most contributions report average speed-ups without explicit error guarantees. Second, key hyperparameters, sub-sampling factors, divergence thresholds, and switching instants are still chosen by trial and error, limiting portability across platforms and duty cycles. Third, few studies demonstrate strict compliance with user-specified execution-time budgets; results are generally stated as mean accelerations rather than hard real-time guarantees. These open questions motivate the framework developed in the present study. Our method casts the accuracy–efficiency compromise as a global optimization problem that is explored offline with a GA. The resulting database of prognostic parameter triplets $\langle F_1, F_2, T_{\text{sw}} \rangle$ is then queried online by three lightweight XGBoost models, which map the current state of the system and a user-defined time budget to an optimal configuration. The selected triplet drives the two-phase sub-sampling strategy, which guarantees robust and verifiable compliance with execution-time limits while remaining agnostic to the particular PF implementation.

It is worth noting that the literature includes deep learning-based prognostic algorithms and end-to-end recurrent models that directly generate a RUL distribution, offering an alternative approach that eliminates the need for online particle propagation [28], [29]. However, these algorithms lack intrinsic guarantees of bounded execution time, which is a critical requirement for real-time decision support systems. Meeting such constraints typically necessitates explicit deadline-aware scheduling or the use of anytime filtering schemes [30]. Moreover, these methods generally lack explicit mechanisms for regulating the lower quantiles of the failure-time distribution–most notably the α-quantile that delineates the Just-in-Time Point within the early-risk region [31]–unless they are trained using quantile-sensitive objectives, such as quantile regression neural networks [32]. Last but not least, deep-learning-based prognostic algorithms and end-to-end recurrent models bypass the state-space structure that particle filtering exploits to propagate uncertainty under Markovian process assumptions, which is valued for traceability and physical consistency [33]. In contrast, our framework enforces a strict upper bound on execution time by incorporating a two-rate PF-based subsampling scheme $(F_1, T_{\text{sw}}, F_2)$, which integrates an offline GA-driven exploration phase with an XGBoost-based selector that maps operating conditions and execution-time constraints to optimal prognostic algorithm parameters. During online operation, the framework minimizes computational overhead by executing a PF-based algorithm conditioned on a lightweight selection of prognostic algorithm parameters. The design follows a modular principle in which alternative models or control laws can replace the current instantiation of the PAP selector without altering the enforcement of the maximum execution time or the PF-based uncertainty propagation, which

positions our approach as complementary to end-to-end deep models and retains formal control over latency and risk-sensitive behavior.

From an experimental standpoint, we deliberately adopt the conventional PF implemented at the full sampling rate as the reference baseline, rather than reimplementing each of the efficiency-oriented variants summarized above. This choice isolates the contribution of the proposed two-rate sub-sampling and offline PAP selection from hardware-specific accelerations or additional algorithmic heuristics, and preserves a clear one-to-one correspondence between the baseline and accelerated configurations (same state-space model, number of particles, and resampling scheme). At the same time, the framework is designed to be complementary to these advanced methods: In principle, any of the selective-update, parallel, or reformulated PF kernels discussed in this section could be used as the underlying propagator within our two-phase sub-sampling scheme, which would then modulate their propagation schedule under a specified execution-time budget instead of directly competing with them.

The rest of this article is organized as follows. Section II reviews the theoretical foundations of PF and PF-based prognostics. Section III details the proposed framework for computationally efficient failure prediction. Section IV applies the method to a Li-ion battery discharge case study, describing the implementation of the prognostic algorithm, the search for optimal sampling periods, and the test protocol. Section V presents and discusses the results. Finally, Section VI concludes this article.

II. Theoretical Background

A. Particle Filters (PFs)

PFs are a class of Monte Carlo algorithms used to estimate the state of nonlinear, non-Gaussian dynamical systems. More specifically, PFs are useful to represent the posterior Probability Density Function (PDF) of a dynamical system target state when a closed-form solution of the Bayesian Filtering (BF) equation is not obtainable. In fact, when the system under study is nonlinear and non-Gaussian, obtaining an exact analytical solution to the BF problem is generally not feasible [34]. PFs approach the formulation of the BF by simulating a set of possible states and their evolution over time.

To accomplish this, PFs approximate the posterior PDF of the target state by propagating a set of $N_p \gg 1$ random samples (called particles) x_k^i with associated weights w_k^i, so that $\{x_k^{(i)}, w_k^{(i)}\}_{i=1}^{N_p}$.

As new observations become available, the particle weights are updated according to their likelihood. Resampling can then be performed to focus on the most relevant regions of the solution space. As a result, the posterior PDF of x_k at any given time point k, based on the measurements $z_{1:k}$ acquired up to the instant k, can be approximated as reported in (1), where $\delta(\cdot)$ is the Dirac delta function [12]

$$p(x_k|z_{1:k}) \approx \sum_{i=1}^{N_p} w_k^{(i)} \delta(x_k - x_k^{(i)}). \tag{1}$$

Given a new observation z_k, the importance weights $w_k^{(i)}$ of the particles are updated as follows:

$$w_k^{(i)} = w_{k-1}^{(i)} \cdot \frac{p\left(z_k|x_k^{(i)}\right) p\left(x_k^{(i)}|x_{(k-1)}^{(i)}\right)}{q\left(x_k^{(i)}|x_{(k-1)}^{(i)}, z_k\right)} \tag{2}$$

where $p(z_k|x_k^{(i)})$ is the likelihood, $p(x_k^{(i)}|x_{k-1}^{(i)})$ is the state transition prior, and $q(\cdot)$ is called the importance density distribution [35], also known as the proposal distribution. The proper selection of $q(\cdot)$ is key in determining the performance of the implementation of PF [15], as it influences the particle weight variance and the effectiveness of the filter. Finally, it is clear that, to propagate the particles and predict future system behavior, a state-space transition equation must be available to describe the system's underlying dynamics.

B. Failure Prognosis Based on Particle Filters

PFs have gained unparalleled attention in the field of PHM, because of their ability to estimate and predict the evolution of degradation processes in complex systems, and because of their ability to capture both epistemic and aleatoric uncertainty. In fact, in the context of failure prognostics, a central step is devoted to estimating the ToF (or RUL). After the particles are propagated forward in time, each trajectory can be evaluated to determine the earliest time at which the failure condition is met.

Consider the binary stochastic process $\{E_k\}_{k\in\mathbb{N}}$ that depends on the state x_k. $E_K = 1$ indicates that a failure event ϵ occurs at instant k, while $E_k = 0$ indicates its nonoccurrence ϵ^c. If the notion of certain events [8] is considered, the ToF can be defined as the time at which the event ϵ occurs for the first time, considering the prediction carried out at the present instant k_p. This time is denoted as $\tau_\epsilon(k_p)$, and its probability mass function is $\mathbb{P}(\tau_\epsilon = k)$. $\mathbb{P}(\tau_\epsilon = k)$ can be expressed as the probability that the failure event occurs at instant k, but has not occurred between instants k_p to $k-1$. After a few mathematical steps explained and demonstrated in [8] and [36], the resulting mathematical formulation is reported as follows:

$$\hat{\mathbb{P}}_{N_p}(\tau_\epsilon{=}k){=}\frac{1}{N_p}\sum_{i=1}^{N_p}\mathbb{1}(x_k^{(i)})\prod_{j=k_p+1}^{k-1}\left(1-\mathbb{1}(x_k^{(i)})\right) \tag{3}$$

where the indicator function $\mathbb{1}(x_k^{(i)})$ signals whether the ith particle reaches the failure threshold $\mathcal{T}$ at time k. Moreover, $x_k^{(i)}$ denotes the state of the ith particle at time k, obtained by propagating the system dynamics according to a state-space model of the form

$$x_{k+1} = f(x_k, u_k) + \omega_k, \qquad z_{k+1} = h(x_{k+1}) + \eta_{k+1} \tag{4}$$

where u_k is the exogenous input, ω_k is the process noise, η_k is the measurement noise, and $f(\cdot,\cdot)$ is an arbitrary two-variable function that could be very well nonlinear. Finally, by carefully observing this expression, it can be noticed that, in the case of particles sharing the same weights, the probability function

$\mathbb{P}(\tau_\epsilon = k)$ can be further approximated

$$\mathbb{P}(\tau_\epsilon = k) \approx \frac{\text{N}^\circ \text{ traj. crossing } \mathcal{T} \text{ for the first time at } k}{\text{N}^\circ \text{ of simulated trajectories}}. \quad (5)$$

In this way, it is straightforward to construct a histogram to represent the probability mass function of τ_ϵ. The formulation reported in (5) is therefore the one adopted in this article.

III. Proposed Methodology

As mentioned above, a traditional Riemann-based prognostic algorithm uses the default sampling time of the system, predicting the state at every instant during the prognostic horizon. In contrast, and to drastically reduce the computational burden, a prognostic scheme based on sub-sampling is proposed, thus enabling the possibility to calculate the state of the system only at certain instants throughout the prediction horizon. The envisioned scheme requires two adaptations as follows.

1) First of all, the state equation has to be adapted to obtain a mathematical expression that directly relates the state at instant $k_p + J$ to the state at the previous instant k_p. In this article, this is performed through a system model linearization.
2) Once the mathematical formulation supports sub-sampled evolution, the next step involves selecting the optimal PAPs set (F_1, T_{sw}, F_2) based on the user's requirements regarding maximum execution time.

A. Adaptation of the State Equation for Sub-Sampling Prognostic

Following the mathematical hypothesis stated in Section I, the general form for describing a system with the aforementioned characteristics is the state-space equation expressed in (4). Let us consider only the first equation, since the measurement model, despite relevance in the filtering stage, is excluded in forward RUL estimation.

To derive the mathematical expression that directly relates the state of the system at an arbitrary time k to the state at $k + J$, J instants later in the more general nonlinear case, a classical linearization approach is selected, where each step is locally approximated via a first-order Taylor expansion around a nominal operating point

$$\begin{aligned} x_{k+J} &= f(x_{k+J-1}, u_{k+J-1}) + \omega_{k+J-1} \\ &\approx f(x_k, u_k) + \left.\frac{\partial f}{\partial x}\right|_{(x_k)} (x_{k+J-1} - x_k) \\ &\quad + \left.\frac{\partial f}{\partial u}\right|_{(u_k)} (u_{k+J-1} - u_k) + \omega_{k+J-1}. \end{aligned} \quad (6)$$

After some formal mathematical steps omitted for the sake of brevity, along with a recursive substitution of the linearized state-space equation, the following linearized expression for x_{k+J} as a function of x_k can be derived (7). It is important to note that linearization inherently involves a reduction in accuracy due to the exclusion of higher-order terms and the assumption that the exogenous input remains constant between consecutive samples.

In practice, this reduction in accuracy is the tradeoff required to obtain a more computationally efficient prognostic algorithm

$$\begin{aligned} x_{k+J} &\approx f(x_k, u_k) + \tilde{M}_k(J)\delta x_k \\ &\quad + \sum_{i=1}^{J-1} M_k^{i-1} N_k \left(u_{k+J-i} - u_k\right) \\ &\quad + \sum_{i=0}^{J-1} M_k^i \omega_{k+J-1+i} \end{aligned} \quad (7)$$

where

$$\delta x_k = f(x_k, u_k) - x_k, \quad (8)$$

$$\tilde{M}_k(J) = \sum_{i=1}^{J-1} M_k^i, \quad (9)$$

$$M_k = \left.\frac{\partial f}{\partial x}\right|_{(x_k, u_k)} \text{ is the state sensitivity matrix,} \quad (10)$$

$$N_k = \left.\frac{\partial f}{\partial u}\right|_{(x_k, u_k)} \text{ is the input Jacobian.} \quad (11)$$

The multiplications in (7) are 1-D operations when the state x_k is univariate but may involve matrix operations in the general case of multivariate data, as the formulation remains unchanged. Moreover, (7) can be simplified to make the implementation more efficient by further elaborating the last two terms.

1) The first simplification consists in assuming that the exogenous input remains constant during the interval $(k, k + J]$, which means that $u_{k+J-i} - u_k = u_{k+J-1} - u_k$ holds for any i such that $1 \leq i \leq J - 1$. This allows the summation of exogenous terms to be factored in, as shown as follows:

$$\begin{aligned} &\sum_{i=1}^{J-1} M_k^{i-1} N_k \left(u_{k+J-i} - u_k\right) \\ &\quad = \left(\sum_{i=1}^{J-1} M_k^{i-1} N_k\right)\left(u_{k+J} - u_k\right). \end{aligned} \quad (12)$$

It is important to emphasize that this expression requires only two values of the exogenous input, u_k and u_{k+J}, which is particularly convenient since the model must satisfy the Markov property. In fact, one of the key properties of Markov chain modeling is that the probability of transitioning from one state to another in J steps can be computed by raising the generic transition matrix P to the Jth power ($P^{(J)} = P^J$). This property enables cost-effective calculation of sampled transition probabilities using matrix operations, as the corresponding power of the transition matrix is calculated just once.
2) For the last term, the proposed simplification avoids the need to extract a large number of samples from the random variable, as required in principle by (7). To achieve this, the accumulated effect of noise over the interval of J instants is modeled as a single Gaussian random variable, but with its covariance matrix W_p given by (13), calculated using

the properties of the variance for a linear combination of random variables

$$W(J) = \sum_{i=0}^{J-1} M_k^i W (M_k^i)^T \tag{13}$$

where W is the covariance matrix of the original process noise.

With these simplifications, the final expression to approximate (7) is obtained, as shown in (14).

$$x_{k+J} \approx f(x_k, u_k) + \tilde{M}_k(J)\delta x_k + \left(\sum_{i=1}^{J-1} M_k^{i-1} N_k \right) \left(u_{k+J} - u_k \right) + \omega_k(J) \tag{14}$$

where $\omega_k(J)$ is a single sample of a random variable of the same type as the process noise, but whose covariance matrix is given by $W(J)$, as shown in (13).

B. Sub-Sampling Prognostic Scheme

Following the presentation of the mathematical formulation of the state equation and the derivation of the prognostic expression for an arbitrary sampling interval J, the question remains as to how to select the optimal set of PAPs.

Since the proposed prognostic scheme is intended for real-time prognostics, it must include a module that can dynamically and efficiently suggest optimal PAPs. In fact, comparing prognostic results online with different PAPs would lead to excessive computational time due to the required multiple runs of the baseline PF algorithm with different initial conditions, affecting overall optimization. The best choice is to move the computational burden into the offline phase, where the baseline PF-algorithm is run. Then, during the online operation, the optimal set of PAPs is suggested by a pre-trained Machine Learning (ML) model with minimal computational cost, and the optimized prediction can be run with the suggested parameters.

For this purpose, an XGBoost model has been selected as the ML model that can, during operation, select appropriate PAPs based on the current system state and the execution time requirement given by the user. Like any supervised machine learning model, XGBoost requires training on a representative dataset $\mathcal{D}$. In this context, the dataset consists of a set precomputed PAPs, in the shape of three-value tuples (F_1, T_{sw}, F_2) obtained under a variety of initial system conditions $x_0 \in \mathcal{X}$ and computational time constraints $T_{\max} \in \mathbb{R}_+$. The creation of this PAP pool involves two main steps. First, a computationally intensive offline run of a baseline prognostic model is performed for each initial condition x_0. Then, a GA is used to explore the parameter space and generate a diverse and representative dataset of PAPs across the range of operating scenarios. Fig. 2 illustrates the primary stages of the process, highlighting which steps are completed offline and which are performed in real time.

Importantly, PAP calibration is conducted entirely offline to generate the configurations that will later be suggested by the XGBoost model—no calibration occurs during run-time, ensuring that the reported latency reflects only the online execution. In other words, the online stage is computationally lightweight,

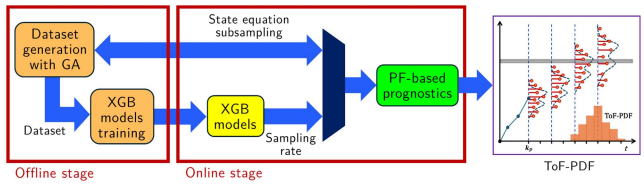


Fig. 2. Diagram of the proposed prognostic scheme. The XGBoost models trained during the offline stage are used in real-time to select the PAPs. In the online phase, the prognostic is performed using the recommended PAPs to obtain the probability distribution of the ToF. Consequently, the online cost is drastically reduced by sub-sampling and not impacted by dataset generation or model training, which occur offline; online processing is limited to a fast surrogate selection and PF execution.

as it relies on sub-sampling and is unaffected by the GA-based dataset generation or XGBoost training, both confined to the offline calibration phase. During online operation, the system performs only a fast surrogate inference to select the configuration, followed by the execution of the PF. Each stage of the algorithm is described in detail in the rest of this section.

1) PAP Dataset Generation With GA: A GA is used to map and save a pool of PAPs set in a wide range of conditions (x_0 and $T_{\max}$). GAs are population-based metaheuristics inspired by biological evolution, iteratively refining candidate solutions to converge towards global or local optima [37]. In GAs, each solution (individual) encodes its parameters as a chromosome $p = (F_1, T_{sw}, F_2) \in \mathcal{P}$. A fitness function evaluates each chromosome, guiding the selection of the fittest individuals for reproduction and ensuring progressive improvement. The search was performed using a modified GA implementation available online [38], where the roulette wheel method for chromosome selection was replaced with the tournament method. An exhaustive review of GAs can be found in [37]. With the aim of achieving higher temporal resolution as the ToF approaches, a lowest frequency is used at the beginning of the prognostic horizon, and a higher one towards the end.

The GA fitness function has been defined leveraging the formulation of Just-in-Time Point (JITP), a metric that aims to characterize the risk associated with decisions based on prognostic algorithms, assuming that this risk depends on the tails of the failure probability distribution [39], [40], [41]. The JITP can be defined as shown as follows:

$$JITP_\alpha = \min \{k \mid P(\text{ToF} \leq k) \geq \alpha\}. \tag{15}$$

That is, it represents the earliest point at which the probability of system failure exceeds a specified threshold $\alpha\%$. This parameter α thus allows the definition of an acceptable level of risk.

The formulation of the fitness function considers three different terms: 1) the difference between the JITP values obtained from the proposed and baseline approaches; 2) a penalty term that enforces the execution-time constraint; and 3) a quadratic regularization component over the PAP N_2, with N_2 such that $F_2 = f/N_2$, which turns out to be the most critical parameter, discouraging excessively large values for it. These elements are

combined as shown in (16). Consequently, an offline baseline PF algorithm must be executed for each initial condition x_0 in order to provide the reference JITP values

$$f(\mathbf{p}; x_0, T_{\max}) = \max_{\alpha} \left(\mathbb{E}[\text{JITP}_\alpha(\mathbf{p}, x_0)]\right) + \tau \cdot H\left(T_{ex}(\mathbf{p}, x_0) - T_{\max}\right) + \lambda \cdot {N_2}^2 \quad (16)$$

where τ is a constant quantifying the cost of not respecting the execution time constraint, T_{ex} is the execution time, $H(\cdot)$ is the Heaviside step function, and λ is a weighting factor that scales the regularization term to ensure a balanced contribution among all terms. Moreover, the error $\mathbb{E}[\text{JITP}_\alpha(\mathbf{p}, x_0)]$, with α an arbitrary percentage, is defined as follows:

$$\mathbb{E}[\text{JITP}_\alpha(\mathbf{p}, x_0)] = \frac{\left|JITP_{\alpha B} - JITP_{\alpha A}\right|}{JITP_{\alpha B}} \times 100 \quad (17)$$

with $JITP_{\alpha B}$ and $JITP_{\alpha O}$ representing the JITP values with respect to the standard strategy (B, baseline) and the approach proposed in this research (O, optimized).

The fitness function has been constructed considering low values of $\alpha = 5, 10, 15\%$ because the aim is to maintain a conservative criterion to provide an alert for the ToF. The value of the constant τ has been obtained through trial and error, as it is only required to be sufficiently high to penalize solutions whose computation time strongly exceeds the established limit to be discarded. On the other hand, the value of the weighting factor λ has been calibrated to balance the regularization term introduced to discourage excessively large frequency divisors N_2, which would lead to overly coarse sampling and degraded prognostic accuracy. At the same time, λ must not overpenalize moderate divisor values, so as to avoid unnecessarily fine sampling that would compromise computational efficiency. Similar to τ, its value was determined empirically through iterative trial and error adjustment until achieving a satisfactory trade-off between accuracy and execution time. The GA is then executed multiple times for each pair of input variables $(x_0, T_{\max})$, to obtain several alternative sets of optimal parameters, and thus a large dataset to subsequently train the XGBoost models. In each GA run, the algorithm first receives the prognostic results according to the standard method and compares them with hundreds of simulations performed using the proposed approximated approach, employing different sets of randomly generated parameters, trying to minimize f

$$\min_{\mathbf{p} \in \mathcal{P}} f(\mathbf{p}; x_0, T_{\max}). \quad (18)$$

For a grid of input conditions

$$(x_0^{(j)}, T_{\max}^{(j)}), \quad j = 1, \ldots, N_{\text{samples}}$$

the GA is run M times to gather a dataset $\mathcal{D}$ such that

$$\mathcal{D} = \bigcup_{j=1}^{N_{\text{samples}}} \bigcup_{m=1}^{M} \left\{ \left(x_0^{(j)}, T_{\max}^{(j)}, \mathbf{p}^{*(j,m)} \right) \right\}. \quad (19)$$

The resulting dataset $\mathcal{D}$ is then used to train the XGBoost suggestion models with mapping

$$(x_0, T_{\max}) \mapsto (F_1, T_{sw}, F_2). \quad (20)$$

2) XGBoost Suggestion Models: Three cascade XGBoost models (one for each PAP) were trained so that they could determine, for any feasible initial state x_0, a combination of parameters that respects the maximum execution time $T_{\max}$ established by the user, while aiming to minimize the loss in accuracy for the ToF estimation. It is important to note that $T_{\max}$ acts as an upper bound: if the best solution is found with a $T_{ex} < T_{\max}$, it is adopted. The three specialized models were trained for each of the three PAPs, and optimal hyperparameters were selected using GridSearch and a greedy algorithm [42]. The set of PAPs, obtained as output of the XGBoost models, can then be employed to run the prognostic simulations with an optimal sub-sampling. The PAP selector is interchangeable and can be implemented with different ML models or control laws without altering the two-rate PF subsampling scheme or the enforcement of the time budget. This flexibility ensures that the design objectives for both the offline and online stages remain intact.

IV. Case Study

To validate the proposed optimized prognostic strategy, which is model-agnostic by design, a case study analyzing battery discharge in an electric bicycle operating on a predetermined route (as outlined in [43]) is selected. This scenario offers a realistic and dynamic context for evaluating the effectiveness of the method in monitoring degradation and estimating ToF under varying system conditions.

A. Battery Model

Since the proposal presented in this research focuses on PF-based failure prognostic algorithms, a state-space model for the battery discharge dynamics is required. Note that the focus of this paper is on the PHM scheme, rather than on the battery model itself, whose detailed discussion is outside of its scope. Readers interested in battery discharge models are encouraged to consult [43]. The considered model corresponds to a simplified version of the one proposed in [39], and the selected state-space model is shown in (21) and (22). The first equation represents the dynamics of the only state variable of the system, the State of Charge (SoC) x_k, in terms of the power delivered from the battery. The SoC refers to the ratio between the remaining energy and the maximum nominal storage capacity Q_c. The second equation corresponds to the observation equation, which characterizes the voltage measured at the battery terminals $v(x_k)$ as a function of the SoC

$$x_{k+1} = x_k - v(x_k) i_k \Delta t Q_c^{-1} + \omega_k \quad (21)$$

$$v(x_k) = v_l + (v_o - v_l) e^{\gamma(x_k - 1)} + \alpha v_l (x_k - 1) + (1 - \alpha) v_l (e^{\beta} - e^{\beta\sqrt{x_k}}) - i_k \cdot R + \eta_k. \quad (22)$$

TABLE I
THE BATTERY MODEL PARAMETERS ARE THE SAME AS IN [43], EXCEPT FOR THE INTERNAL RESISTANCE R WHICH, IN THE CONTEXT OF THIS CASE STUDY, IS CONSIDERED CONSTANT AND SET TO THE MEAN OF THE INITIAL STATE PROBABILITY DISTRIBUTION DURING THE FILTERING STAGE

Parameter	Description	Values
R	Internal resistance	0.26 Ω
Δt	Sample time	1 s
α	Observation parameter	5.319×10^{-3}
β	Observation parameter	11.505
γ	Observation parameter	1.5538
v_0	Observation parameter	41.405 V
v_l	Observation parameter	33.481 V
Q_c	Maximum energy capacity	1389900 J
σ_ω	Process noise std dev	10^{-6}
σ_η	Measurement noise std dev	0.9 V

Here, i_k represents the discharge current (exogenous input) and R the internal impedance of the battery. On the other hand, the parameters α, β, γ, v_0, and v_l are battery parameters [44], whose values are estimated using the procedure presented in [39]. Finally, the sampling time is denoted by Δt. The simplifications in relation to the model presented in [39] include the use of constant internal resistance. Consequently, the current model does not account for the influence of variables such as SoC, State of Health (SOH), and discharge currents on the internal impedance. These minor simplifications do not impact the final results of the model and are considered justified, since in this context, the model is used only as a case study for strategy testing.

1) Failure Condition. Cut-Off Voltage: De facto, this research focuses on short-term failure prediction and does not include a long-term battery SOH analysis. As commonly performed in battery-related PHM studies, the ToF coincides with the battery End of Discharge (EoD) [45], [46]: $\epsilon = EoD$. Moreover, as already stated in Section II-B, the notion of certain events [8] is considered, such that the EoD can be seen as a threshold crossing-triggered event. In battery-powered systems, the battery is generally managed by a Battery Management System (BMS), which, to preserve the battery integrity and to avoid risky overheating conditions, cuts off the battery from the circuit when the voltage across the battery terminal v drops lower than a certain value, called cut-off voltage V_c. As a result, as commonly performed in the literature [43], the EoD event is identified with the instant k when $v(k) < V_c$.

B. Characteristics of the Battery

The primary parameter from the datasheet that defines the failure condition is the cut-off voltage $V_c = 33$V. Furthermore, Table I shows the specific values used for the parameters of the battery discharge model, previously introduced in (21) and (22) as obtained by [43], except for the simplifications previously explained.

C. Exogenous Input: Discharge Current Profile

Given the inherent uncertainty in prognostic problems regarding the future inputs of the system being monitored, it is necessary to statistically characterize them in a way that enables the prediction process. In particular, a point-by-point estimate is not required; instead, a probabilistic model is more appropriate. In the case study employed in this research, the exogenous input of the system, the discharge current, has been modeled using a two-state Markov chain approach, as proposed in [7] and later applied in [43]. The current values used for the states of the Markov chain are shown in (23), while the resulting transition matrix, computed using maximum likelihood estimators, is reported in (24). While the Markov Chain model provides valuable discharge current predictions, it is essential to recognize that the model's purpose is to statistically characterize the current trend with probabilistic forecasting, not exact replication of measurement-level discharge currents

$$C = \begin{bmatrix} 3.4979 & 5.0526 \end{bmatrix} \tag{23}$$

$$P = \begin{bmatrix} 0.9388 & 0.0612 \\ 0.0554 & 0.9446 \end{bmatrix}. \tag{24}$$

Further details on the methodology can be found in [43].

TABLE II
HYPERPARAMETER VALUES OF THE XGBOOST MODELS FOR DETERMINING N_2, N_1, AND T_{sw} IN EACH FORECAST

Output	N. tree	Max Depth	lr	γ_x	λ_x	α_x
N_2	13	4	0.2	0.25	0	1
N_1	89	4	0.05	0.25	15	25
T_{sw}	61	3	0.1	0	10	0

D. GA and XGBoost

In the offline phase, the search for optimal parameters through the GA has been carried out over a wide range of SOC, from $x = 1$ to $x = 0.2$, spaced by a difference of $\Delta x = 0.05$, totaling 17 values. On the other hand, simulating the user requirements, four computation time limits have been chosen, $T_{\max} = 5, 4, 3, 2$ s. Furthermore, to further restrict the search space, both N_1 and N_2, were limited to the range $[1, 200]$, while the possible values of T_{sw} were restricted to the interval $[1, k_{20\%}]$, where $k_{20\%}$ is defined as the instant when a probability of 20% that has already failed is reached, according to the CDF calculated using the classical approach. In other words, $k_{20\%} = JITP_{20\%B}$. Finally, the termination conditions for the GA included a maximum of 40 generations, each with a population of 100 individuals, or a limit of 10 iterations without improvement in the found solution. In total, the GA was executed in such a way that a dataset of 680 entries was obtained. With reference to (19), N_{samples} is equal to 68 samples (17 and 4 different x_0 and $T_{\max}$ values, respectively) and $M = 10$. Uniform mutation and crossover probabilities were set to 0.9, with 30% of parents retained in each generation and an elitism rate of 1% , The dataset has then been split into 80% for training and the remaining 20% for validation to train the XGBoost models. The XGBoost hyperparameters were selected with a grid search and are reported in Table II.

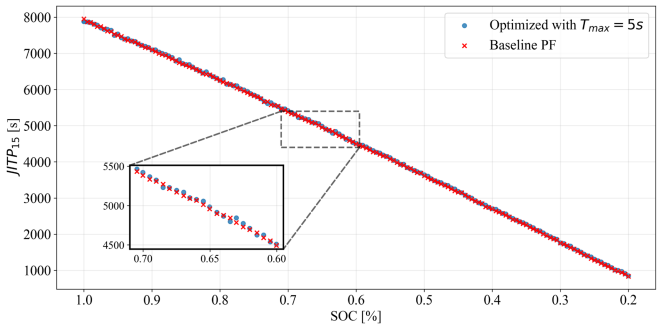


Fig. 3. Baseline JITP trend with different SoC x_0 initial condition and optimized trend with $T_{\max} = 5s$. The macroscopic differences are negligible, and a zoomed-in view is provided.

E. Simulation

To evaluate the proposed methodology, more than 1500 prognostic simulations were performed using data derived from [43], with a baseline prognostic update frequency of 5 s. These initial conditions span SoC values from 100% to 20% , the minimum recommended threshold according to [47], thereby enabling the assessment of the approach throughout the discharge cycle. For each x_0 SOC level, both the standard PF algorithm and the proposed efficient method were executed to enable direct comparison of their results. The parameters for the PF approach are: 500 particles and a standard deviation for the initial condition of $\sigma_\omega = 10^{-6}$. The simulations were conducted using Python on a system equipped with an AMD Ryzen 5 4500U processor (2.375GHz) and 8 GB of RAM.

V. Results

The performance of the proposed approach has been evaluated in terms of the similarity of the JITP estimation compared to the baseline standard PF method. To enhance clarity and prevent overloading the graphs, the results are presented for a fixed value of $\alpha = 15\%$. However, a comprehensive evaluation of different values α is provided in the ablation study in Section V-A.

Fig. 3 presents the optimized prediction alongside the baseline PF result across a broad range of SoC values. The differences between the two trends are minimal, as highlighted in the zoomed-in view. The x-axis reports the initial state of the SoC x_0 at the beginning of each prediction. For example, a prediction started at 65% of the SoC yields an EoD JITP prediction of around $5000s$. In Fig. 4, the error analysis between the two trends is examined in greater detail. In particular, Fig. 5(a) shows the trend of absolute error between the optimized JITP and the standard PF JITP, while Fig. 5(b) presents the trend of error relative to the PF baseline value. Trends are only reported for three values of $T_{\max}$ (5s, 4s, and 2s) for clarity in the graph.

Given the high number of simulations performed, the trends have been smoothed with a first-degree Savitzky-Golay filter with a window of 9 samples [48]. Furthermore, the uncertainty band represents the standard deviation calculated with a rolling window of 80 samples. Overall, the relative error with respect to the baseline PF remains below 3% across the entire prediction horizon, validating the operating principle of the proposed

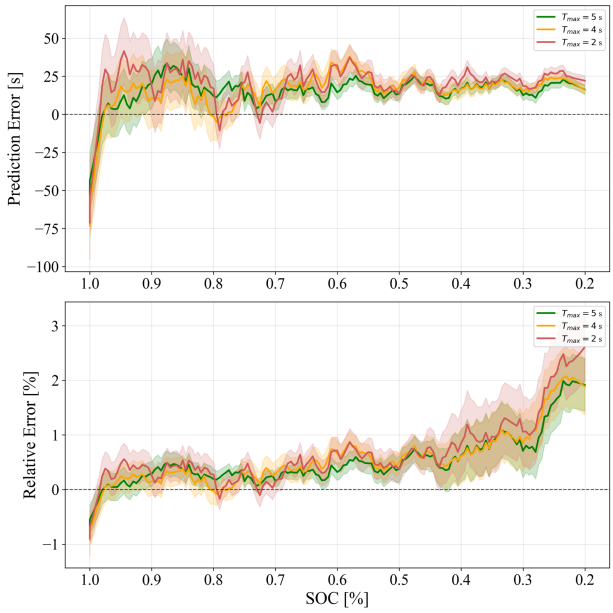


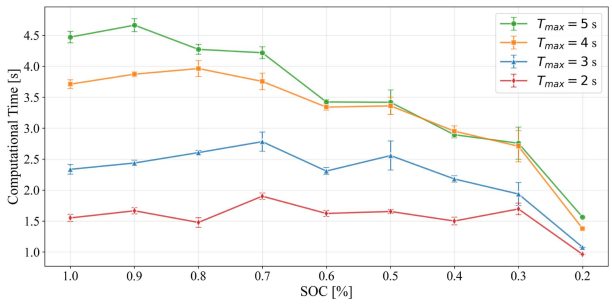
Fig. 4. Absolute and relative error trend for a series of predictions with three different values of $T_{\max}$.

TABLE III
Error Trend Metrics.

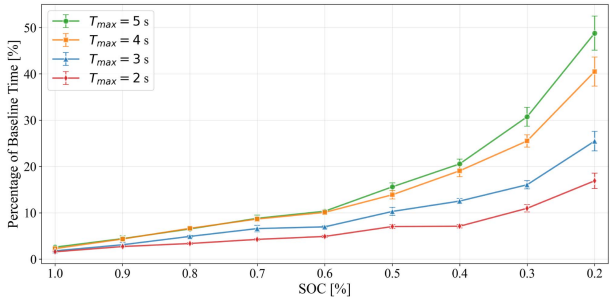
$Trend$	RMSE (s)	SNR
$T_{max} = 5\,s$	**26.91**	**473.24**
$T_{max} = 4\,s$	29.80	391.68
$T_{max} = 2\,s$	37.92	312.36

The bold values represent the lowest RMSE (s) and the highest SNR.

strategy. The fidelity of the methodology improves as the system approaches the EoD, which is an expected outcome given the two-step framework, which enhances the accuracy in the second part of the sub-sampling. In the example proposed before, the same prediction performed at 65% SoC leads to an error of around $12s$ (i.e., around 0.27%). The trends of absolute and relative errors are similar; however, relative error increases toward the end. This rise is due to the decrease in the JITP value provided by the PF standard method, which is used as the basis for calculating the relative error, as the ToF approaches. Consequently, the relative error percentage increases. Table III summarizes the RMSE of the error trend with respect to the zero line. The 5-s trend clearly represents the best compromise, yielding the lowest error and the highest Signal-to-Noise Ratio (SNR), which evidences the higher noise level in the 2-s prediction. Although imposing a stricter constraint on the maximum computational time ($T_{\max}$) results in a slight increase in error, the overall trend remains virtually unchanged across all three cases. Only minor deviations are observed, primarily within the SOC range of 1 to 0.6, even for $T_{\max} = 2$ s. This indicates that the marginal gain in accuracy obtained under more relaxed time constraints does not justify the additional computational



(a)



(b)

Fig. 5. Computational time and percentage of baseline time of the proposed prognostic scheme at different SoC initial conditions. For reference, the original PF strategy for a fully charged battery (SoC at 100%) requires approximately one minute; at the end of the cycle (SoC at 20%), this duration reduces to approximately 5 seconds.

cost, suggesting that near-optimal performance can be achieved under the explored conditions.

The computational times of the optimized method are reported in Fig. 5: the first graph shows the absolute computational time, while the second graph reports the percentage with reference to the baseline model. This is reported for four values of $T_{\max}$ ($5s$, $4s$, $3s$, and $2s$) and report the time to obtain the PDF, regardless of the α value. Overall, all the trends decrease as the SoC initial condition is closer to the EoD, given the lower number of simulations to perform. However, if due to the inherent math behind the PF methodology, the baseline method exhibits a nearly linear decline throughout the discharge cycle, the decrease for the optimized methodology is not linear. It should be noted that, as mentioned in Section III-B2, the requirement $T_{\max}$ is an upper bound, and, as a consequence, most runs reach the end of the simulation before $T_{\max}$. The error bar shows a slightly higher number than $T_{\max}$ due to the rolling window for the calculation of uncertainty.

Fig. 5(b) shows the percentage of time with respect to the baseline calculation, showing a saved time up to 95% . The percentage starts from 5% and then increases to 50% in the worst case as the battery approaches the EoD. In fact, near the EoD, the duration of the baseline PF algorithm also decreases, resulting in a reduced share of saved time. A further observation of the data presented in Figs. 4 and 5(b) reveals that at approximately 0.5% SoC, the computational time is reduced by more than 90% , while the associated absolute error remains significantly below 1% . This indicates an order-of-magnitude enhancement in computational efficiency while preserving the predefined accuracy threshold, as previously specified.

In order to further extend the analysis, beyond execution time and point-estimate accuracy, Fig. 6 compares the PDFs at the EoD obtained with the baseline algorithm (blue) and the proposed sub-sampling scheme (orange) for six representative scenarios. In all cases, the focus of the analysis must be placed on the left tail of the distribution, the region where the probability of premature EoD, as this domain is critical to robust and conservative operational decision-making. Furthermore, each subfigure in Fig. 6 reports the sampling frequency applied under the respective initial SoC and execution time constraints, providing a direct means to evaluate the influence of sampling rate on both the dispersion and shape of the predicted distributions. For Fig. 6(a) and (b), the sub-sampled PDF reproduces the baseline remarkably well when $T_{\max}$ is set to 5 s. Under the stricter limit of 2 s, the distribution broadens and exhibits a slightly lower peak than the baseline, resulting in nonzero failure probabilities a few seconds earlier than in the baseline and marginally shifting the mode to the right; however, Although some distortion in shape is observed, the resemblance between both distributions remains evident, and their supports largely overlap—indicating that the associated risk is not significantly underestimated. For Figs. 6(c) and (d), the $5s$ configuration aligns closely with the baseline, particularly in the critical left tail. The right tail shows a slower decay, resulting in a delayed onset of certain EoD; however, this has minimal practical significance, as operational measures are typically initiated well before reaching that region. For $T_{\max} = 2$ s, the overall shape is preserved and the support is essentially the same, although the left tail is lighter and the distributions present a slight shift to the right, which could imply a less conservative estimate, depending on the chosen threshold. For Fig. 6(e) and (f), both time constraints yield a satisfactory agreement, in particular the case $T_{\max} = 5$ s. For $T_{\max} = 2$ s, the same as happened in the previous scenarios, the overall shape and support are preserved; however, the rightward shift may lead to a slight underestimation of risk. This shift, approximately 100 s, is not critical for the practical purposes of this case study. Moreover, even under the 2 s computational budget, the distributions remain strongly similar, with their supports overlapping throughout the region of interest. Regarding the sampling frequency, it can be observed that F_2, the frequency used during the second stage of the prognostic, becomes coarser when the prognostic is performed at higher SOC values and under tighter execution time constraints, as expected. The same trend is observed for F_1 in the case of $T_{\max} = 2$s; however, the opposite occurs for $T_{\max} = 5$ s, although in this latter case the sampling step size remains quite similar across the three initial SOC conditions analyzed.

A. *Ablation Study*

In order to quantify and validate the performance of the XGBoost model for different conditions, an ablation study has been carried out for different values of α and for the whole

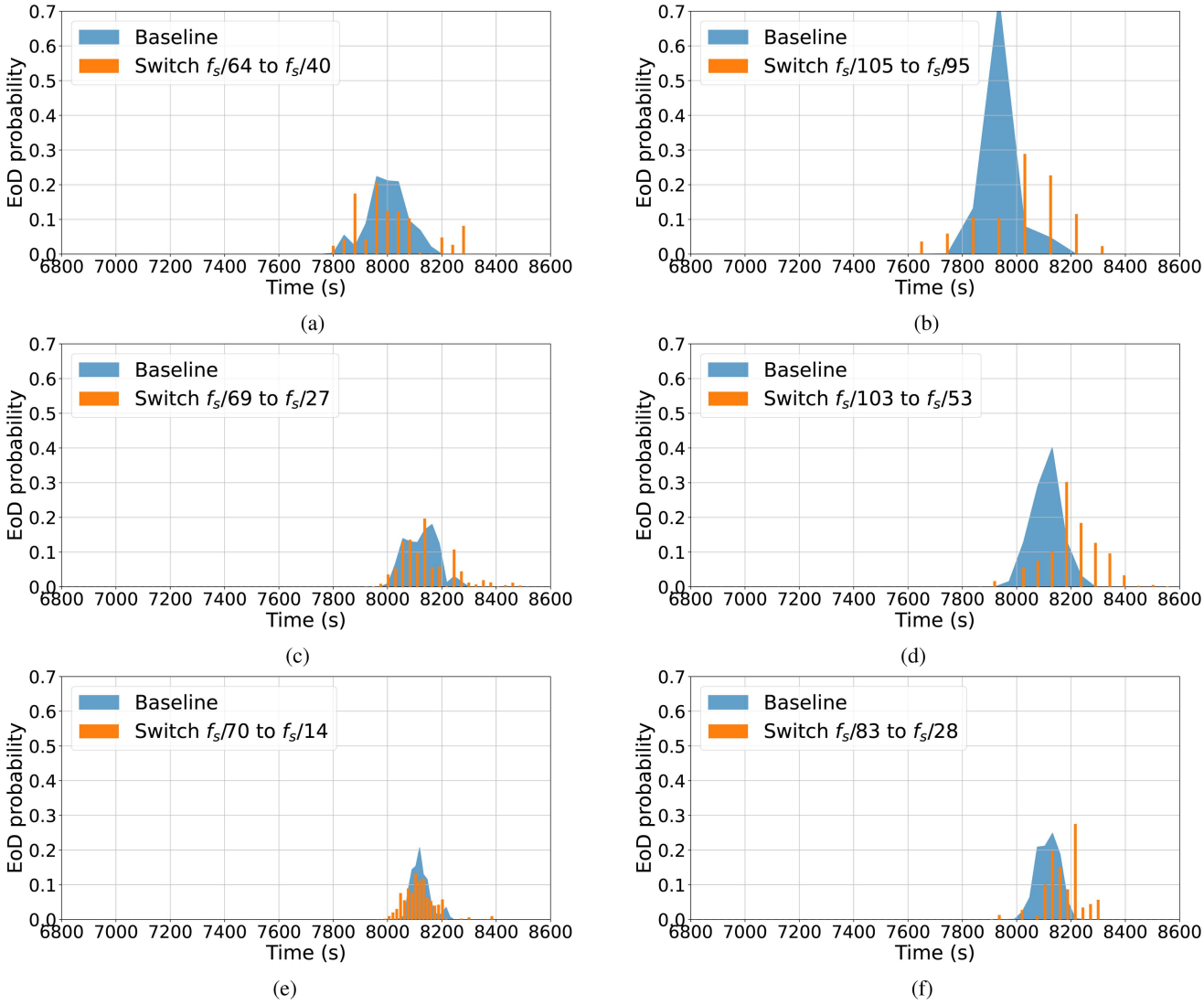


Fig. 6. PDF distributions at different SoC values and for different $T_{\max}$. Each PDF also indicates the two sampling frequencies used during the prognostic process. The second frequency, F_2, becomes coarser as the initial SoC condition increases or as the execution time constraint tightens.

range of $T_{\max}$. Several metrics (i.e., RMSE, MAE, and MAPE) are reported. Moreover, a Performance Coefficient (PC) has been employed to take into account both error and computational time in a single metric [49]. To calculate the PC, the $T_{\max}$ was considered for each set of simulations to account for the worst-case scenario in which the simulations run for the maximum available duration. Table IV shows the results obtained, integrating the partial error metrics introduced in Table III, which only included the relative error trend.

As expected, the error metrics improve when the time constraints are less restrictive. These results demonstrate the high accuracy of the approach, as the largest errors remain below one minute over a prediction horizon exceeding two hours of operation (approximately 8,600 seconds). This is further supported by the metric MAPE (see Table IV), which is smaller than 1% for all $T_{\max}$. More specifically, over the full set of settings, the sub-sampled PF delivers RMSE of 26–41s, MAE

TABLE IV
ERROR METRICS FOR OPTIMIZED XGBOOST MODELS.

T_{max}(s)	α	RMSE(s)	MAE(s)	MAPE	PC
5	5%	**25.94**	**20.71**	**0.26**%	**69.66**%
	10%	26.08	21.41	0.27%	68.53%
	15%	26.91	22.26	0.28%	68.44%
4	5%	29.51	23.85	**0.30**%	**72.39**%
	10%	**28.77**	**23.69**	**0.30**%	72.23%
	15%	29.80	24.74	0.31%	72.04%
3	5%	32.45	25.92	0.33%	77.23%
	10%	**31.92**	**25.85**	**0.32**%	76.89%
	15%	32.25	26.82	0.34%	**77.31**%
2	5%	41.20	31.52	0.40%	80.72%
	10%	**36.58**	**29.63**	**0.37**%	**82.35%**
	15%	37.92	31.41	0.39%	82.21%

The bold values indicate the lowest error (or highest PC) within each $T_{\max}$ group, whereas underlined values indicate the overall lowest error (or highest PC) across all settings.

of 21–32s, and MAPE of 0.26% –0.40%. If considered together with the relative error of 3% observed in Figure 4, these results show that the two-phase sub-sampling strategy maintains the JITP accuracy characteristics of the standard PF while enabling substantial computational savings throughout the remaining discharge cycle. If all error metrics highlight the superiority of the more relaxed time constraints, the PC, which considers both the RMSE and the time, shows that the best trade-off is achieved with a $T_{\max}$ of 2 seconds, the strictest execution time limit.

VI. Conclusion

PF-based prognostic implementation is associated with high computational costs related to the intrinsic underlying working principles, which require the propagation of particles to evaluate the future projection. This article introduces one of the few published proposals to reduce the computational burden while achieving a tradeoff between accuracy and computational effort. The solution implements a two-stage sub-sampling scheme that allows the user to set an upper bound on execution time; the scheme then selects the most appropriate sampling configuration based on current system conditions. The parameter selection scheme is powered by a set of three XGBoost models trained on a custom dataset generated offline via GA.

The framework has been tested on a battery discharge data set, yielding a satisfactory tradeoff between accuracy and efficiency, drastically reducing the computation time in comparison to the baseline, at the cost of just small errors in terms of JITP estimation. The results show that it is indeed possible to compute failure prognostics with the advantages of PF-based methodologies in real-time operation, paving the way for further adoption of this reliable PHM strategy.

In addition, in this study, we benchmark the framework against a standard PF implementation to cleanly attribute the observed computational savings to the proposed two-phase sub-sampling and PAP selection; as discussed in Section I-A, many efficiency-oriented PF variants are largely orthogonal and could in principle be wrapped by the same mechanism.

The balance observed between prediction accuracy and computational efficiency can be attributed to the design of the proposed two-step sub-sampling PF framework. The coarser sub-sampling frequency, F_1, effectively reduces the computational cost, whereas the finer sub-sampling frequency, F_2, helps preserve the accuracy in the estimation of the Time of Failure (ToF). Moreover, the offline learning stage determines the optimal switching point between F_1 and F_2 for each operational condition, ensuring an adaptive trade-off between performance and computational efficiency.

Several opportunities for further research directions stem from this work. Additional research directions may include exploring the feasibility of implementing more than one frequency switch, better evaluating the impact of the optimization scheme on other case studies, extending the methodology to systems with multivariate state vectors, and evaluating the impact of non-Gaussian noise. Furthermore, alternative approaches to the XBGoost suggestion model can be explored to enhance prediction accuracy and stability with strict time requirements. These approaches would broaden the applicability of this framework beyond the current case study of lithium-ion battery discharge.

References

[1] F. Calabrese, A. Regattieri, L. Botti, and F. G. Galizia, "Prognostic health management of production systems. new proposed approach and experimental evidences," *Procedia Manuf.*, vol. 39, pp. 260–269, 2019.

[2] M. Kordestani, M. Saif, M. E. Orchard, R. Razavi-Far, and K. Khorasani, "Failure prognosis and applications—A survey of recent literature," *IEEE Trans. Rel.*, vol. 70, no. 2, pp. 728–748, Jun. 2021.

[3] I. Shin et al., "A framework for prognostics and health management applications toward smart manufacturing systems," *Int. J. Precis. Eng. Manuf.-Green Technol.*, vol. 5, pp. 535–554, 2018.

[4] E. Balaban, S. B. Johnson, and M. J. Kochenderfer, "Unifying system health management and automated decision making," *J. Artif. Intell. Res.*, vol. 65, pp. 487–518, 2019.

[5] S. Fu and N. P. Avdelidis, "Prognostic and health management of critical aircraft systems and components: An overview," *Sensors*, vol. 23, 2023, Art. no. 8124.

[6] J. Lee, F. Wu, W. Zhao, M. Ghaffari, L. Liao, and D. Siegel, "Prognostics and health management design for rotary machinery systems—reviews, methodology and applications," *Mech. Syst. Signal Process.*, vol. 42, pp. 314–334, 2014.

[7] M. E. Orchard and G. J. Vachtsevanos, "A particle-filtering approach for on-line fault diagnosis and failure prognosis," *Trans. Inst. Meas. Control*, vol. 31, pp. 221–246, 2009.

[8] D. E. Acuña-Ureta and M. E. Orchard, "Near-instantaneous battery end-of-discharge prognosis via uncertain event likelihood functions," *ISA Trans.*, vol. 135, pp. 199–212, 2023.

[9] L. Dingeldein, "Simulation framework for real-time phm applications in a system-of-systems environment," *Aerospace*, vol. 10, no. 1, 2023, Art. no. 58.

[10] B. Wang, Y. Lei, N. Li, and N. Li, "A hybrid prognostics approach for estimating remaining useful life of rolling element bearings," *IEEE Trans. Rel.*, vol. 69, no. 1, pp. 401–412, Mar. 2020.

[11] W. Peng, Y. Chen, A. Xu, and Z.-S. Ye, "Collaborative online RUL prediction of multiple assets with analytically recursive bayesian inference," *IEEE Trans. Rel.*, vol. 73, no. 1, pp. 506–520, Mar. 2024.

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

[13] D. Kim et al., "Accelerated particle filter with GPU for real-time ballistic target tracking," *IEEE Access*, vol. 11, pp. 12139–12149, 2023 .

[14] W. Yan, B. Zhang, X. Wang, W. Dou, and J. Wang, "Lebesgue-sampling-based diagnosis and prognosis for lithium-ion batteries," *IEEE Trans. Ind. Electron.*, vol. 63, no. 3, pp. 1804–1812, Mar. 2016.

[15] F. Jaramillo, J. M. Gutiérrez, M. Orchard, M. Guarini, and R. Astroza, "A Bayesian approach for fatigue damage diagnosis and prognosis of wind turbine blades," *Mech. Syst. Signal Process.*, vol. 174, 2022, Art. no. 109067.

[16] K. Astrom and B. Bernhardsson, "Comparison of Riemann and Lebesgue sampling for first order stochastic systems," in *Proc. 41st IEEE Conf. Decis. Control*, 2002, vol. 2, pp. 2011–2016.

[17] H. Rozas, F. Jaramillo, A. Perez, D. Jimenez, M. E. Orchard, and K. Medjaher, "A method for the reduction of the computational cost associated with the implementation of particle-filter-based failure prognostic algorithms," *Mech. Syst. Signal Process.*, vol. 135, 2020, Art. no. 106421.

[18] H. Xiao, J. Coble, and J. W. Hines, "Auxiliary particle filter for prognostics and health management," *Int. J. Prognostics Health Manage.*, vol. 14, pp. 1–16, 2023.

[19] G. E. Gorospe Jr., M. J. Daigle, S. Sankararaman, C. S. Kulkarni, and E. Ng, "GPU accelerated prognostics," in *Proc. Annu. Conf. PHM Soc.*, vol. 9, pp. 1–7, 2017.

[20] Özcan Dülger, H. Oğuztüzün, and M. Demirekler, "Uphill resampling for particle filter and its implementation on graphics processing unit," *Parallel Comput.*, vol. 115, 2023, Art. no. 102994.

[21] J. A. Chesser, H. V. Nguyen, and D. C. Ranasinghe, "The megopolis resampler: Memory coalesced resampling on gpus," *Digit. Signal Process.*, vol. 120, 2022, Art. no. 103261.

[22] D. Harder, H. Shoushtari, and H. Sternberg, "Real-time map matching with a backtracking particle filter using geospatial analysis," *Sensors*, vol. 22, 2022, Art. no. 3289.

[23] L. Middleton, G. Deligiannidis, A. Doucet, and P. E. Jacob, "Unbiased Markov chain Monte Carlo for intractable target distributions," *Electron. J. Statist.*, vol. 14, no. 2, pp. 2842–289, 2020.
[24] C. Barletta, W. Garn, C. Turner, and S. Fallah, "Hybrid fleet capacitated vehicle routing problem with flexible monte–carlo tree search," *Int. J. Syst. Science: Operations Logistics*, vol. 10, 2023, Art. no. 2102265.
[25] J. Liang, Y. Shao, W. Lio, J. Liu, and R. Kang, "Uncertain particle filtering: A new real-time state estimation method for failure prognostics," *Mathematics*, vol. 13, Art. no. 702, 2025.
[26] H. Zhang, W. Chen, and Q. Miao, "Adaptive diagnosis and prognosis for lithium-ion batteries via Lebesgue time model with multiple hidden state variables," *Appl. Energy*, vol. 392, 2025, Art. no. 125986.
[27] J. E. Garcıá Bustos et al., "A novel data-driven framework for driving range prognostics in electric vehicles," *Eng. Appl. Artif. Intell.*, vol. 142, 2025, Art. no. 109925.
[28] M. Dhada, A. K. Parlikad, O. Steinert, and T. Lindgren, "Weibull recurrent neural networks for failure prognosis using histogram data," *Neural Comput. Appl.*, vol. 35, pp. 3011–3024, 2023.
[29] R. Zhu, Y. Chen, W. Peng, and Z.-S. Ye, "Bayesian deep-learning for RUL prediction: An active learning perspective," *Rel. Eng. System Saf.*, vol. 228, 2022, Art. no. 108758.
[30] C. Kwok, D. Fox, and M. Meilă, "Real-time particle filters," in *Proc. Adv. Neural Inform. Process. Syst.*, S. Becker, S. Thrun, and K. Obermayer, Eds., vol. 15. MIT Press, pp. 1081–1088, 2002.
[31] A. Saxena, S. Sankararaman, and K. Goebel, "Performance evaluation for fleet-based and unit-based prognostic methods," in *Proc. Euro. Conf. PHM Soc.*, pp. 1–12, 2014.
[32] A. J. Cannon, "Quantile regression neural networks: Implementation in R and application to precipitation downscaling," *Comput. Geosciences*, vol. 37, no. 9, pp. 1277–1284, 2011.
[33] Y. Lei, N. Li, S. Gontarz, J. Lin, S. Radkowski, and J. Dybala, "A model-based method for remaining useful life prediction of machinery," *IEEE Trans. Rel.*, vol. 65, no. 3, pp. 1314–1326, Sep. 2016.
[34] S. Arulampalam, N. Gordon, and B. Ristic, *Beyond the Kalman Filter: Particle Filters for Tracking Applications*. Norwood, MA, USA: Artech House, 2004.
[35] E. Paccha-Herrera et al., "A particle filter-based approach for real-time temperature estimation in a lithium-ion battery module during the cooling-down process," *J. Energy Storage*, vol. 94, 2024, Art. no. 112413.
[36] D. E. Acuña-Ureta, M. E. Orchard, and P. Wheeler, "Computation of time probability distributions for the occurrence of uncertain future events," *Mech.l Syst. Signal Process.*, vol. 150, 2021, Art. no. 107332.
[37] S. Katoch, S. S. Chauhan, and V. Kumar, "A review on genetic algorithm: Past, present, and future," *Multimedia Tools Appl.*, vol. 80, pp. 8091–8126, 2021.
[38] R. M. Solgi, "Geneticalgorithm: An easy implementation of genetic-algorithm (GA) in Python," GitHub Repository, accessed on Mar. 14, 2025, 2020, *version 1.0.1, available at GitHub repository*. [Online]. Available: https://github.com/rmsolgi/geneticalgorithm
[39] D. A. Pola et al., "Particle-filtering-based discharge time prognosis for lithium-ion batteries with a statistical characterization of use profiles," *IEEE Trans. Rel.*, vol. 64, no. 2, pp. 710–720, Jun. 2015.
[40] D. Acuña and M. Orchard, "Theoretically rigorous approach to failure prognosis," in *Proc. Annu. Conf. PHM Soc.*, vol. 10, pp. 1–14, 2018.
[41] D. E. Acuña and M. E. Orchard, "Particle-filtering-based failure prognosis via sigma-points: Application to lithium-ion battery state-of-charge monitoring," *Mech. Syst. Signal Process.*, vol. 85, pp. 827–848, 2017.
[42] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, ser. KDD '16. New York, NY, USA: Association for Computing Machinery, 2016, pp. 785–794.
[43] C. Diaz et al., "Particle-filtering-based prognostics for the state of maximum power available in lithium-ion batteries at electromobility applications," *IEEE Trans. Veh. Technol.*, vol. 69, no. 7, pp. 7187–7200, Jul. 2020.
[44] C. Burgos, D. Sáez, M. E. Orchard, and R. Cárdenas, "Fuzzy modelling for the state-of-charge estimation of lead-acid batteries," *J. Power Sources*, vol. 274, pp. 355–366, 2015.
[45] S. Redner, *A Guide to First-Passage Processes*. Cambridge, U.K.: Cambridge Univ. Press, 8 2001.
[46] F. Jaramillo, M. Valderrama, V. Quintero, A. Pérez, and M. Orchard, "Time-of-failure probability mass function computation using the first-passage-time method applied to particle filter-based prognostics," in *Proc. Annu. Conf. PHM Soc.*, vol. 12, pp. 1–11, 2020.
[47] D. Troncoso-Kurtovic, "A prognostic decision-making approach under uncertainty for an electric vehicle fleet routing problem," M.S. thesis, Univ. Chile, Univ. Chile, 2023. [Online]. Available: https://repositorio.uchile.cl/handle/2250/193059
[48] M. U. Bromba and H. Ziegler, "Application hints for Savitzky-Golay digital smoothing filters," *Anal. Chem.*, vol. 53, no. 11, pp. 1583–1586, 1981.
[49] L. Baldo, I. Querques, M. D. L. Dalla Vedova, and P. Maggiore, "A model-based prognostic framework for electromechanical actuators based on metaheuristic algorithms," *Aerospace*, vol. 10, no. 3, 2023, Art. no. 293.

Open Access funding provided by 'Politecnico di Torino 2025-2027 Deposit Account' within the CRUI CARE Agreement