Original Article

Machine learning approaches for improving condition-based maintenance of naval propulsion plants

Andrea Coraddu[1], Luca Oneto[1], Aessandro Ghio[1], Stefano Savio[1], Davide Anguita[2] and Massimo Figari[1]

Abstract

Availability, reliability and economic sustainability of naval propulsion plants are key elements to cope with because maintenance costs represent a large slice of total operational expenses. Depending on the adopted strategy, impact of maintenance on overall expenses can remarkably vary; for example, letting an asset running up until breakdown can lead to unaffordable costs. As a matter of fact, a desideratum is to progress maintenance technology of ship propulsion systems from breakdown or preventive maintenance up to more effective condition-based maintenance approaches. The central idea in condition-based maintenance is to monitor the propulsion equipment by exploiting heterogeneous sensors, enabling diagnosis and, most of all, prognosis of the propulsion system's components and of their potential future failures. The success of condition-based maintenance clearly hinges on the capability of developing effective predictive models; for this purpose, effective use of machine learning methods is proposed in this article. In particular, authors take into consideration an application of condition-based maintenance to gas turbines used for vessel propulsion, where the performance and advantages of exploiting machine learning methods in modeling the degradation of the propulsion plant over time are tested. Experiments, conducted on data generated from a sophisticated simulator of a gas turbine, mounted on a Frigate characterized by a COmbined Diesel eLectric And Gas propulsion plant type, will allow to show the effectiveness of the proposed machine learning approaches and to benchmark them in a realistic maritime application.

Introduction

According to the British Standard,[1] maintenance includes all actions necessary to retain a system or an item in, or restoring it to, a state in which it can perform its required functions. The most common way of inflecting such concepts in practice has been always deployed according to a "fix it when it breaks" approach.[2] However, this has been becoming an unaffordable and gold-brick methodology since data gathering from the field is ever cheaper and costs related to a breakdown may overcome the asset value.[3,4] Indeed, in the last decades, going-smart technology and cross-industry needs in maintenance, for example, ranging from manufacturing[5,6] to the transportation domain,[7,8] have engendered a pivotal change from a reactive to a proactive perspective,[9] trespassing the original focus on

repairing–replacing actions toward more sophisticated preventive and prescriptive activities. In particular, maintenance actions can be framed into a taxonomy, which includes three categories:[10] corrective, preventive and condition-based.

In corrective maintenance (CM), the equipment or asset is run down to its breaking down point, and maintenance activities are carried out afterward with the purpose of restoring the system at earliest. In this case,

[1]DITEN—University of Genova, Genova, Italy
[2]DIBRIS—University of Genova, Genova, Italy

Corresponding author:
Andrea Coraddu, DITEN—University of Genova, Via Opera Pia 11a, I-16145 Genova, Italy.
Email: andrea.coraddu@unige.it

maintenance activities are triggered by an unscheduled event (e.g. failure), and no a priori strategies can be deployed. Consequently, costs related to such approach are usually high; they comprise direct costs, for example, due to potentially concatenated failures of other parts of the asset, and indirect costs, for example, related to potential losses in (environmental, worker, etc.) safety and integrity, and asset unavailability.

Preventive maintenance (PM), instead, is carried out before breakdowns in order to avoid them and minimize the possibility of potential issues related to failures. Several variations exist, such as adjustments, replacements, renewals and inspections, which take place subject to a predetermined planning and schedule; this allows to establish time-slots of unavailability of an asset (or part of it), opposite to the unpredictability, which characterize random failure patterns in CM. With particular reference to systematic PM, parts are replaced independently of their actual status, as a safety-level lifetime is established for them; in such a way, the probability of failures for a system decreases, as popularly proved through adverting to the so-called bathtub curve.[11]

Condition-based maintenance (CBM), instead, refers to triggering maintenance activities as they are necessitated by the condition of the target system.[2] This approach enables determining the conditions of in-service assets to predict potential degradations and to plan, consequently, when maintenance activities will be needed and should be performed to minimize disruptions. In other words, CBM switch maintenance view from pure diagnosis to high-valued prognosis of faults. CBM thus entails the diagnosis of the target system and timely, accurate and valuable identification of potential issues, by exploiting effective approaches from the predictive analytics domain.[2,3,12]

While CM is usually avoided due to the high costs characterizing this approach, CBM is gaining popularity because of the competitive advantages over PM, related to the extreme conservativeness of preventive methods in several domains (leading, once again, to often unjustified remarkable costs). As a matter of fact, assets are typically featured by complex interrelations among the parts of which they consist; such correlations are not straightforward to grasp and identify, thus leading to the unfeasibility of a simple approach like the above-cited bathtub model.[2,11] CBM, instead, proved to be effective to maximize availability, efficiency, sustainability and safety of assets, enabling the upgrading of maintenance role from being seen as evil up to a way for creating added value.[13,14]

The CBM approach to maintenance is very generic, depicting a horizontal view to a crosscut problem in different heterogeneous domains. When a more vertical view is taken into consideration, focusing on the maritime domain, repair maintenance expenses for conventional ships can amount up to about 20% of total operability costs, including manning expenses, insurance and administration costs.[15] For naval applications, maintenance optimization is a key task, focused to reduce operations costs while getting the optimal availability of the ship for the intended service; such optimization is the result of a trade-off between excessive maintenance and machinery downtime, where CBM helps opening the door toward best balancing costs and availability. In other words, CBM enables a just-in-time deployment of ship maintenance, by allowing to plan and execute maintenance activities only when needed.

The CBM approach requires that a reliable and effective diagnostic policy on one (or more) naval asset(s) is implemented, in order to enable further refinement toward identifying potential failures in advance. Gas turbines (GTs), used for naval propulsion, represent a key example of how gathering data from components can be a driver for optimizing maintenance strategies.[16,17] Diagnostic methods for GTs[18,19] allow to obtain information on the health status of the components without the need to inspect the machine, thus causing an interruption of operations with consequent disruptions of services. Diagnosis represents the starting milestone to describe the status of the turbine, in order to evolve such systems toward prognostic models, able to predict potential future issues with a reasonable advance. In this sense, the use of these approaches assumes a central importance when applied to propulsion systems in naval vessels; the implementation of CBM strategies allows to improve the reliability and efficiency of the asset and to provide support to maintenance operations in a non-invasive way and by limiting interventions, which are carried out in accordance with a when-needed fashion.

The cornerstone of applying CBM consists in the adoption of a proactive approach to maintenance in place of more conventional reactive interventions; to implement such key change, predictive modeling assumes a central importance to allow automatic triggering of alarms without the need of designing complex, often unknown and, in some sense, unknowable models. For this purpose, the use of robust machine learning (ML) models[20,21] is here proposed and tested. These models show appealing modeling and predictive capabilities even in very complex and heterogeneous problem domains. In particular, the authors focus on regularized least square (RLS)[21] and support vector machine (SVM),[20] that is, two state-of-the-art methods whose effectiveness in coping with real-world problems is also confirmed by the inclusion in commercial software packages (e.g. it is the case of SVM, included in the Oracle Data Mining suite[22]). RLS and SVM models are induced only from data collected from the field on the assets that need to be modeled, that is, the predictive model does not require any a priori knowledge on the turbines, their architecture/configuration or on the system overall.

In this article, the performance and potentialities of RLS and SVM models are benchmarked by training them on data to forecast the performance decay of

GTs, installed on a naval vessel. The aim of this work is to pave the way toward designing a prognostic CBM methodology to naval propulsion plants equipped with GTs. In particular, the propulsive performance degradation of a naval vessel powered by a GT has been simulated, between two dry docks, in order to make available and to collect a dataset of experiments. Such data are then exploited to derive a predictive model to automatically detect the system decay state, targeting effective CBM in the considered domain.

For these purposes, this article is organized as follows. Section "Ship propulsion models" introduces the model and the simulator of the naval vessel, considered for presenting the ML approach to CBM in the subsequent analysis. Then, section "ML framework" focuses on the ML framework, by introducing the main approaches that will be exploited for this work's targets; actionable algorithms will be proposed, which can be implemented for forecasting purposes in CBM applications. With reference to the characteristics of the simulator of the naval vessel and of the needs, emerging from the ML framework, a dataset has been created, according to the setup described in section "Dataset creation"; this dataset will be released for the use to the research community on the widespread well-known dataset repository of the University of California in Irvine (UCI). Section "Experimental results and discussion" subsequently presents some experimental results, derived by applying the proposed ML approaches to the dataset, while section "Conclusion" will finally draws the conclusions.

Ship propulsion models

Introduction

The authors contextualize the proposal for CBM with reference to a Frigate naval vessel, characterized by a COmbined Diesel eLectric And Gas (CODLAG) propulsion plant type, as depicted in Figure 1. In this particular configuration, the GT mechanically drives the two controllable pitch propellers (CPPs) via a cross-connected gearbox. The propulsion plant numerical model is derived from authors' previous works.[23,24] The simulator is developed in Simulink® software environment, a MATLAB® toolbox, structured in a modular arrangement such as each module represents a single propulsion component. In particular, the GT is characterized by a two-shaft arrangement with gas generator (GG), consisting of compressor and high-pressure (HP) turbine for the exhaust gas production, driving the low-pressure (LP) power turbine as reported in Figure 2. Detailed information for the GT performance simulation can be found in Benvenuto and Campora.[25,26] The GT compressor behavior is modeled by means of steady-state performance maps,[27] representing rotational speed and isentropic efficiency η_c as functions of the pressure ratio β_C and the airflow rate M_c.
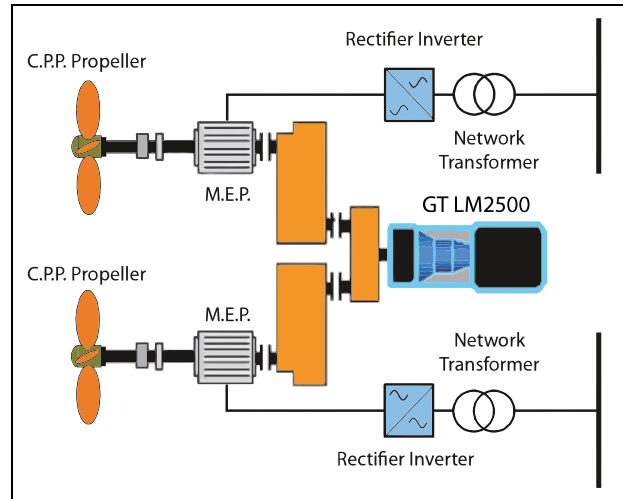

Figure 1. CODLAG propulsion system.
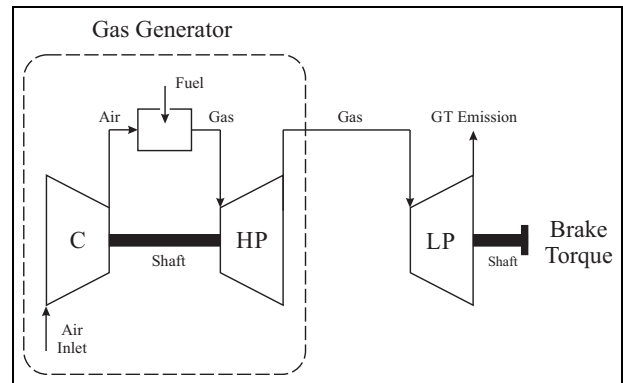CPP: controllable pitch propeller; GT: gas turbine.

Figure 2. Gas turbine scheme.
HP: high pressure; LP: low pressure; GT: gas turbine.

Model description

A generic ship propulsion plant may be considered made up by three main components: the engine, the transmission gear and the propulsor as depicted in Figure 3. It is possible to describe the dynamic performance of the propulsion plant[28] by means of the simultaneous solution of the three non-linear differential equations in the time domain

$$
\begin{cases}
M \cdot \dfrac{dv(t)}{dt} = p \cdot T_p(t) - R_t(t) \\
2\pi \cdot J_p \cdot \dfrac{dn(t)}{dt} = \text{i.e} \cdot \quad Q_e(t) - Q_p(t) \\
\dfrac{dm_f(t)}{dt} = f\left(n, \dfrac{dn(t)}{dt}\right)
\end{cases}
\tag{1}
$$

where T_p is the propulsor thrust, p is the number of propulsors, R_t is the total ship resistance, M is the ship mass, Q_e is the engine torque, i is the gear ratio, e is the number of engines per shaft, Q_p is the propulsor torque and J_p is the polar moment of the rotating masses.
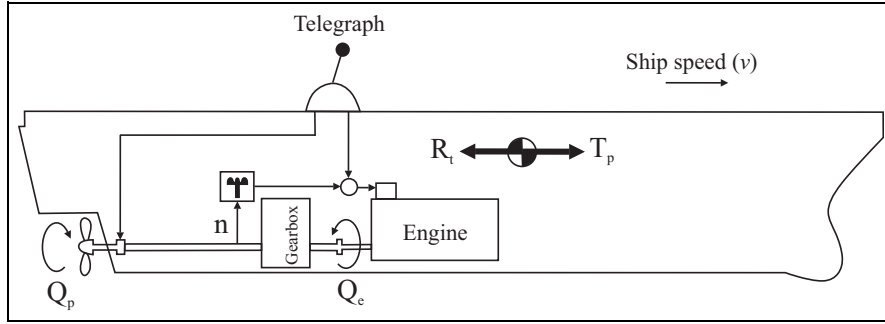
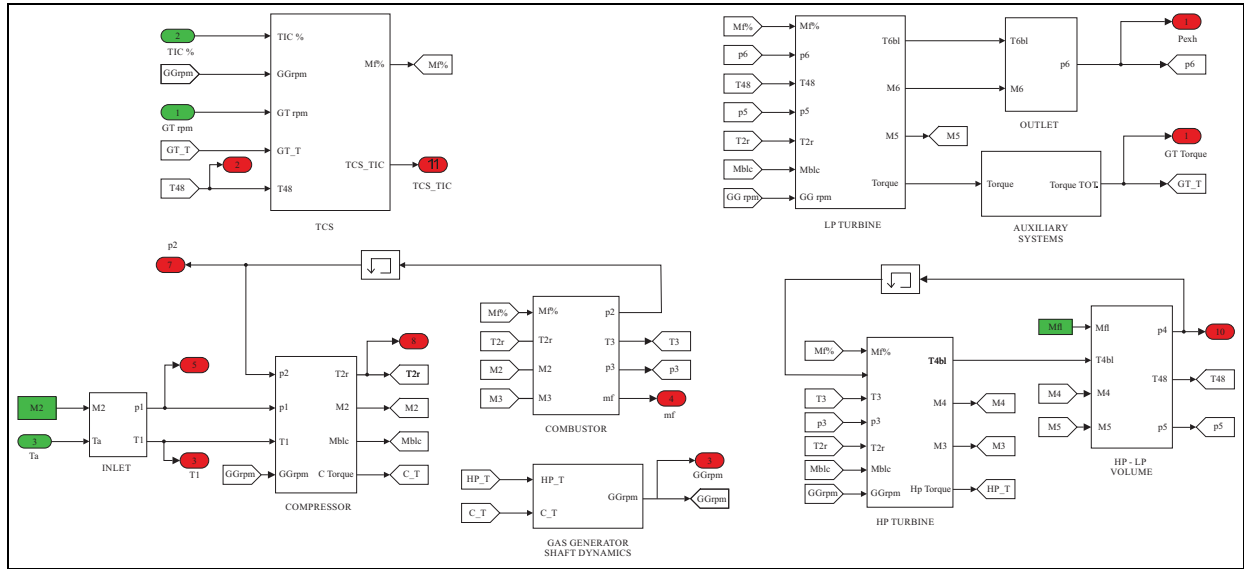**Figure 3.** One-degree-of-freedom ship dynamics.

Figure 4. LM 2500 gas turbine model scheme.
TIC: turbine injection control; GT: gas turbine; LP: low pressure; HP: high pressure.

The first equation represents the longitudinal motion of the ship, the second equation represents the rotational motion of the shaft line and finally the third equation represents the control system behavior. Forces and torques in equation (1) can be represented in different ways depending on the type of the propulsion system, the type of analysis and the expected degree of accuracy. Among the possible available approaches, in order to reduce the calculation time, the inter-component volumes method has been adopted, preferring it to more sophisticated models for the GT dynamic simulation. For the simulation of the General Electric LM 2500 GT, this procedure has been applied,[25] and in order to validate the model, steady-state and transient results obtained by the simulator have been compared with experimental data provided by the GT manufacturer.

The GT mathematical model is structured in a modular arrangement, as shown in Figure 4 where the Simulink top-level scheme of the considered GT is reported. In Figure 4, the input and output variables of the different plant components such as compressor, HP turbine, power turbine and combustor are shown. It is worth mentioning that each module, representing a specific engine component, has been modeled by means of steady-state performance maps, time-dependent momentum, energy and mass equations and non-linear algebraic equations.

The main differential equations to be solved in the time domain refer to both dynamic and thermodynamic engine parameters. In particular, the working fluid processes are governed by the continuity equation

$$\frac{d(\rho \cdot V) \cdot \rho}{dt} = M_i - M_o \qquad (2)$$

and by the energy equation

$$\frac{dm_f(t)}{dt} = f\left(n, \frac{dn(t)}{dt}\right) \qquad (3)$$

where M_i and M_o are the inlet and outlet mass flow rates, respectively; V is the GT component volume; h and u are enthalpy and internal energy, respectively; while P and Q' represent power and heat flow, respectively.
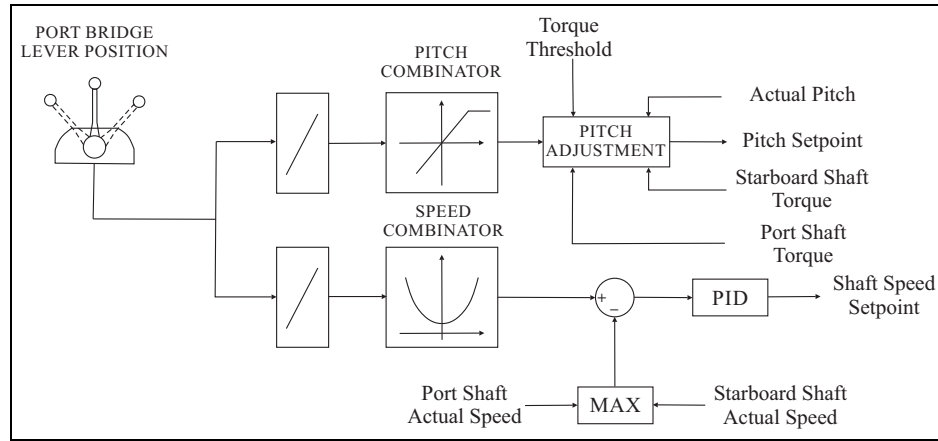
Figure 5. Shaft speed control scheme.
PID: proportional–integral–derivative.

The dynamics of the rotating shafts is determined by the dynamic momentum equation

$$\frac{d\omega}{dt} = \frac{1}{J \cdot \omega}(P_m - P_{br}) \tag{4}$$

P_m and P_{br} being the motor and brake powers, respectively, and ω is the angular velocity. The propulsion system is managed by a typical combinator law whose general layout is presented in Figure 5. The control system has been designed to provide a linear relationship between lever position and ship speed. In this application, the ship propulsion control system dynamics has been represented by a proportional–integral (PI) controller

$$\text{TIC} = K_p \cdot (n_{sp} - n) + K_i \cdot \int (n_{sp} - n)d\tau \tag{5}$$

where turbine injection control (TIC) is the control signal (i.e. percent of fuel flow) for the propulsion engine; n and n_{sp} are the actual and the set point propeller shaft speed, respectively; while K_p and K_i represent the PI control loop constants. This is only a rough approximation, but can be considered satisfactory for the most important ship maneuvers. Finally, it is worth noting that model validation for a ferry, a naval corvette and an aircraft carrier can be found in various works.[29–31]

Asset degradation model

The model described in section "Ship propulsion models" has been updated to take into account the performance decay of the several propulsive components, such as GT, propellers and hull. As reported in Bruzzone,[32] the time domain performance decay has been modeled by means of suitable coefficients inserted in the GT compressor and HP and LP modules of the simulator. In such a way, it has been possible to quantify the variation in performance of each component and consequently of the entire propulsion system.

As far as GT compressor performance decay is concerned, this phenomenon is caused by fouling. Fouling of the compressor of a GT is indicated as a major cause of loss of performance of the machine.[33,34] It is caused by air impurities that together with the exhaust gases and oil vapors produce a layer that joins to cover the blades. The main consequences are an increase in the specific fuel consumption of the GT and an increase in the temperature of the exhaust gas. The effect of fouling is simulated by reducing, over service hours, the numerical values of the airflow rate M_c and of the isentropic efficiency η_c.

Turbines' performance reduction after several running hours is due to their blades' erosion and fouling, so in analogy to the performance decay modeling of the GT compressor, a gas flow rate reduction factor kM_T[32] has been generated on the basis of literature data, regarding heavy-duty and aeronautical turbines as reported in Aker and Saravanamuttoo.[35]

These particular performance decay functions are empirically derived and are only a function of time. The real degradation behavior of the physical asset could be described by means of a more accurate function able to describe both time dependency and the real operation profile. In order to overcome this issue, authors put forward an analysis able to take into account each possible combination of GT compressor and HP and LP turbine conditions.

Under this assumption, authors sample the range of decay of GT compressor and turbines with a uniform grid of precision 0.001 so to have a good granularity of representation. In particular, for the GT compressor decay state, the following state discretization has been adopted

$$kM_c \in [0.95, 1] \tag{6}$$

$$kM_c^i = 1 - i \cdot 0.001, \quad i \in \{0, 1, \ldots, 51\} \tag{7}$$

Similarly, for the GTs

$$kM_t \in [0.975, 1] \tag{8}$$

$$kM_t^i = 1 - i \cdot 0.001, \quad i \in \{0, 1, \ldots, 26\} \tag{9}$$

The hull and propellers' degradation here is not taken into account. An improvement of the model is under development for introducing this phenomenon in the propulsion plant simulator.

For this CBM application, let us consider the propulsion system identified by the following parameters:

1. *Ship speed v*: this parameters allows to take into account different GT load conditions. The authors choose to consider different probability density functions (PDFs) for the speed distribution as there are three variants for this kind of naval vessel: anti-submarine warfare (ASW) variant, general-purpose (GP) variant and anti-aircraft warfare (AAW) variant. In particular, three different PDFs have been taken into account: uniform PDF for the GP variant and bimodal PDF for the AAW and ASW variants. A further analysis has been performed by means of a unimodal distribution for the cruising speed.
2. *GT compressor degradation coefficient kM_c*: this parameter describes the reduction, over service hours, of the numerical values of the airflow rate M_c and of the isentropic efficiency η_c.
3. *GT degradation coefficient kM_t*: this parameter describes the gas flow rate reduction factor over service hours.

In Table 1, the main simulation outputs are reported. These subsets of model's outputs are the same quantities that the automation system installed onboard can acquire and store.

As far as numerical model is concerned, it is worth noting that the ship dynamics part of the model has been validated and fine tuned comparing the simulation results with sea trial campaign data, measured during the full scale test campaign of the vessel, as reported in Martelli.[36]

Table 1. Main simulation output.

Variable	Symbol	Units
Lever position	lp	(·)
Ship speed	v	knot
Gas turbine shaft torque	GT_T	kN m
Gas turbine rate of revolutions	GT_n	r/min
Gas generator rate of revolutions	GG_n	r/min
Starboard propeller torque	T_s	kN
Port propeller torque	T_p	kN
HP turbine exit temperature	T_{48}	°C
GT compressor inlet air temperature	T_1	°C
GT compressor outlet air temperature	T_2	°C
HP turbine exit pressure	P_{48}	bar
GT compressor inlet air pressure	P_1	bar
GT compressor outlet air pressure	P_2	bar
GT exhaust gas pressure	P_{exh}	bar
Turbine injection control	TIC	%
Fuel flow	m_f	kg/s

HP: high-pressure; GT: gas turbine.

ML framework

CBM competitive advantages over CM and PM mostly deal with predictive capabilities in detecting potential failures with a reasonable advance, so to proactively schedule proper interventions only when needed and, in any case, before breakdown. ML approaches play a central role in extracting non-trivial information from amounts of raw data, collected from sensors and heterogeneous inputs and concerning the monitored asset. The learning process for ML approaches usually consists of two phases:

1. During the *training phase*, a set of data is used to induce a model that best fits them, according to some criteria.
2. Then, the trained model is set in motion into the real application during the *feedforward phase*.

In this article, authors focus on the problem of identifying the decay trend in GTs' performance, when GTs are used for naval propulsion. In this sense, a *regression* problem must be solved, where the decay trend over time must be estimated based on the data collected from GTs and, more broadly speaking, from the vessel. In particular, in the conventional regression framework in ML,[20,37,38] a set of training data $X = \{(\boldsymbol{x}_i, y_i)\}$, $i = 1, \ldots, n$, where in general $\boldsymbol{x}_i = \{x_i^{(k)}\} \in \mathbb{R}^d$ ($k = 1, \ldots, d$) and $y_i \in \mathbb{R}$, is considered. As the authors are targeting a regression problem, the purpose is to find the best approximating function $h(\boldsymbol{x})$, where $h : \mathbb{R}^d \to \mathbb{R}$, which should be close (in some sense) to the unknown target function $f(\boldsymbol{x})$.

During the training phase, the quality of the learned regressor $h(\boldsymbol{x})$ is usually measured according to a loss function $\ell(h(\boldsymbol{x}), y)$,[39] which calculates the discrepancy between the true output y and the estimated one. The empirical error then computes the average discrepancy per sample, reported by a model on the training patterns

$$\hat{L}_n(h) = \frac{1}{n} \sum_{i=1}^{n} \ell(h(\boldsymbol{x}_i), y_i) \tag{10}$$

A simple criterion for selecting the final model during the training phase could then consist in simply choosing the approximating function that minimizes the empirical error $\hat{L}_n(h)$; this approach is known as empirical risk minimization (ERM).[40] However, ERM is usually avoided in ML as it leads to severe overfitting of the model on the training dataset; as a matter of fact, in this case, the training process chooses a model, complicated enough to perfectly describe all the training samples (including noise, which afflicts them). In other words, ERM implies memorization of data rather than learning from them.

A more effective approach is represented by minimizing a cost function where the trade-off between accuracy on the training data and a measure of the

complexity of the selected approximating function is implemented[41,42] (for the sake of notational simplicity, $h = h(\boldsymbol{x})$ is identified)

$$h^* : \quad \min_h \quad \hat{L}_n(h) + \lambda \mathcal{C}(h) \tag{11}$$

In other words, the best approximating function h^* is chosen as the one that is complicated enough to learn from data without overfitting them. In particular, $\mathcal{C}(\cdot)$ is a complexity measure; depending on the exploited ML approach, different measures are realized. (Note that according to the no free lunch theorem,[43] no optimal choice exists, as opting for the best procedure is a problem-dependent task.) Instead, λ is a hyperparameter that must be aprioristically set and is not obtained as an output of the optimization procedure; it regulates the trade-off between the overfitting tendency, related to the minimization of the empirical error, and the underfitting tendency, related to the minimization of $\mathcal{C}(\cdot)$. The optimal value for λ is problem-dependent, and tuning this hyperparameter is a non-trivial task; the authors will devote a subsection in the following for this purpose.

Note that when monitoring the decay of complex environments and systems like in CBM applications, the authors could have to deal with multiple output (MO) problems instead of the single output (SO) ones, considered so far; in the MO framework, the objective value of the decay is an array $\boldsymbol{y}_i \in \mathbb{R}^m$, with $m > 1$. Although several approaches, such as the ones based on multi-task learning[44] or transfer learning,[45] can be applied to training models in the MO framework, the most effective solution often consists in decomposing the MO dataset into m SO problems.[46,47] In addition to allow simplifying the presentation of the ML method, this approach enables grasping the phenomena, characterizing the decay of components for CBM purposes, whenever the considered outputs are not very strongly correlated one to each other; as this hypothesis sounds reasonable in the considered application, the authors opt for following this path.

RLSs

In the next sections, the authors present two effective approaches, which rely on the framework depicted above. The first approach exploited to deploy predictive capabilities for CBM purposes is RLS.[21,48,49] In RLS, approximation functions are defined as

$$h(\boldsymbol{x}) = \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}) \tag{12}$$

where a non-linear mapping $\boldsymbol{\phi} : \mathbb{R}^d \to \mathbb{R}^D$, $D \gg d$, is applied in such a way non-linearity is pursued while still coping with linear models.

For RLS, problem (11) is configured as follows. The complexity of the approximation function is measured as

$$\mathcal{C}(h) = \|\boldsymbol{w}\|_2^2 \tag{13}$$

that is, the Euclidean norm of the set of weights describing the regressor, which is a quite standard complexity measure in ML.[20] Regarding the loss function, the mean squared error (MSE) loss is adopted

$$\hat{L}_n(h) = \frac{1}{n} \sum_{i=1}^{n} \ell(h(\boldsymbol{x}_i), y_i) = \frac{1}{n} \sum_{i=1}^{n} [h(\boldsymbol{x}_i) - y_i]^2 \tag{14}$$

Consequently, problem (13) is reformulated as

$$\boldsymbol{w}^* : \min_{\boldsymbol{w}} \quad \frac{1}{n} \sum_{i=1}^{n} [\boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}) - y_i]^2 + \lambda \|\boldsymbol{w}\|_2^2 \tag{15}$$

By exploiting the representer theorem,[50] the solution h^* of RLS problem (15) can be expressed as a linear combination of the samples projected in the space defined by $\boldsymbol{\phi}$

$$h^*(\boldsymbol{x}) = \sum_{i=1}^{n} \alpha_i \boldsymbol{\phi}(\boldsymbol{x}_i)^T \boldsymbol{\phi}(\boldsymbol{x}) \tag{16}$$

It is worth underlining that according to the kernel trick,[37,51] it is possible to reformulate $h^*(\boldsymbol{x})$ without an explicit knowledge of $\boldsymbol{\phi}$ by using a proper kernel function $K(\boldsymbol{x}_i, \boldsymbol{x}) = \boldsymbol{\phi}(\boldsymbol{x}_i)^T \boldsymbol{\phi}(\boldsymbol{x})$

$$h^*(\boldsymbol{x}) = \sum_{i=1}^{n} \alpha_i K(\boldsymbol{x}_i, \boldsymbol{x}) \tag{17}$$

Several kernel functions can be retrieved in literature,[37,51] but the Gaussian kernel is often used as it enables learning every possible function[52]

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = e^{-\gamma \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2} \tag{18}$$

where γ is a hyperparameter, which must be aprioristically tuned (analogously to λ). In particular, for small values of γ, simpler functions are realized (note that for $\gamma \to 0$, the authors derive a linear regressor); instead, if large values of γ are employed, more complicated functions can be realized.

As a final issue, RLS problem (15) can be reformulated by exploiting kernels

$$\boldsymbol{\alpha}^* : \min_{\boldsymbol{\alpha}} \quad \frac{1}{n} \sum_{i=1}^{n} \left[\sum_{j=1}^{n} \alpha_j K(\boldsymbol{x}_j, \boldsymbol{x}_i) - y_i \right]^2$$
$$+ \lambda \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j K(\boldsymbol{x}_j, \boldsymbol{x}_i) \tag{19}$$

By defining $\boldsymbol{y} = [y_1, \ldots, y_n]^T$, $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_n]^T$, the matrix K such that $K_{i,j} = K_{ji} = K(\boldsymbol{x}_j, \boldsymbol{x}_i)$, and the identity matrix $I \in \mathbb{R}^{n \times n}$, a matrix-based formulation of Problem (19) can be obtained

$$\boldsymbol{\alpha}^* : \min_{\boldsymbol{\alpha}} \frac{1}{n} \|K\boldsymbol{\alpha} - \boldsymbol{y}\|_2^2 + \lambda \boldsymbol{\alpha}^T K \boldsymbol{\alpha} \tag{20}$$

By setting equal to 0 the derivative with respect to $\boldsymbol{\alpha}$, $\boldsymbol{\alpha}$ can be found as follows

$$(K + n\lambda I)\boldsymbol{\alpha}^* = \boldsymbol{y} \tag{21}$$

which is a linear system for which effective solvers have been developed throughout the years, allowing to cope with even very large sets of training data.[53]

Support vector regression

As an effective alternative to RLS, the SVM models for regression (briefly support vector regression (SVR))[38] can be used. The main idea behind SVR consists in searching for the approximating function. (Note that the originally proposed function for SVR is $h(x) = w^T \phi(x) + b$, where $b \in \mathbb{R}$ is a bias term. However, the introduction of the bias would unnecessarily complicate the analysis of this case; in fact, it is possible to prove that the theoretical framework of SVM is maintained also if $b = 0$ as far as a positive definite kernel, like the Gaussian one of equation (18), is used.[54]) $h(x) = w^T \phi(x)$ which, in principle, should not be far from the target function $f(x)$ more than a user-defined threshold ε, that is, $|f(x) - h(x)| < \varepsilon$. Consequently, the epsilon-insensitive loss function is adopted

$$\ell(h(x), y) = |h(x) - y|_\varepsilon \tag{22}$$

where $|\cdot|_\varepsilon = \max(0, |\cdot| - \varepsilon)$. On the other hand, the same complexity measure, exploited by RLS, is used by SVR.

As a consequence, problem (11) can be reformulated as follows

$$w^* : \min_{w, \hat{\xi}, \breve{\xi}} \quad \frac{1}{2}\|w\|_2^2 + C \sum_{i=1}^n \left(\hat{\xi}_i + \breve{\xi}_i\right) \tag{23}$$

$$w^T \phi(x_i) - y_i \leqslant \varepsilon + \hat{\xi}_i, \quad \forall i \in \{1, \ldots, n\} \tag{24}$$

$$y_i - w^T \phi(x_i) \leqslant \varepsilon + \breve{\xi}_i, \quad \forall i \in \{1, \ldots, n\} \tag{25}$$

$$\hat{\xi}_i, \breve{\xi}_i \geqslant 0, \quad \forall i \in \{1, \ldots, n\}$$

where $C = 1/\lambda n$ is conventionally adopted as hyperparameter in place of λ. The two constraints of equations (24) and (25) are added to force learning of approximation functions which are close to the target one; note, in fact, that $\hat{\xi}_i, \breve{\xi}_i$ are slack variables.

Note that problem (23) is convex (20). Nevertheless, for computational reasons, the dual formulation of the SVR problem above is usually adopted, which can be derived by introducing the Lagrange multipliers $\hat{\alpha}, \breve{\alpha}, \hat{\eta}, \breve{\eta}$[55]

$$\mathcal{L}\left(w, \hat{\xi}, \breve{\xi}, \hat{\alpha}, \breve{\alpha}, \hat{\eta}, \breve{\eta}\right) = \frac{1}{2}\|w\|_2^2 + C \sum_{i=1}^n \left(\hat{\xi}_i + \breve{\xi}_i\right)$$
$$+ \sum_{i=1}^n \hat{\alpha}_i \left[w^T \phi(x_i) - y_i - \varepsilon - \hat{\xi}_i\right]$$
$$+ \sum_{i=1}^n \breve{\alpha}_i \left[y_i - w^T \phi(x_i) - \varepsilon - \breve{\xi}_i\right]$$
$$- \sum_{i=1}^n \hat{\eta}_i \hat{\xi}_i - \sum_{i=1}^n \breve{\eta}_i \breve{\xi}_i \tag{26}$$

for which the following Karush–Kuhn–Tucker (KKT) conditions hold

$$\frac{\partial \mathcal{L}}{\partial w} = 0 \rightarrow w = \sum_{i=1}^n \left(\breve{\alpha}_i - \hat{\alpha}_i\right) \phi(x_i) \tag{27}$$

$$\frac{\partial \mathcal{L}}{\partial \hat{\xi}_i} = 0 \rightarrow C - \hat{\alpha}_i - \hat{\eta}_i = 0 \rightarrow \hat{\alpha}_i \leqslant C, \quad \forall i \in \{1, \ldots, n\} \tag{28}$$

$$\frac{\partial \mathcal{L}}{\partial \breve{\xi}_i} = 0 \rightarrow C - \breve{\alpha}_i - \breve{\eta}_i = 0 \rightarrow \hat{\alpha}_i \leqslant C, \quad \forall i \in \{1, \ldots, n\} \tag{29}$$

$$w^T \phi(x_i) - y_i \leqslant \varepsilon - \hat{\xi}_i, \quad \forall i \in \{1, \ldots, n\} \tag{30}$$

$$y_i - w^T \phi(x_i) \leqslant \varepsilon - \breve{\xi}_i, \quad \forall i \in \{1, \ldots, n\} \tag{31}$$

$$\hat{\xi}_i, \breve{\xi}_i, \hat{\alpha}_i, \breve{\alpha}_i, \hat{\xi}_i, \breve{\xi}_i \geqslant 0, \quad \forall i \in \{1, \ldots, n\} \tag{32}$$

$$\hat{\alpha}_i \left[w^T \phi(x_i) - y_i - \varepsilon + \hat{\xi}_i\right] = 0, \quad \forall i \in \{1, \ldots, n\} \tag{33}$$

$$\breve{\alpha}_i \left[y_i - w^T \phi(x_i) - \varepsilon + \breve{\xi}_i\right] = 0, \quad \forall i \in \{1, \ldots, n\} \tag{34}$$

$$\hat{\eta}_i \hat{\xi}_i, \breve{\eta}_i \breve{\xi}_i = 0, \quad \forall i \in \{1, \ldots, n\} \tag{35}$$

By exploiting these relationships in equation (26), the dual formulation of problem (23) is derived, for which several efficient solvers have been developed throughout the last years[56,57]

$$\hat{\alpha}^*, \breve{\alpha}^* : \min_{\hat{\alpha}, \breve{\alpha}} \quad \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \left(\breve{\alpha}_i - \hat{\alpha}_i\right)\left(\breve{\alpha}_j - \hat{\alpha}_j\right) \phi(x_i)^T \phi(x_j)$$
$$+ \sum_{i=1}^n \hat{\alpha}_i[y_i + \varepsilon] - \sum_{i=1}^n \breve{\alpha}_i[y_i - \varepsilon]$$
$$0 \leqslant \breve{\alpha}_i \leqslant C, \quad \forall i \in \{1, \ldots, n\}$$
$$0 \leqslant \hat{\alpha}_i \leqslant C, \quad \forall i \in \{1, \ldots, n\} \tag{36}$$

The trained approximating function $h^*(x)$ can be expressed through $\hat{\alpha}^*, \breve{\alpha}^*$

$$h^*(x) = \sum_{i=1}^n \left(\breve{\alpha}_i^* - \hat{\alpha}_i^*\right) \phi(x_i)^T \phi(x) \tag{37}$$

It is worth noting that if $|w^T \phi(x_i) - y_i| \leqslant \varepsilon$, it derives from equations (33) and (34) that $\hat{\alpha}_i = \breve{\alpha}_i = 0$; as a consequence, the solution of SVR is usually sparse, that is, it can be described by exploiting only a limited subset of parameters (and, obviously, of training data). In particular, the less the data outside the ε-insensitive tube, the more sparse the SVR solution.[20,37,38] In other words, let $\mathcal{S}_{sv} = \{i : \hat{\alpha}_i = \breve{\alpha}_i = 0\}$ be the set of indexes, and $|\mathcal{S}_{sv}| \leqslant n$ (and, usually, $|\mathcal{S}_{sv}| \ll n$) its cardinality; then, in order to compute $h^*(x)$, only data indexed by $\mathcal{S}_{sv}$ can be contemplated

$$h^*(x) = \sum_{i \in \mathcal{S}_{sv}} \left(\breve{\alpha}_i^* - \hat{\alpha}_i^*\right) \phi(x_i)^T \phi(x) \tag{38}$$

This allows to reduce the computational burden with respect to RLS, for which the solution is dense (i.e. almost all $\alpha_i \neq 0$ in equation (21)).

As a final issue, analogously to what is done for RLS, the kernel trick can be exploited in order to reformulate the SVR training problem by exploiting a matrix formulation

$$\hat{\boldsymbol{\alpha}}^*, \breve{\boldsymbol{\alpha}}^* : \min_{\hat{\boldsymbol{\alpha}}, \breve{\boldsymbol{\alpha}}} \frac{1}{2} \begin{bmatrix} \breve{\boldsymbol{\alpha}} \\ \hat{\boldsymbol{\alpha}} \end{bmatrix}^T \begin{bmatrix} K & -K \\ -K & K \end{bmatrix} \begin{bmatrix} \breve{\boldsymbol{\alpha}} \\ \hat{\boldsymbol{\alpha}} \end{bmatrix} + \begin{bmatrix} \breve{\boldsymbol{\alpha}} \\ \hat{\boldsymbol{\alpha}} \end{bmatrix}^T \begin{bmatrix} -\boldsymbol{y} + \boldsymbol{\varepsilon} \\ \boldsymbol{y} + \boldsymbol{\varepsilon} \end{bmatrix}$$

$$0 \leqslant \breve{\boldsymbol{\alpha}}, \hat{\boldsymbol{\alpha}} \leqslant C \qquad\qquad (39)$$

Tuning the hyperparameters λ, γ, C and ε

The performance of RLS and SVR models depends on the quality of the hyperparameters' tuning. As highlighted while presenting these approaches, while the parameters $\boldsymbol{\alpha}^*, \hat{\boldsymbol{\alpha}}^*$ and $\breve{\boldsymbol{\alpha}}^*$ are identified by the optimization procedure, the tuples of hyperparameters (λ, γ) for RLS and (γ, C, ε) for SVR must be set before finding the model; this a priori tuning phase is known as the *model selection* step[58–60] and is run as a part of the training phase. Some rule-of-thumb approaches have been proposed throughout the years,[22] which allow to tune hyperparameters in a quick way; however, the quality of the generated model is obviously not the best one. On the contrary, the most effective approaches consist in performing an exhaustive hyperparameter grid search; the optimization problem for RLS/SVR is solved several times for different values of the tuples, and the learned models are benchmarked according to some criteria. The most widespread quality index for ML models is the estimated generalization error of the regressor, that is, the error that the function will be likely performing on new unknown data, generated by the same probability distribution that originated the learning set.[61]

Several methods exist for this purpose; *out-of-sample* methods, like the well-known *k-fold cross validation (KCV)*[62] approach, represent the state-of-the-art model selection approaches when targeting several applications.[60] In KCV, the original dataset X is split into k-independent subsets (namely, the folds), each one consisting of n/k samples; $(k - 1)$ parts are used, in turn, as a training set, and the remaining fold is exploited as a validation set. Note, in fact, that the error performed by the trained model on the validation set can be reliably used for estimating the generalization error because this fold has not been used for training the model.[61–64] The procedure is iterated k times. It is worth underlining that k itself could be considered as a hyperparameter;[65] however, in practice, $k = 5$, 10 or 20 represent feasible choices in the largest part of applications.[65,66]

The KCV procedures for RLS and SVR are presented in Algorithms 1 and 2, respectively; they only differ in the way the hyperparameter tuples are defined and, obviously, in the procedure used for model training. Given a training set X, a fixed number of folds k, and a grid of values to be explored for the hyperparameters λ, γ, C and/or ε (depending on the chosen

approach, that is, either RLS or SVR), the original dataset is split into k folds, as a first step; let $\mathcal{S} = \{1, \ldots, n\}$ be the set of indexes of the samples in X; then, k sets of indexes $\mathcal{S}^{(t)}$, $t = 1, \ldots, k$, are created. (Ideally, the range where to search for the hyperparameters should be $\lambda, \gamma, C, \varepsilon \in [0, +\infty)$, and the search grid should have infinite resolution. Obviously, both the coarseness of the grid and the size of the searching space severely influence the quality of the solution and the amount of computation time needed by the learning procedure. Nevertheless, some practical suggestions can be retrieved in literature,[66] which help finding good trade-offs between performance and computational time.) These subsets are such that

$$\bigcup_{t=1}^{k} \mathcal{S}^{(t)} = \mathcal{S} \quad \vee \quad \bigcap_{t=1}^{k} \mathcal{S}^{(t)} = \varnothing \qquad (40)$$

The best error for the KCV procedure is initialized to $L_{\mathrm{KCV}}^* = +\infty$. Then, for every possible set of values in the grid for the hyperparameters' tuple, (λ, γ) for RLS or (γ, C, ε) for SVR, a KCV stage is run. At the tth KCV step, the patterns in the validation set are indexed by $\mathcal{S}^{(t)}$, while the training set consists of the remaining samples, whose indexes are $(\mathcal{S} \backslash \mathcal{S}^{(t)})$. A model is then trained on the training set, and its performance is verified on the validation patterns by computing the empirical error

$$\hat{L}_{n/k}^{(t)}(h^{(t)}) = \frac{1}{\frac{n}{k}} \sum_{i \in \mathcal{S}^{(t)}} \ell(h^{(t)}(\boldsymbol{x}_i), y_i) \qquad (41)$$

After the k KCV steps are over, the estimated generalization error for the hyperparameters' tuple can be computed by averaging the t empirical errors on the validation sets

$$L_{\mathrm{KCV}} = \frac{1}{k} \sum_{t=1}^{k} \hat{L}_{n/k}^{(t)}(h) \qquad (42)$$

If the obtained error is lower than the best L_{KCV}^* reported so far, the hyperparameters' tuple is saved as potential best configuration for the learning procedure.

As a final issue, the authors want to point out that in order to select the best hyperparameters' tuple, k models are created at every KCV step. Once the best tuple is found, the final model is trained on the whole set X by running the learning procedure where hyperparameters are set to the identified best values.

Dataset creation

In this article, the authors aim to develop a predictive model for CBM purposes with reference to GTs mounted on a Frigate naval vessel. Since the CODLAG propulsion plant, mounted on the Frigate, is of recent installation, no real-world data gathered from ship's automation system are currently available, which could help studying and validating an ML solution to the

---

Algorithm 1. The *k*-Fold Cross Validation (KCV) model selection procedure for RLS.

Require: X, k, a grid for every hyperparameter involved $\mathcal{G}_{\lambda, \gamma}$

1. $L^*_{KCV} = +\infty$
2. $(\lambda^*, \gamma^*) = (-\infty, -\infty)$
3. $\mathcal{S}^{(t)} = \text{split}(\mathcal{S})$, $t = 1, \ldots, k$
4. **for all** $\lambda \in \mathcal{G}_\lambda$, $\gamma \in \mathcal{G}_\gamma$ **do**
5.   Let $(\lambda^{(p)}, \gamma^{(q)})$ be the current hyperparameters tuple
6. $L_{KCV} = 0$
7.   **for** $t = 1 \rightarrow k$ **do**
8.   Train $h^{(t)}$ with RLS Problem (21) on the set of data indexed by $(\mathcal{S} \backslash \mathcal{S}^{(t)})$, by using the hyperparameter tuple $(\lambda^{(p)}, \gamma^{(q)})$
9.   Compute $\hat{L}^{(t)}_{n/k}(h^{(t)})$ according to Eq. (41)
10.   **end for**
11.   Compute L_{KCV} according to Eq. (42)
12.   **if** $L_{KCV} < L^*_{KCV}$ **then**
13. $L^*_{KCV} = L_{KCV}$
14. $(\lambda^*, \gamma^*) = (\lambda^{(p)}, \gamma^{(q)})$
15.   **end if**
16. **end for**
17. Train h^* with RLS Problem (21) by using the whole set X and the tuple (λ^*, γ^*)
18. **return** h^*

Algorithm 2. The *k*-Fold Cross Validation (KCV) model selection procedure for SVR.

Require: X, k, a grid for every hyperparameter involved $\mathcal{G}_{\gamma, C, \varepsilon}$

1. $L^*_{KCV} = +\infty$
2. $(C^*, \gamma^*, \varepsilon^*) = (-\infty, -\infty, -\infty)$
3. $\mathcal{S}^{(t)} = \text{split}(\mathcal{S})$, $t = 1, \ldots, k$
4. **for all** $C \in \mathcal{G}_C$, $\gamma \in \mathcal{G}_\gamma$, $\varepsilon \in \mathcal{G}_\varepsilon$ **do**
5.   Let $(C^{(p)}, \gamma^{(q)}, \varepsilon^{(s)})$ be the current hyperparameters tuple
6. $L_{KCV} = 0$
7.   **for** $t = 1 \rightarrow k$ **do**
8.   Train $h^{(t)}$ with SVR Problem (39) on the set of data indexed by $(\mathcal{S} \backslash \mathcal{S}^{(t)})$, by using the hyperparameter tuple $(C^{(p)}, \gamma^{(q)}, \varepsilon^{(s)})$
9.   Compute $\hat{L}^{(t)}_{n/k}(h^{(t)})$ according to Eq. (41)
10.   **end for**
11.   Compute L_{KCV} according to Eq. (42)
12.   **if** $L_{KCV} < L^*_{KCV}$ **then**
13. $L^*_{KCV} = L_{KCV}$
14. $(C^*, \gamma^*, \varepsilon^*) = (C^{(p)}, \gamma^{(q)}, \varepsilon^{(s)})$
15.   **end if**
16. **end for**
17. Train h^* with SVR Problem (39) by using the whole set X and the tuple $(C^*, \gamma^*, \varepsilon^*)$
18. **return** h^*

CBM needs of this asset. Nevertheless, much of the effort has been spent in realizing the physical model and, consequently, an accurate simulator of the system, as introduced in section "Ship propulsion models" as well as in previous works.[23,24] As a consequence, the model of section "Ship propulsion models" can be exploited in order to create a virtual (but ground-truth complying[23,24]) experimental datasets, to be used for ML modeling purposes.

As described in section "Asset degradation model," the propulsion system is depicted through three parameters:

1. *Speed*: this parameter is controlled via the control lever. The latter can only assume a finite number of position $lp_k, k \in \{0, \ldots, 9\}$, which in turn corresponds to a finite set of possible configurations for fuel flow and blade position. Each set point is designed to reach a desired speed $v_k, k \in \{0, \ldots, 9\}$

$$lp_i = i, \quad i \in \{0, \ldots, 9\} \tag{43}$$

$$lp_i \rightarrow v_i = 3 * lp_i(\text{knots}), \quad \forall i \in \{0, \ldots, 9\} \tag{44}$$

Note that, if the transients are not taken into account, lp_i and v_i are deterministically related by a linear law. In the presented analysis, the transients between different speeds are not considered since, in the operational life of a ship, these phenomena cover a negligible time with respect to steady states.

2. *Compressor decay*

$$kM_c \in [0.95, 1] \tag{45}$$

The decay rate has been sampled through a uniform grid with resolution 10^{-3}, so as to reach a good granularity when representing the GT compressor decay state

$$kM_c^i = 1 - i \cdot 0.001, \quad i \in \{0, 1, \ldots, 51\} \tag{46}$$

3. *GT decay*

$$kM_t \in [0.975, 1] \tag{47}$$

Analogous to the case of compressor, the decay rate is sampled through a uniform grid with resolution 10^{-3}. The GT decay state can be then represented as

$$kM_t^i = 1 - i \cdot 0.001, \quad i \in \{0, 1, \ldots, 26\} \tag{48}$$

Once these quantities are fixed, several heterogeneous measures, related to the propulsion plant and listed in Table 1, can be derived by exploiting the simulator, punctually describing the state of the system. The purpose of the analysis is to train a model on such measures, in order to estimate kM_c and kM_t from them. It is worthwhile noting that the measures, derived by the simulator, can also be obtained by the automation system equipping the ship and could be used in a real-world application scenario as well.

Given the above premises, the evolution of the system can exhaustively explored by simulating all its possible states. The space of possible states is described via the triples

$$(lp, kM_c, kM_t)_i, \quad i \in \{0, 1, \ldots, 11,934\} \tag{49}$$

since

$$lp_i = i, \quad i \in \{0, \ldots, 9\} \tag{50}$$

$$kM_c^i = 1 - i \cdot 0.001, \quad i \in \{0, 1, \ldots, 51\} \tag{51}$$

$$kM_t^i = 1 - i \cdot 0.001, \quad i \in \{0, 1, \ldots, 26\} \tag{52}$$

It is then possible to run the simulation for the $9 \cdot 51 \cdot 26 = 11,934$ conditions considered in the MATLAB Simulink environment for each triple $[lp, kM_c, kM_t]_i$. Every model is run until the steady state is reached; then, the above-cited 16 measures listed in Table 1 can be extracted from the simulation and finally organized into a dataset. The latter defines a conventional MO problem, where every sample $\boldsymbol{x}_i \in \mathbb{R}^{16}$ and every output is $\boldsymbol{y}_i \in \mathbb{R}^2$, as it includes the values of kM_c and kM_t.

The authors intend to publish this dataset for free use by the scientific community through the well-known UCI repository.[67] While waiting for final approval for publication, news and preliminary releases of the dataset will be soon available at the dedicated webpage http://cbm.smartlab.ws

Experimental results and discussion

Given the complexity of the considered problem, the authors decided to tackle it through decomposition in simpler subproblems. In particular, the authors start by targeting the prediction of the GT compressor decay when the GT turbine decay has no effects on the system; then, a similar analysis is pursued, where GT compressor decay effects are bypassed; finally, a complete analysis is carried out, where both decay effects are included.

Modeling and predicting the compressor decay

The dataset is simply created by isolating the effects of kM_c from the ones of kM_t; starting from the dataset created according to the methodology of section "Dataset creation," this goal can be reached by only considering those experiments that correspond to values of $kM_t = 1$. In this way, only the GT compressor decay effects on the system, thus allowing to focus on a simpler, but still significant,[34,35] problem. The dataset X consists of $9 \cdot 51 = 459$ experiments, that is, the ones corresponding to the following tuples

$$(lp, kM_c)_i, \quad i \in \{0, 1, \ldots, 459\} \tag{53}$$

$$lp_i = i, \quad i \in \{0, \ldots, 9\} \tag{54}$$

$$kM_c^i = 1 - i \cdot 0.001, \quad i \in \{0, 1, \ldots, 51\} \tag{55}$$

In order to train an effective model and to test its performance in a real-world-like scenario, the dataset X of available experiments is divided into two subsets:

* X_{train}, consisting of n_{train} samples, which is used for the training phase (thus including model selection and creation).
* $X_{\text{test}} = X \backslash X_{\text{train}}$, consisting of $n_{\text{test}} = n - n_{\text{train}}$ data. The feedforward phase of the model h^*, trained on X_{train}, is run on these samples analogously to an application setting of the regressor. As the target outputs are available for X_{test}, it is always possible to compute the error that h^* would obtain in the application scenario, which is defined as

$$\hat{L}_{n_{\text{test}}}^{\text{test}}(h^*) = \frac{1}{n_{\text{test}}} \sum_{\boldsymbol{x}_i \in X_{\text{test}}} \ell(h^*(\boldsymbol{x}_i), y_i) \tag{56}$$

In particular, the authors opt for exploiting the relative error[68,69] as loss function, which can be expressed as

$$\ell(h^*(\boldsymbol{x}_i), y_i) = \left| \frac{h^*(\boldsymbol{x}_i) - y_i}{y_i} \right| \tag{57}$$

The splitting of the dataset into training and test sets is a key point to deal with. In particular, two aspects must be considered: (1) the choice of n_{train}, that is, the amount of data for training purposes and (2) the way data are picked from X to be included into X_{train}. Both points can be analyzed from the modeling and application point of view. Regarding issue (1) above:

- From the ML point of view, choosing the most proper number of data for training and test purposes has a non-negligible impact on the quality of the final model; if n_{train} is too large, the model will generally be more accurate, but the test assessment probably will not be relevant from a statistical point of view; on the contrary, if too many samples are reserved for testing purposes, the regressor could become too inaccurate, being trained on a limited knowledge base. Usually, a feasible approach consists in performing some trials, by using different increasing values for n_{train}, in order to find the best trade-off.
- From the naval application point of view, it is worth underlining that the role of the simulator would be played by an expert operator, who checks the decay state of the component and reports any measures, coming from the ship's automation, into a database. The purpose of the ML predictive approach for CBM, targeted by this article, is to support the expert in such monitoring operations. Determining the value of n_{train} is equivalent to quantifying the number of manual inspection the expert has to complete before the model can be trained, or, in other words, before he is supported in his activities by the ML predictive model for CBM. Too large values for n_{train} would require a remarkable effort by the expert and/or long waiting time before operating the model, reducing the potential impact of such technology in the maritime domain.

Instead, when issue (2) above is taken into account:

- From the ML point of view, data can be picked either in a random way or through a more rigorous procedure, which exploits some information on the observed phenomenon. Random selection is obviously easier to implement, but usually leads to poorer quality models, as the space of possible values for measures and decay rates is not exhaustively explored. Approaches, allowing to explore the topology representation of data, usually enable deriving more effective regressors at the expenses of a small supplementary delta effort; it is the case, for example, of the nearly homogeneous sampling method,[70] that can be exploited in the experiments.
- The way data are sampled in this virtual framework maps into the protocol for data gathering for maintenance purposes in the real-world application. In other words, a random sampling corresponds to an expert operator that checks for the health status of the vessel components without following any predetermined and/or rigorous time schedule and/or procedure; on the contrary, using more sophisticated sampling techniques is equivalent to an operator, who follows a planned schedule on predetermined operational conditions for monitoring the vessel.

Though this second scenario seems to be more plausible, both settings will be analyzed in the following.

As described in section "Asset degradation model," the Frigate can be characterized by different mission profiles (AAW, ASW, GP), which correspond to different operational profiles; they are distinguished by the different PDFs, used for depicting the distribution of the speed. Such information cannot be straightforwardly retrieved in the dataset, created as described in the previous section. Nevertheless, the whole space of potential operating conditions is mapped; it is always possible to resample the dataset according to different speed distributions, designed with reference to the different mission profiles. Unfortunately, such profiles are confidential information not available in public literature; however, it is always possible to design some reasonable distributions, based on authors' expertise. In particular, the following scenarios will be considered:

1. A uniform distribution (UD) (Figure 6), characterizing a general-purpose mission of the Frigate;
2. A bimodal distribution (BD) (Figure 7), where the two modes are centered on the cruising speed and on the maximum speed, respectively (while other speed values are neglected);
3. A monomodal distribution (MD) (Figure 8), where the mode is centered over the cruise speed and the time, spent by the vessel at different speed values, is considered negligible.

For the sake of completeness, it is worth noting that the ASW mission profile can be characterized by a trimodal distribution; in this case, another mode is introduced in addition to the ones of the BD profile, centered on the optimal speed for the antisubmarine operations. However, in this analysis, the last case is
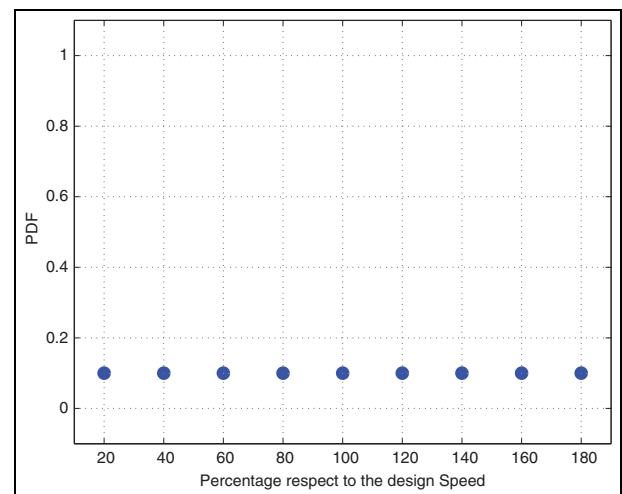

Figure 6. Uniform distribution (UD), used as PDF of the speed (percentage with respect to the design speed equals to 15 knot) of the Frigate.
PDF: probability density function.

Figure 7. Bimodal distribution (BD), used as PDF of the speed (percentage with respect to the design speed equals to 15 knot) of the Frigate.
PDF: probability density function.
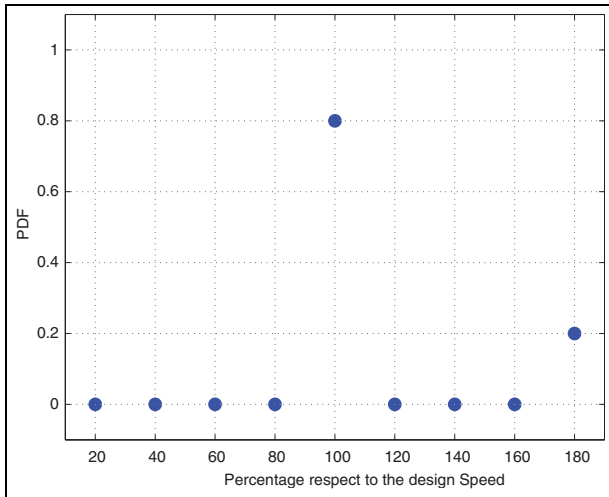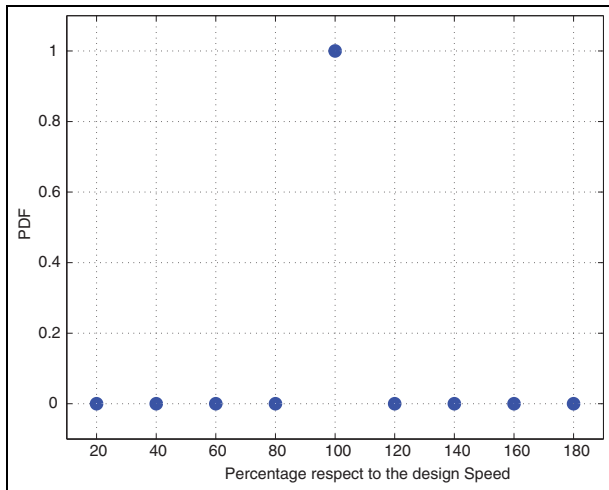

Figure 8. Monomodal distribution (MD), used as PDF of the speed (percentage respect to the design speed equals to 15 knot) of the Frigate.
PDF: probability density function.

not taken into account since the vessel exploits the electrical propulsion mode, which is not contemplated by the MATLAB Simulink model, for reaching the first peak[71,72] in place of the GT mode. Consequently, the ASW mission profile can still be reasonably represented by BD, as introduced above.

Table 2 shows the results for the experiments, where only GT compressor decay is taken into account. In particular, the error on the independent test set $\hat{L}_{n_{test}}^{test}(h^*)$ is reported. The error is computed for different experimental setups listed below:

- Different experiments are performed by varying the amount of patterns used for training purposes (refer to issue (1), introduced above). In particular, $n_{train} \in \{10, 25, 50, 100, 200\}$ were tested.

- The experiments have been replicated for different values of n_{train}, by opting for different sampling strategies (refer to issue (2) above). In particular, a random sampling and the nearly homogeneous sampling[70] strategies are exploited. Note, moreover, that the splitting procedure for creating X_{train} and X_{test} from X is replicated 30 times per each experiment, so as to derive statistically relevant results.

- The experiments were replicated by varying the probability distribution of vessel speed according to the three distributions, previously introduced (UD, BD, and MD).

- The RLS and SVR learning procedures presented in Algorithms 1 and 2 are used, respectively, to derive the model h^*. In particular, the authors set the following:

- *RLS*: the grids for hyperparameters' tuning purposes are fixed to $\mathcal{G}_\lambda \in [10^{-6}, 10^4]$ and $\mathcal{G}_\gamma \in [10^{-6}, 10^4]$, consisting of 30 values equally spaced in logarithmic scale.

- *SVR*: the grids are fixed to $\mathcal{G}_C \in [10^{-6}, 10^4]$ and $\mathcal{G}_\gamma \in [10^{-6}, 10^4]$, consisting of 30 values equally spaced in logarithmic scale. Note that SVR requires that the hyperparameter ε is tuned as well. Nevertheless, this hyperparameter is application-dependent and can be usually aprioristically set to a fixed value without compromising performance while reducing the computational burden. The value of ε is set to $\varepsilon = 10^{-3}$, that is, equal to the considered granularity when modeling the system behavior.

Table 3 presents the results for the set of experiments, performed on the data; in particular, the average error and its variance, as recorded on the different performed analyses, are shown. From them, it is worthwhile deriving the following:

- Broadly speaking, SVR is characterized by better performance than RLS; as a matter of fact, SVR average error is smaller, and the variance values for SVR show that this approach is more stable than RLS as the training set is changed (e.g. because of different speed distributions or of different sampling strategies).

- The model derived from datasets, sampled according to the nearly homogeneous sampling[70] approach, is usually more accurate. However, the difference with respect to the performance of regressors, trained on randomly sampled training sets, is negligible; such result allows to derive precious information for designing the procedures for CBM data gathering on the Frigate.

- On equal terms (e.g. value of n_{train}), if the UD or the vessel speed is contemplated, which takes into account a broad range of potential speed values, the forecasting is less accurate than in those experiments, where more concentrated distributions are

Table 2. Test error rates (in percentage) when only the gas turbine compressor decay is considered.

n_{train}	Random sampling					
	UD		BD		MD	
	RLS $\hat{L}_{n_{\text{test}}}^{\text{test}}(h^*)$	SVR $\hat{L}_{n_{\text{test}}}^{\text{test}}(h^*)$	RLS $\hat{L}_{n_{\text{test}}}^{\text{test}}(h^*)$	SVR $\hat{L}_{n_{\text{test}}}^{\text{test}}(h^*)$	RLS $\hat{L}_{n_{\text{test}}}^{\text{test}}(h^*)$	SVR $\hat{L}_{n_{\text{test}}}^{\text{test}}(h^*)$
10	30.23±5.57	27.57±5.18	5.36±1.95	4.81±1.70	0.96±0.02	0.87±0.02
25	12.39±2.50	11.16±2.28	1.08±0.22	0.98±0.20	0.44±0.00	0.39±0.00
50	5.13±0.57	4.67±0.52	0.34±0.01	0.31±0.01	0.28±0.00	0.24±0.00
100	1.59±0.15	1.42±0.13	0.17±0.00	0.15±0.00	0.13±0.00	0.12±0.00
200	0.50±0.01	0.45±0.01	0.08±0.00	0.08±0.00	0.04±0.00	0.04±0.00
n_{train}	Nearly homogeneous sampling[70]					
	UD		BD		MD	
	RLS $\hat{L}_{n_{\text{test}}}^{\text{test}}(h^*)$	SVR $\hat{L}_{n_{\text{test}}}^{\text{test}}(h^*)$	RLS $\hat{L}_{n_{\text{test}}}^{\text{test}}(h^*)$	SVR $\hat{L}_{n_{\text{test}}}^{\text{test}}(h^*)$	RLS $\hat{L}_{n_{\text{test}}}^{\text{test}}(h^*)$	SVR $\hat{L}_{n_{\text{test}}}^{\text{test}}(h^*)$
10	25.96±4.93	23.59±4.56	4.61±1.60	4.28±1.43	0.83±0.02	0.74±0.02
25	10.55±2.16	9.35±1.99	0.91±0.19	0.82±0.17	0.37±0.00	0.33±0.00
50	4.47±0.49	4.10±0.44	0.30±0.01	0.27±0.01	0.23±0.00	0.21±0.00
100	1.35±0.13	1.20±0.11	0.14±0.00	0.13±0.00	0.12±0.00	0.10±0.00
200	0.42±0.01	0.37±0.01	0.07±0.00	0.07±0.00	0.04±0.00	0.03±0.00

UD: uniform distribution; BD: bimodal distribution; MD: monomodal distribution; RLS: regularized least square; SVR: support vector regression.

exploited (e.g. MD and BD, respectively). This is expected; in fact, the use of MD and BD simplifies the problem by neglecting several different speeds, which are considered, on the contrary, by UD. This allows to highlight the importance of correctly and informatively depicting the speed distribution of the vessel, as a basic condition for deriving more effective models with a smaller amount of gathered data.

- Finally, $n_{\text{train}} \approx 50$ samples are usually enough, in the proposed approaches, to derive an accurate model for CBM purposes, in this preliminary study framework.

Modeling and predicting the GT decay

As a second issue, the dataset is created by isolating the effects of the GT decay kM_t from the ones of the GT compressor kM_c (i.e. only those tuples where $kM_c = 1$ are considered). In this case, the dataset X consists of $9 \cdot 26 = 234$ experiments, that is, the ones that correspond to

$$(lp, kM_t)_i, \quad i \in \{0, 1, \ldots, 234\} \tag{58}$$

$$lp_i = i, \quad i \in \{0, \ldots, 9\} \tag{59}$$

$$kM_t^i = 1 - i \cdot 0.001, \quad i \in \{0, 1, \ldots, 26\} \tag{60}$$

The experimental setup is not changed from the one depicted in the previous section; however, as a very partial exception, the case $n_{\text{train}} = 200$ is neglected due to the more limited amount of available data (234 vs 459). In fact, when $n_{\text{train}} = 200$, only 60 samples are available for testing purposes, which could be too few to target a reliable analysis.

Table 3 shows the results obtained from the performed tests; the quantities, presented in the table, are analogous to the ones of Table 2, and so are the conclusions that can be drawn from them.

Modeling and predicting both the GT compressor and the GT decays

As a final issue, the case where both decay factors are taken into account is analyzed, so depicting an MO problem with $m = 2$ (refer to section "ML framework" for more details). By exploiting the approach proposed by Zhang et al.[46] and Baldassarre et al.,[47] that is, by decomposing the MO problem into m SO subproblems, a model to estimate kM_c and kM_t in the general CBM scenario is created. The objective is to verify whether the models are able to accurately track variations in GT parameters, due to decay, even when two components are degrading their performance (and not necessarily in a synchronous fashion).

The experimental setup is kept unmodified with reference to the one applied for previous experiments. Clearly, at this stage, the authors considered the whole dataset generated according to the considerations, reported in section "Dataset creation"; having 11,934 samples available, larger cardinalities for the training set are tested, that is, $n_{\text{train}} \in \{10, 25, 50, 100, 205, 500\}$. The results are shown in Tables 4 and 5, which are referred to the GT compressor and the GT decay, respectively. Note that

- In spite of the contemporaneous variations in two decay factors, the trend of the error, performed by

Table 3. Test error rates (in percentage) when only the gas turbine decay is considered.

n_{train}	Random sampling					
	UD		BD		MD	
	RLS $\hat{L}_{n_{test}}^{test}(h^*)$	SVR $\hat{L}_{n_{test}}^{test}(h^*)$	RLS $\hat{L}_{n_{test}}^{test}(h^*)$	SVR $\hat{L}_{n_{test}}^{test}(h^*)$	RLS $\hat{L}_{n_{test}}^{test}(h^*)$	SVR $\hat{L}_{n_{test}}^{test}(h^*)$
10	29.77±5.26	27.07±4.84	4.14±1.44	3.77±1.29	0.50±0.00	0.45±0.00
25	15.41±4.45	13.94±4.08	0.40±0.01	0.36±0.01	0.38±0.00	0.34±0.00
50	4.50±1.35	4.07±1.24	0.28±0.03	0.25±0.03	0.21±0.00	0.18±0.00
100	1.60±0.17	1.44±0.16	0.11±0.00	0.09±0.00	0.04±0.00	0.03±0.00
n_{train}	Nearly homogeneous sampling[70]					
	UD		BD		MD	
	RLS $\hat{L}_{n_{test}}^{test}(h^*)$	SVR $\hat{L}_{n_{test}}^{test}(h^*)$	RLS $\hat{L}_{n_{test}}^{test}(h^*)$	SVR $\hat{L}_{n_{test}}^{test}(h^*)$	RLS $\hat{L}_{n_{test}}^{test}(h^*)$	SVR $\hat{L}_{n_{test}}^{test}(h^*)$
10	25.52±4.60	23.25±4.23	3.63±1.20	3.38±1.12	0.43±0.00	0.38±0.00
25	13.20±3.88	11.67±3.50	0.34±0.00	0.31±0.00	0.32±0.00	0.29±0.00
50	3.90±1.18	3.57±1.07	0.23±0.03	0.22±0.02	0.17±0.00	0.15±0.00
100	1.36±0.15	1.22±0.13	0.09±0.00	0.08±0.00	0.03±0.00	0.03±0.00

UD: uniform distribution; BD: bimodal distribution; MD: monomodal distribution; RLS: regularized least square; SVR: support vector regression.

Table 4. Test error rate (in percentage) when forecasting the compressor decay in the multiple output setup.

n_{train}	Random sampling					
	UD		BD		MD	
	RLS $\hat{L}_{n_{test}}^{test}(h^*)$	SVR $\hat{L}_{n_{test}}^{test}(h^*)$	RLS $\hat{L}_{n_{test}}^{test}(h^*)$	SVR $\hat{L}_{n_{test}}^{test}(h^*)$	RLS $\hat{L}_{n_{test}}^{test}(h^*)$	SVR $\hat{L}_{n_{test}}^{test}(h^*)$
10	28.95±4.31	26.32±4.02	6.98±2.34	6.20±2.06	1.98±0.05	1.79±0.05
25	15.80±2.45	14.26±2.24	1.50±0.12	1.38±0.11	0.69±0.01	0.61±0.00
50	6.21±0.41	5.62±0.37	0.55±0.01	0.49±0.01	0.44±0.00	0.38±0.00
100	2.72±0.19	2.43±0.17	0.32±0.00	0.29±0.00	0.27±0.00	0.25±0.00
250	0.78±0.03	0.69±0.02	0.20±0.00	0.18±0.00	0.16±0.00	0.14±0.00
500	0.40±0.00	0.35±0.00	0.09±0.00	0.08±0.00	0.09±0.00	0.08±0.00
n_{train}	Nearly homogeneous sampling[70]					
	UD		BD		MD	
	RLS $\hat{L}_{n_{test}}^{test}(h^*)$	SVR $\hat{L}_{n_{test}}^{test}(h^*)$	RLS $\hat{L}_{n_{test}}^{test}(h^*)$	SVR $\hat{L}_{n_{test}}^{test}(h^*)$	RLS $\hat{L}_{n_{test}}^{test}(h^*)$	SVR $\hat{L}_{n_{test}}^{test}(h^*)$
10	24.78±3.83	22.54±3.48	5.97±1.95	5.56±1.79	1.68±0.04	1.51±0.04
25	13.49±2.14	12.01±1.93	1.30±0.10	1.16±0.09	0.58±0.00	0.53±0.00
50	5.40±0.35	4.91±0.32	0.46±0.01	0.42±0.01	0.37±0.00	0.33±0.00
100	2.30±0.16	2.06±0.15	0.28±0.00	0.25±0.00	0.24±0.00	0.21±0.00
250	0.65±0.02	0.58±0.02	0.17±0.00	0.15±0.00	0.14±0.00	0.12±0.00
500	0.33±0.00	0.29±0.00	0.08±0.00	0.07±0.00	0.08±0.00	0.07±0.00

UD: uniform distribution; BD: bimodal distribution; MD: monomodal distribution; RLS: regularized least square; SVR: support vector regression.

the trained models, is qualitatively the same as the one observed when contemplating one decay at a time. This validates the choice of opting for the decomposition of the MO problem into separate SO ones.

- Nevertheless, slightly larger number of training samples is generally needed to create effective models. As a matter of fact, around 100 data are typically required to gain low error rates, which can rise up to 100/200 when a UD is used to describe the vessel speed.

- Finally, similar conclusions to the ones drawn in the previous subsections can be made.

Conclusion

The impact of maintenance cost in the naval domain is remarkable; as a consequence, this enables the necessity

Table 5. Test error rate (in percentage) when forecasting the turbine decay in the multiple output setup.

n_{train}	Random sampling					
	UD		BD		MD	
	RLS $\hat{L}_{n_{test}}^{test}(h^*)$	SVR $\hat{L}_{n_{test}}^{test}(h^*)$	RLS $\hat{L}_{n_{test}}^{test}(h^*)$	SVR $\hat{L}_{n_{test}}^{test}(h^*)$	RLS $\hat{L}_{n_{test}}^{test}(h^*)$	SVR $\hat{L}_{n_{test}}^{test}(h^*)$
10	32.00±4.97	29.31±4.61	9.41±2.34	8.52±2.03	1.77±0.04	1.60±0.04
25	25.27±3.56	22.83±3.23	1.96±0.36	1.78±0.32	0.65±0.00	0.58±0.00
50	12.71±2.23	11.50±2.01	0.67±0.02	0.60±0.02	0.46±0.00	0.40±0.00
100	4.95±0.93	4.43±0.85	0.39±0.00	0.35±0.00	0.33±0.00	0.30±0.00
250	1.34±0.10	1.19±0.09	0.24±0.00	0.22±0.00	0.18±0.00	0.16±0.00
500	0.63±0.02	0.55±0.02	0.17±0.00	0.15±0.00	0.10±0.00	0.09±0.00
n_{train}	Nearly homogeneous sampling[70]					
	UD		BD		MD	
	RLS $\hat{L}_{n_{test}}^{test}(h^*)$	SVR $\hat{L}_{n_{test}}^{test}(h^*)$	RLS $\hat{L}_{n_{test}}^{test}(h^*)$	SVR $\hat{L}_{n_{test}}^{test}(h^*)$	RLS $\hat{L}_{n_{test}}^{test}(h^*)$	SVR $\hat{L}_{n_{test}}^{test}(h^*)$
10	27.62±4.40	25.26±3.97	8.20±1.93	7.54±1.75	1.50±0.04	1.36±0.03
25	21.59±3.09	19.19±2.79	1.67±0.32	1.49±0.28	0.55±0.00	0.49±0.00
50	11.02±1.92	10.05±1.72	0.57±0.02	0.52±0.01	0.39±0.00	0.34±0.00
100	4.18±0.80	3.75±0.70	0.34±0.00	0.30±0.00	0.29±0.00	0.26±0.00
250	1.12±0.08	1.00±0.08	0.20±0.00	0.19±0.00	0.15±0.00	0.14±0.00
500	0.52±0.02	0.46±0.01	0.14±0.00	0.13±0.00	0.09±0.00	0.08±0.00

UD: uniform distribution; BD: bimodal distribution; MD: monomodal distribution; RLS: regularized least square; SVR: support vector regression.

of designing more effective approaches, for example, based on CBM. Relying on the need of constantly monitoring the amounts of data, gathered from the field, CBM strategies require that proper ML models are implemented, able to highlight potential safety issues and/or intervention requests with a reasonable advance. In this article, in particular, the authors focused on an application of CBM to GTs used for naval vessel propulsion to propose an innovative modeling strategy based on effective ML approaches, targeted toward forecasting the degradation of the propulsion plant over time.

In order to test the proposed techniques, a sophisticated and realistic simulator of a naval propulsion system is exploited, mounted on a Frigate characterized by a CODLAG, to generate a large dataset, that will be soon released to the research community for benchmarking purposes. The proposed approach has been tested on this dataset, showing its effectiveness for CBM purposes in a real-world maritime application, even in the challenging scenario where more than one decay factor is varied over time.

The approach and results, shown in this article, clearly represent a first step to pave the way toward effective and actionable CBM systems, powered by ML capabilities. Future works will mostly concern the extension of the approach to contemplate the decay of a larger number of components, which drive the maintenance needs of the propulsion plant; moreover, validation with data, gathered in the field, will also allow to validate the ground-truth compliancy of the estimation with real-world evidences. Such activities will open the door to further evolutions of the proposed CBM approach, so as to predict, with a reasonable advance, potential breakdowns without requiring a preliminary and propaedeutic labeling activity by the operators (i.e. exploiting an unsupervised method[73]).

References

 1. British Standard. British Standard Glossary of Maintenance Management Terms in Terotechnology. British Standard Institution, London, 1984.
 2. Kothamasu R and Huang SH. Adaptive Mamdani fuzzy model for condition-based maintenance. *Fuzzy Set Syst* 2007; 158(24): 2715–2733.
 3. Widodo A and Yang BS. Support vector machine in machine condition monitoring and fault diagnosis. *Mech Syst Signal Pr* 2007; 21(6): 2560–2574.
 4. Mobley RK. *An introduction to predictive maintenance.* USA: Butterworth-Heinemann, 2002.
 5. Peng Y, Dong M and Zuo MJ. Current status of machine prognostics in condition-based maintenance: a review. *Int J Adv Manuf Tech* 2010; 50(1–4): 297–313.
 6. Yam R, Tse P, Li L, et al. Intelligent predictive decision support system for condition-based maintenance. *Int J Adv Manuf Tech* 2001; 17(5): 383–391.
 7. Budai G, Huisman D and Dekker R. Scheduling preventive railway maintenance activities. *J Oper Res Soc* 2005; 57(9): 1035–1044.
 8. Jimenez -Redondo N, Bosso N, Zeni L, Minardo A, Schubert F, Heinicke F and Simroth A, Automated and Cost Effective Maintenance for Railway (ACEM -Rail), Procedia - Social and Behavioral Sciences, Volume 48, 2012, Pages 1058-1067, ISSN 1877-0428, http://dx.doi.org/10.1016/j.sbspro.2012.06.1082.
 9. Murthy DP and Kobbacy KAH. *Complex system maintenance handbook.* USA: Springer, 2008.
10. Budai-Balke G. *Operations research models for scheduling railway infrastructure maintenance.* Netherlands: Rozenberg Publishers, 2009.
11. Stamatis DH. *Failure mode and effect analysis: FMEA from theory to execution.* USA: ASQ Press, 2003.
12. McNamara D, Cunningham A, Jenkinson I and Wang J, *Proceedings of the Institution of Mechanical Engineers, Part M: Journal of Engineering for the Maritime Environment*, published online 14 November 2013, DOI: 10.1177/1475090213509609.
13. Al-Najjar B. The lack of maintenance and not maintenance which costs: a model to describe and quantify the impact of vibration-based maintenance on company's business. *Int J Prod Econ* 2007; 107(1): 260–273.
14. Liyanage JP and Kumar U. Towards a value-based view on operations and maintenance performance management. *J Qual Maint Eng* 2003; 9(4): 333–350.
15. Sebastiani L, Pescetto A and Ambrosio L (FINCANTIERI S.p.A. - B.U. Offshore) Offshore Mediterranean Conference and Exhibition, 20–22 March, Ravenna, Italy, 2013, Offshore Mediterranean Conference.
16. Urban LA. Parameter selection for multiple fault diagnostics of gas turbine engines. *J Eng P* 1975; 97(2): 225–231.
17. Palmé T, Breuhaus P, Assadi M, et al. New Alstom monitoring tools leveraging artificial neural network technologies. In: *Turbo expo: turbine technical conference and exposition*, ASME and Alstom Technology Ltd., Vancouver, British Columbia, Canada, June 6–10, 2011.
18. Volponi AJ, DePold H, Ganguli R, et al. The use of Kalman filter and neural network methodologies in gas turbine performance diagnostics: a comparative study. *J Eng Gas Turb Power* 2003; 125(4): 917–924.
19. Urban LA. Gas path analysis applied to turbine engine condition monitoring. *J Aircraft* 1973; 10(7): 400–406.
20. Vapnik VN. *Statistical learning theory.* USA: Wiley, 1998.
21. Györfi L. *A distribution-free theory of nonparametric regression.* USA: Springer, 2002.
22. Boriana L. Milenova, Joseph S. Yarmus, and Marcos M. Campos. 2005. SVM in oracle database 10g: removing the barriers to widespread adoption of support vector machines. In Proceedings of the 31st international conference on Very large data bases Trondheim, Norway from August 30 to September 2, 2005 (VLDB '05). VLDB Endowment 1152-1163.
23. Altosole M, Benvenuto G and Campora U. Numerical modelling of the engines governors of a CODLAG propulsion plant. In: *Proceedings of the 10th international conference on marine sciences and technologies*, Varna, Bulgaria, 2010.
24. Altosole M, Benvenuto G, Figari M, et al. Real-time simulation of a COGAG naval ship propulsion system. *Proc IMechE, Part M: J Engineering for the Maritime Environment* 2009; 223(1): 47–62.
25. Benvenuto G and Campora U. A gas turbine modular model for ship propulsion studies. In: *Symposium on high speed marine vehicles, Proceedings of the 7th Symposium on High Speed Marine Vehicles,* Naples, Italy, 21-23 September 2005.
26. Benvenuto G and Campora U. Performance prediction of a faulty marine diesel engine under different governor settings. In: *International conference on marine research and transportation, 2nd International Conference on Marine Research and Transportation,* Naples, Italy, 28-30 June 2007.
27. Saravanamuttoo H, Rogers GF and Cohen H. *Gas turbine theory.* Pearson Education, September 2008, ISBN10: 0132224372.
28. Figari M and Altosole M. Dynamic behaviour and stability of marine propulsion systems. *Proc IMechE, Part M: J Engineering for the Maritime Environment* 2007; 221(4): 187–205.
29. Campora U and Figari M. Numerical simulation of ship propulsion transients and full-scale validation. *Proc IMechE, Part M: J Engineering for the Maritime Environment* 2003; 217(1): 41–52.
30. Altosole, M., Figari, M. and D'Arco, S. Design and optimisation of propulsion systems by dynamic numerical simulation, *Proceedings of International conference on ship and shipping research (NAV),* 24–27 June 2003.
31. Altosole, M., Benvenuto, G., and Campora, U. Marine gas turbine propulsion system simulation: Comparison of different approaches. In: *Proceedings of the International Conference on Ship and shipping research (NAV 2006),* Genova, Italy, June 2006.
32. Bruzzone M. *Decadimento delle prestazioni degli impianti di propulsione navale con elica mossa da turbine a gas.* Master's Thesis, University of Genova, Genova, 2011.
33. Tarabrin A, Schurovsky V, Bodrov A, et al. An analysis of axial compressor fouling and a blade cleaning method. *J Turbomach* 1998; 120(2): 256–261.
34. Meher-Homji, C.B., Focke, A. B., and Wool dridge, M. B., 1989, "Fouling of Axial Flow Compressors" - Causes, Effects, Detection, and Control, In: *Proceedings of the Eighteenth Turbomachinery Symposium,* Turbomachinery

Laboratory, Texas A & M University, College Station, Texas, pp. 55–76.

35. Aker GF and Saravanamuttoo HIH. Predicting gas turbine performance degradation due to compressor fouling using computer simulation techniques. *J Eng Gas Turb Power* 2006; 111(2): 343–350.

36. Martelli M. PhD Thesis, Marine Propulsion Simulation : Methods and Results, Marine Propulsion Simulation : Methods and Results, University of Genova, Genova, 2014.

37. Cristianini N and Shawe-Taylor J. *An introduction to support vector machines and other kernel-based learning methods*. UK, Cambridge University Press, 2000.

38. Smola AJ and Schölkopf B. A tutorial on support vector regression. *Stat Comput* 2004; 14(3): 199–222.

39. Lee WS, Bartlett PL and Williamson RC. The importance of convexity in learning with squared loss. *IEEE T Inform Theory* 1998; 44(5): 1974–1980.

40. Lugosi G and Zeger K. Nonparametric estimation via empirical risk minimization. *IEEE T Inform Theory* 1995; 41(3): 677–687.

41. Tikhonov A and Arsenin VY. *Methods for solving ill-posed problems*. Moscow: Nauka, 1979.

42. Engl HW, Hanke M and Neubauer A. *Regularization of inverse problems*. Netherlands, Springer, 1996.

43. Wolpert DH and Macready WG. No free lunch theorems for optimization. *IEEE T Evolut Comput* 1997; 1(1): 67–82.

44. Evgeniou T, Micchelli CA and Pontil M. Learning multiple tasks with Kernel methods. *J Mach Learn Res* 2005; 6: 615–637.

45. Pan SJ and Yang Q. A survey on transfer learning. *IEEE T Knowl Data En* 2010; 22(10): 1345–1359.

46. Zhang W, Liu X, Ding Y, et al. Multi-output LS-SVR machine in extended feature space. In: *IEEE international conference on computational intelligence for measurement systems and applications (CIMSA)*, Tianjin, IEEE, 2-4 July 2012.

47. Baldassarre L, Rosasco L, Barla A, et al. Multi-output learning via spectral filtering. *Mach Learn* 2012; 87(3): 259–301.

48. Pollard D. Empirical processes: theory and applications. In: *NSF-CBMS regional conference series in probability and statistics*, Institute of Mathematical Statistics, Vol. 2, Empirical Processes: Theory and Applications, 1990, pp. 1-86. Available at: http://www.jstor.org/stable/4153175.

49. Caponnetto A and De Vito E. Optimal rates for the regularized least-squares algorithm. *Found Comput Math* 2007; 7(3): 331–368.

50. Schölkopf B, Herbrich R and Smola AJ. A generalized representer theorem. In Computational learning theory pp. 416-426. Springer Berlin Heidelberg. Amsterdam, The Netherlands, July 16 to 19, 2001.

51. Schölkopf B. The kernel trick for distances. In: *Neural information processing systems*, Vancouver, British Columbia, Canada, December, 2001: 301–307.

52. Keerthi SS and Lin CJ. Asymptotic behaviors of support vector machines with Gaussian kernel. *Neural Comput* 2003; 15(7): 1667–1689.

53. Young DM. *Iterative solution of large linear systems*. USA, Dover Publications, 2003.

54. Poggio, T., Mukherjee, S., Rifkin, R., Raklin, A., & Verri, A. .Uncertainty in geometric computations. USA, Kluwer Academic Publishers, 2002, 22: 131–141.

55. Boyd SP and Vandenberghe L. *Convex optimization*. UK, Cambridge University Press, 2004.

56. Shawe-Taylor J and Sun S. A review of optimization methodologies in support vector machines. *Neurocomputing* 2011; 74(17): 3609–3618.

57. Keerthi SS, Shevade SK, Bhattacharyya C, et al. Improvements to Platt's SMO algorithm for SVM classifier design. *Neural Comput* 2001; 13(3): 637–649.

58. De Vito E, Caponnetto A and Rosasco L. Model selection for regularized least-squares algorithm in learning theory. *Found Comput Math* 2005; 5(1): 59–85.

59. Bartlett PL, Boucheron S and Lugosi G. Model selection and error estimation. *Mach Learn* 2002; 48(1–3): 85–113.

60. Anguita D, Ghio A, Oneto L, et al. In-sample and out-of-sample model selection and error estimation for support vector machines. *IEEE T Neural Networ Learn Syst* 2012; 23(9): 1390–1406.

61. Guyon I, Saffari A, Dror G, et al. Model selection: beyond the Bayesian/frequentist divide. *J Mach Learn Res* 2010; 11: 61–87.

62. Kohavi R. A study of cross-validation and bootstrap for accuracy estimation and model selection. In: *International joint conference on artificial intelligence*, Palais de Congres Montreal, Quebec Canada, USA, Morgan Kaufmann, August 20–25, 1995.

63. Anguita D, Ghio A, Ridella S, et al. K-fold cross validation for error rate estimate in support vector machines. In: *International conference on data mining*, USA, Morgan Kaufmann Publishers Inc., 2009. Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2.

64. Dietterich TG. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Comput* 1998; 10(7): 1895–1923.

65. Anguita D, Ghelardoni L, Ghio A, et al. The K in K-fold cross validation. In: *European symposium on artificial neural networks*, 20th European Symposium on Artificial Neural Networks Bruges, Belgium, April 25-26-27, 2012.

66. Hsu CW, Chang CC and Lin CJ. A practical guide to support vector classification, http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf (2009).

67. Bache K and Lichman M. UCI machine learning repository, http://archive.ics.uci.edu/ml (2013).

68. Narula SC and Wellington JF. Prediction, linear regression and the minimum sum of relative errors. *Technometrics* 1977; 19(2): 185–190.

69. Chen K, Guo S, Lin Y, et al. Least absolute relative error estimation. *J Am Stat Assoc* 2010; 105(491): 1104–1112.

70. Aupetit M. Nearly homogeneous multi-partitioning with a deterministic generator. *Neurocomputing* 2009; 72(7): 1379–1389.

71. Coraddu A and Figari M. Ship electric propulsion: analyses through modeling and simulation marine 2011. In: IV Computational methods in marine engineering, MARINE 2011, Lisbon, Portugal, 28-30 September 2011

72. Li Z, Yan X and Peng Z. Ship electric propulsion with a sensorless permanent magnet synchronous motor: a simulation study. *Proc IMechE, Part M: J Engineering for the Maritime Environment* 2012; 226(4): 378–386.

73. Weber M, Welling M and Perona P. *Unsupervised learning of models for recognition*. GErmany, Springer, 2000.