Modeling Blade-Pitch Actuation Power Use in Wind Turbines

Aoife Henry[1], Manuel Pusch[1,2], Lucy Pao[1]

***Abstract*— Estimating the levelized cost of energy (LCOE) of a wind turbine is useful for performing a cost-benefit analysis of potential designs. The power consumed by blade-pitch actuation is an often neglected, but nontrivial factor in LCOE estimation. The peak power consumption determines the required rating of the actuation motors and the mean power consumption impacts the net annual energy production (nAEP) of the turbine. The closed-loop blade-pitch actuation and the power consumed by its motors are complex functions of the wind field disturbance and internal turbine states. They can only be predicted well with reasonably high-fidelity and computationally expensive simulations or field tests. We present an alternative approach to modeling these signals using the Sparse Identification of Nonlinear Dynamics with Control (SINDyC) methodology. It is computationally tractable to generate these models for large datasets and to efficiently evaluate the required pitching power for a given wind field. Furthermore, the models provide intuition as to how the wind disturbance and blade pitch contribute to the signal dynamics. By generating a closed-form dynamic state equation for the blade-pitch actuation and an algebraic equation for the blade-pitch motor power, we can efficiently predict the mean and maximum power required for a given turbulent wind field and turbine design. The model is trained and validated using data generated from the open-source aero-servo-hydro-elastic wind turbine simulation tool OpenFAST.**

I. INTRODUCTION

Wind energy, produced primarily by horizontal axis wind turbines (HAWTs) offers enormous potential for renewable energy generation as part of the clean energy transition necessary to fulfill demand for increasing quantities of affordable and reliable energy with lower associated carbon emissions. The widespread and accelerated installation of wind farms is strongly influenced by the levelized cost of energy (LCOE) metric, which quantifies the cost of energy per kilowatt-hour (kWh) produced by a wind farm.

The LCOE contribution of a single turbine in a wind farm over the course of a year is the sum of the annual costs associated with that turbine divided by the net annual energy production (nAEP). nAEP is defined as the difference between the annual energy production (AEP) and the energy consumed by the actuators. In the current study, we neglect the energy consumed by all actuators except for the blade-pitch actuation system and denote that annual energy consumption as ΔAE_β. Extensions to include the other actuators is a subject of future work. The LCOE is defined as:

$$LCOE = \frac{CAPEX \cdot FCR + OPEX}{AEP - \Delta AE_\beta} \tag{1}$$

[1] University of Colorado Boulder
[2] Munich University of Applied Sciences

where $CAPEX$ [\$/kW] is the capital expenditure, FCR [$\%$/year] is the fixed charge rate that annualizes the upfront project capital cost, $OPEX$ [\$/kW-year] is the annualized operation and maintenance expenditure [1], AEP [kW-year] is the total energy produced by the wind turbine over the course of a year and ΔAE_β [kW-year] is the total energy consumed by the blade pitch actuation motors over the same time period.

We consider a variable-speed HAWT [2] with collective pitch control, whereby identical blade-pitch signal commands are issued to each blade. This allows us to reduce our analysis to the study of a single blade's pitching behavior. However, the modeling procedure could easily be extended to consider the case of individual pitch control (IPC), where each blade receives a distinct pitching command from the controller.

The control scheme of a wind turbine is typically divided into four distinct *control regions* for different wind speeds as shown in Fig. 1. Of interest in this work is *Region 3* (R3), where the wind speed is greater than the rated value. In R3, the blade pitch controller is activated to regulate the generator speed to its rated value and the generator torque is controlled to maintain a constant power output [3].

The *closed-loop* HAWT model we consider is based on a particular iteration of the Segmented Ultralight Morphing Rotor (SUMR) wind turbine model [4]. Our goal is to model the closed-loop dynamics of the blade-pitch angle β, and the power P_β that its motors consume in R3.

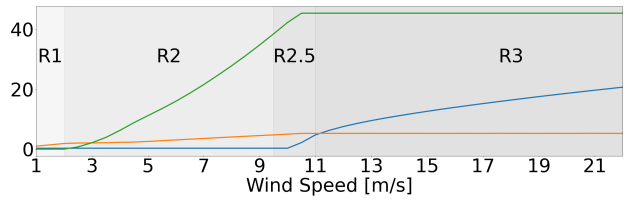

Fig. 1: Wind turbine operating regions, including the steady-state blade-pitch angle β [deg] (blue), generator speed ω_g [rpm] (orange), and generator torque τ_g [MN·m] (green) curves for the SUMR wind turbine

Blade-pitch actuation involves adjusting the angle of attack of each of the blades as a means of controlling wind turbine rotational speed and fatigue loading. The actuation system at each blade is powered by a motor that consumes a time-varying power $P_\beta(k)$, where k denotes the discrete time-step. Of particular interest are the values of *mean*, $\bar{P}_\beta$, and *peak*, $P^\star_\beta$, power consumed by the motors over a given time period. $\bar{P}_\beta$ serves as a predictor of the power consumption

during normal wind conditions and thus determines ΔAE_β in (1) and the average fatigue loads we can expect on the turbine components. $P^\star_\beta$ determines what power rating the motors require for extreme wind conditions and the ultimate loads we can expect, and thus contributes to $CAPEX$ in (1). The relative cost of the wind turbine pitching system is estimated to be as high as 5.5% [5], and the motor power rating significantly affects the investment cost of the turbine.

Machine learning (ML) techniques have been applied to wind energy systems with a variety of objectives. Stetco et al. present a review of ML methods for wind turbine condition monitoring for prognosis, whereby a predictor model for failures in turbine components is learned from data [6]. Clifton et al. [7] and Pei and Li [8] employ ML techniques to learn the *power curve* of a wind turbine (i.e. the power generated for different mean wind speeds) from data. Neither of the aforementioned works consider the power consumed by wind turbine actuators and the resulting impact on the *net* power generated by a wind turbine. Kane presents and validates a ML-based controller for IPC of floating offshore wind turbines [9]. It is not, however, applicable to learning a pre-defined control law for an operational wind turbine.

The cost of predicting the power consumed by a blade-pitch actuation system for a given wind-field time-series using medium- to high-fidelity simulation tools such as OpenFAST is computationally expensive, requiring several hours for a 10-minute simulation. Such an approach is not conducive to efficient optimization of potential new turbine designs. To address this challenge, this work develops a deterministic, closed-form model of the power consumed by the blade-pitch actuation as a function of the wind field. Such a model allows us to swiftly generate a time-series of $P_\beta(k)$ for a given turbine design and wind field and thus evaluate the mean and peak power required by the actuation motors for the particular wind case and turbine design. The main contributions of this paper are as follows:

- Development of a dynamic model of the blade-pitch actuation as a function of the wind disturbance. based on the Sparse Identification of Nonlinear Dynamics with Control (SINDyC) methodology
- Development of an algebraic model of the power consumed by the blade-pitch actuation system based on the SINDyC methodology.
- Validation of the above two models using data from OpenFAST simulations.

The rest of the paper is organized as follows. Section II describes the SINDyC modeling approach implemented. Section III describes the datasets and modeling parameters used for training and validation of the model. Section IV presents the results of the models generated and assesses their accuracy. Section V discusses the insights gained from the results and provides suggestions for future work.

II. METHODOLOGY

A. Modeling

We develop two models in this work: a) a dynamic state equation for the discrete-time closed-loop blade-pitch control input $\beta(k)$ as a function of the previous filtered horizontal wind speed $\hat{u}_x(k-1)$ and its first-derivative $\dot{\hat{u}}_x(k-1)$, and b) an algebraic equation for the power $P_\beta(k)$ consumed by the blade-pitch motor as a function of the blade-pitch angle $\beta(k)$ and its derivatives $\dot{\beta}(k)$ and $\ddot{\beta}(k)$, where k represents the discrete time step, see Fig. 2.

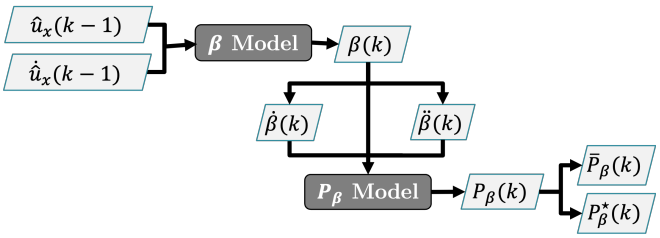

Fig. 2: Block Diagram of Model Structure

1) Closed-Loop Blade-Pitch Dynamics, β: In this work, we employ Discrete-Time SINDyC [10] to model the dynamics of the blade-pitch actuation, β. SINDyC is essentially a regularized linear regression method for identifying the dynamics of nonlinear systems with control inputs from measured data and relies on the premise that most real systems will have relatively few dominant terms in their underlying governing equations. It is thus a *sparsity-promoting* method in that the trade-off between accuracy and parsimony of the model is integrated into the solution and controlled by a *thresholding* parameter. Given a large library of candidate functions, the algorithm selects relatively few to represent the dynamics of the system.

To formulate the SINDyC problem, we measure K snapshots of the state $\boldsymbol{x} \in \mathbb{R}^n$ and the control input $\boldsymbol{u} \in \mathbb{R}^m$ in time and arrange these into two matrices $\boldsymbol{X} \in \mathbb{R}^{n\times K}$ and $\boldsymbol{U} \in \mathbb{R}^{m\times K}$. We also construct the subsequent state matrix $\boldsymbol{X}^+$ via direct measurement. The full set of measurement data matrices are:

$$\boldsymbol{X} = [\boldsymbol{x_1} \quad \boldsymbol{x_2} \quad \dots \quad \boldsymbol{x_K}] \tag{2a}$$
$$\boldsymbol{X}^+ = [\boldsymbol{x_1^+} \quad \boldsymbol{x_2^+} \quad \dots \quad \boldsymbol{x_K^+}] \tag{2b}$$
$$\boldsymbol{U} = [\boldsymbol{u_1} \quad \boldsymbol{u_2} \quad \dots \quad \boldsymbol{u_K}] \tag{2c}$$

The library of candidate nonlinear functions $\boldsymbol{\Theta}(\boldsymbol{X}, \boldsymbol{U}) \in \mathbb{R}^{N\times K}$ is also constructed, where each row corresponds to one of N nonlinear candidate functions of $\boldsymbol{x}$ terms, $\boldsymbol{u}$ terms, and cross-terms in $\boldsymbol{x}$ and $\boldsymbol{u}$; and each column corresponds to the evaluation of the library of functions at a particular snapshot in time.

SINDyC assumes the dynamical system to be a linear combination of the candidate functions:

$$\boldsymbol{X}^+ = \boldsymbol{\zeta}\boldsymbol{\Theta}(\boldsymbol{X}, \boldsymbol{U}) \tag{3}$$

where $\boldsymbol{\zeta} \in \mathbb{R}^{n\times N}$ is a matrix of coefficients representing the relative contribution of each candidate function to the model, where the ith row gives the coefficients modeling the dynamics of the ith state. Reformulating this notation to a per-state basis, the N coefficients corresponding to the ith state in the vector $\boldsymbol{x}$ are given by $\boldsymbol{\zeta_i} \in \mathbb{R}^{1\times N}$.

To find the coefficients, $\boldsymbol{\zeta_i}$, best describing the state dynamics for a given library of candidate functions, we can solve the Sparse Relaxed Regularized Regression (SR3) [11] optimization problem for each state x_i. SR3 is a sparsity-promoting regression methodology applicable to SINDyC that has been shown to:

a) be superior with respect to modeling errors and problem conditioning when compared to non-relaxed counterparts (e.g. LASSO [12]),
b) facilitate extremely fast optimization algorithms for both convex and non-convex problems, and
c) apply well to composite regularization functions.

In SR3, an *auxiliary* variable w_i is forced to be close to a transformation of the coefficients $\boldsymbol{\zeta_i}$, $\boldsymbol{C_i}\hat{\boldsymbol{\zeta}}_{\boldsymbol{i}}^{\boldsymbol{T}}$, by introducing the normed error into the objective function. Solving for the dynamics of the ith state, we then have:

$$\begin{aligned}\boldsymbol{\zeta_i} = \underset{\hat{\boldsymbol{\zeta}}_{\boldsymbol{i}}, w_i}{\arg\min} &\frac{1}{2}\|\boldsymbol{X}_{\boldsymbol{i}}^{+} - \hat{\boldsymbol{\zeta}}_{\boldsymbol{i}}\boldsymbol{\Theta_i}(\boldsymbol{X},\boldsymbol{U})\|_2^2 + \lambda\mathcal{L}(w_i) \\ &+ \frac{1}{2\nu}\|\boldsymbol{C_i}\hat{\boldsymbol{\zeta}}_{\boldsymbol{i}}^{\boldsymbol{T}} - w_i\|_2^2\end{aligned} \tag{4}$$

where $\boldsymbol{X}_{\boldsymbol{i}}^{+} \in \mathbb{R}^{1\times K}$ represents the rows corresponding to the ith state, $\boldsymbol{\zeta_i} \in \mathbb{R}^{1\times N}$ is the ith row of $\boldsymbol{\zeta}$, $\mathcal{L}(\cdot)$ is the *regularization* function, $\boldsymbol{C_i} \in \mathbb{R}^{1\times N}$ is the *relaxation* matrix, $\lambda \in \mathbb{R}_+$ is the *threshold* parameter that controls the trade-off between model accuracy and parsimony, and ν is the *relaxation* parameter that controls the trade-off between regularity of the problem and proximity of the solution to the true optimal. The purpose of the $\lambda\mathcal{L}(w_i)$ term is then to promote sparsity in the variable, w_i that is forced to be close to a transformation of the coefficients $\boldsymbol{C_i}\hat{\boldsymbol{\zeta}}_{\boldsymbol{i}}^{\boldsymbol{T}}$.

2) Blade-Pitch Motor Power, P_β: The power P_β consumed by the blade-pitch motor is an algebraic function of the blade-pitch dynamics and the turbine specifications at each time step. We employ a static variation of SINDyC to model P_β as a function of the blade pitch and its first two derivatives for a given turbine model. By 'static', we are assuming that P_β has no associated dynamics, and that its value at time-step k is a function of $\dot{\beta}(k)$ and $\ddot{\beta}(k)$.

From the raw blade-pitch time-series data, the first and second blade-pitch derivatives, $\dot{\beta}(k)$ and $\ddot{\beta}(k)$, respectively, are computed via the second-order finite-difference method. The 'true' power consumed by the blade-pitch motor is not provided directly by the data and is computed as $P_\beta = M\dot{\beta}$, where M [kN·m] is the blade torsional moment. While this may result in negative values of the power, using this form of P_β in the training data is found to result in significantly more accurate models when compared to those generated from the absolute values.

We construct the matrices $\boldsymbol{X}$ and $\boldsymbol{U}$ as in (2) and the library of candidate nonlinear functions $\boldsymbol{\Theta}(\boldsymbol{X},\boldsymbol{U})$. Linear regression assumes the algebraic relation to be a linear combination of the candidate functions:

$$\boldsymbol{X} = \boldsymbol{\zeta}\boldsymbol{\Theta}(\boldsymbol{X},\boldsymbol{U}) \tag{5}$$

Again, we use the SR3 method to find a solution to (5):

$$\begin{aligned}\boldsymbol{\zeta_i} = \underset{\hat{\boldsymbol{\zeta}}_{\boldsymbol{i}}}{\arg\min} &\frac{1}{2}\|\boldsymbol{X_i} - \hat{\boldsymbol{\zeta}}_{\boldsymbol{i}}\boldsymbol{\Theta_i}(\boldsymbol{X},\boldsymbol{U})\|_2^2 + \lambda\mathcal{L}(w_i) \\ &+ \frac{1}{2\nu}\|\boldsymbol{C_i}\hat{\boldsymbol{\zeta}}_{\boldsymbol{i}}^{\boldsymbol{T}} - w_i\|_2^2\end{aligned} \tag{6}$$

B. Time-Series & Library Ensembling

To improve SINDyC performance in this work, we employ *time-series ensembling*, which generates multiple models for subsets of the training data $\boldsymbol{X}$ and $\boldsymbol{U}$, and *library ensembling*, which generates multiple models for subsets of the candidate function library. We then compute the median of the coefficients for each candidate function over all models generated for the final model. A secondary advantage of this approach is that we can inspect the variance of the coefficients corresponding to different library functions. This tells us how robust those coefficients are to different sets of measurements. We compute the *normalized variance* for each coefficient by dividing the variance of the coefficients generated for each model by the mean value.

C. Maximum-Absolute Scaling

Each candidate library term formed from the training data, $\boldsymbol{\Theta_i}(\boldsymbol{X},\boldsymbol{U})$ for the ith state being modeled, is scaled using the *maximum-absolute* method such that its values have a range of $[-1,1]$, which improves the conditioning of the optimization problem (4). The scaling factors used are the maximum-absolute values for each candidate library term over all the training data, and are also used to scale test data inputs when we simulate the learned model.

D. Defining Candidate Library Functions

Given intuition about the control schemes and dynamics describing the behaviour of a closed-loop wind turbine system, we can try to derive suitable candidate functions for our SINDyC model. The candidate functions map the features listed in Table I to terms that may be included in the final model. Distinct candidate function libraries are derived to model both the dynamic blade pitch and algebraic blade pitch motor power functions.

TABLE I: Model Features

Feature	Description	Unit
β	Blade 1 pitch angle	rad
$\dot{\beta}$	First time-derivative of β	rad/s
$\ddot{\beta}$	Second time-derivative of β	rad/s^2
$\hat{u}_x - \bar{u}_x$	Deviation of filtered horizontal wind speed at hub-height from mean value	m/s
$\dot{\hat{u}}_x$	First time-derivative of $\hat{u}_x$	m/s^2
P_β	Power used by the Blade 1 pitch motor	MW

1) Blade Pitch: We assume that the dynamics of β, for fixed turbine specifications, are described by a function of some of the features given in Table I and can be expressed as $\beta(k) = f(\hat{u}_x(k-1) - \bar{u}_x, \dot{\hat{u}}_x(k-1))$. We consider the *filtered* wind speed, $\hat{u}_x$, as the raw measurement data contains a high degree of noise. To this end, we filter the data with a second-order low-pass filter. For our set of candidate functions for the β model, we consider all possible polynomials, including

interaction between any two different variables, of degree 3 of specific variables and a constant term:

$$a^p b^q \, \forall a, b \in \{\hat{u}_x - \bar{u}_x, \dot{\hat{u}}_x(k)\}; p, q \in \{0, 1, 2, 3\} \quad (7)$$

The current blade-pitch value, $\beta(k)$, is not included in this model, as it is found to be considerably more likely to be unstable and to generate unbounded values when used in simulations.

2) Blade-Pitch Motor Power: The power consumed by the blade-pitch motor can be directly computed as $P_\beta = M\dot{\beta}$, where M [kN·m] is the blade torsional moment and $\dot{\beta}$ [rad/s] is the first time derivative of the blade-pitch signal. We assume that, for fixed turbine specifications, P_β can be modeled as an algebraic function of the first two derivatives of the blade-pitch as $P_\beta(k) = f(\dot{\beta}(k), \ddot{\beta}(k))$, as the blade torsional moment, M, may also be a function of $\dot{\beta}$ and $\ddot{\beta}$. For the P_β candidate function library, we consider all possible polynomials up to degree of 2 of $\dot{\beta}, \ddot{\beta}$, where $\dot{\beta}$ must be included in every feature:

$$\{\dot{\beta}, \dot{\beta}^2, \dot{\beta}\ddot{\beta}\} \quad (8)$$

III. VALIDATION SETUP

To generate training and test datasets, we run closed-loop wind-turbine simulations on the SUMR [4] turbine model for 700 seconds with a sampling rate of 80 Hz, with turbulent wind profiles generated for 100 different seeds with an Extreme Turbulence Model (ETM) and a mean wind speed of 14 m/s using OpenFAST [13]. SUMR is a 3-bladed, 25 MW turbine with a radius of 171.75 m, rated generator speed of 5.2579 rpm and rated wind speed of 10 m/s. The horizontal wind speed at hub-height signal, u_x, is filtered with a second-order low-pass filter to generate the filtered time-series, $\hat{u}_x$. The first 62.5 seconds of data are discarded from each dataset to exclude the initial transient response, leaving 100 time-series datasets with a duration of 637.5 seconds (or, 5100000 data points). We use 80 of the datasets as training datasets and the remaining 20 as test datasets.

We choose $\boldsymbol{C} = \boldsymbol{I}$ as the relaxation matrix, which forces the auxiliary variable to be close to the coefficients, and $\mathcal{L}(\cdot) = \ell_0$ as the regularization function, which penalizes the number of nonzero coefficients. For the time-series ensembling, we generate 20 models with sub-samples of size equal to that of a single dataset. The results are that on average, approximately 60% of the total training data is seen by at least one of these generated models. For the library ensembling, we generate 20 models for each of the above models, where 2 of the candidate functions are randomly selected to exclude from the library for each model. We do not consider library ensembling for the P_β model, since the candidate function library consists of only a few functions.

We conduct a parameter sweep by generating models with the SR3 algorithm for different values of the thresholding parameter, λ, and the relaxation parameter, ν, for the values $[10^{-8}, 10^{-6}, 10^{-4}, 10^{-2}, 10^{0}]$.

To evaluate the accuracy of model predictions compared to true data, we use the $R^2 = 1 - \frac{\sum_i (y_i - f_i)^2}{\sum_i (y_i - \bar{y})^2}$ metric, where

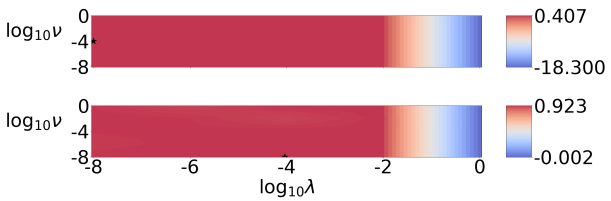

Fig. 3: R^2 vs. λ, ν for β Model (top) and P_β Model (bottom), where each black star denotes the maximum value of R^2 found over all parameter sets.

$\bar{y}$ is the mean of the true data, y_i is the ith true data point and f_i is the ith modeled data point.

We also evaluate the relative error between the true and predicted values of the mean, $\bar{P}_\beta$, and peak, $P^\star_\beta$, values of the blade-pitch motor power for each test time-series dataset as $\epsilon_j = \frac{f_j - y_j}{y_j} \times 100\%$, where y_j is the true mean/peak value for the jth test dataset and f_j is the corresponding value computed from the model for the same dataset.

Codes adapted from the open-source Python library, pySINDy [14], are used to generate the SINDyC models discussed, and are available on GitHub [15].

IV. MODELING RESULTS

A. Parameter Sweep

For each parameter set of λ and ν, models are generated for β and P_β and the R^2 metric is computed for each test dataset, for each of these parameterized models. Fig. 3 shows how the median of these R^2 scores over all test datasets varies with λ and ν for each dynamical model. We see that, in the case of the β model results, of the parameters tested, values of $\lambda = 10^{-8}, \nu = 10^{-4}$ achieve the greatest median score over all test datasets. In the case of the P_β model, values of $\lambda = 10^{-6}, \nu = 10^{-6}$ achieve the greatest median score. We can also see that higher values of λ, which will result in less terms included in the model, result in significantly poorer predictions for both models.

TABLE II: β SINDyC Candidate Function Ensemble Coefficient Median (m_{e,ζ_i}) and Normalized Variance ($\hat{\sigma}^2_{e,\zeta_i}$)

$\beta(k+1)$ Terms	m_{e,ζ_i}	$\hat{\sigma}^2_{e,\zeta_i}$
1	1.958e-01	3.614e-02
$(\hat{u}_x - \bar{u}_x)$	1.267e-01	2.425e-02
$(\hat{u}_x - \bar{u}_x)^2$	-5.048e-02	1.002e+00
$(\hat{u}_x - \bar{u}_x)^3$	3.518e-02	3.044e-01
$\dot{\hat{u}}_x$	-6.765e-03	3.080e-03
$(\hat{u}_x - \bar{u}_x)\dot{\hat{u}}_x$	5.197e-03	2.571e-03
$(\hat{u}_x - \bar{u}_x)^2\dot{\hat{u}}_x$	-4.979e-03	2.708e-02
$\dot{\hat{u}}_x^3$	-3.672e-03	3.680e-02
$P_\beta(k)$ Terms	m_{e,ζ_i}	$\hat{\sigma}^2_{e,\zeta_i}$
$\dot{\beta}$	-3.301e-02	3.567e-04
$\dot{\beta}^2$	-2.023e-02	1.774e-03
$\dot{\beta}\ddot{\beta}$	1.481e-03	1.431e-03

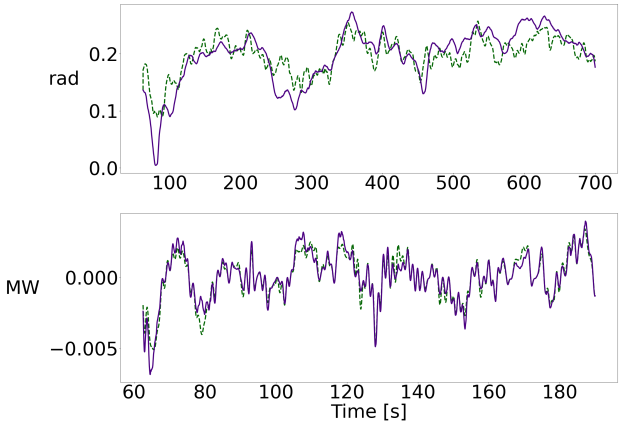

Fig. 4: True (purple, solid) & Modeled (green, dashed) Time-Series Simulation of a Test Dataset for the $\beta(k)$, $\lambda = 10^{-8}, \nu = 10^{-4}$ (top) and $P_\beta(k)$, $\lambda = 10^{-4}, \nu = 10^{-8}$ (bottom) Models

B. SINDyC Candidate Function Coefficients

Table II shows the relative ensemble median values, $m_{e,\boldsymbol{\zeta_i}}$, of the SINDyC coefficients. These values indicate how much each candidate function contributes to the models. The normalized variance, $\hat{\sigma}^2_{e,\boldsymbol{\zeta_i}}$, of the ensemble coefficients (the variance normalized by the mean) tells us how robust these coefficients are to models trained with slightly different input data or candidate function libraries. Note that since we apply scaling to the candidate function terms, it is valid to compare these coefficients for different terms. The candidate functions are listed in order of decreasing median contribution. We see that, for the β model, the terms including the deviation of the filtered wind speed from its mean value, $\hat{u}_x - \bar{u}_x$, have relatively high median but also high normalized variance values in the case of the terms raised to powers. This suggests that such terms contribute significantly to the model, but that its value may vary significantly for different training data and candidate functions. As for the P_β model, it seems that the $\dot{\beta}$ and $\dot{\beta}^2$ terms contribute the greatest to the model. Their corresponding low variance values suggest that we can rely on these coefficients to make predictions on new data.

C. Time-Series Match

Fig. 4 shows the true time-series of β and P_β from a selected test dataset compared to those generated by the respective models. Note that P_β is plotted over a subset of the time-series such that the fluctuations are visible. Variations in β and P_β over time seem to be captured well by both models. Each SINDyC simulation of 700 seconds requires 20 seconds of CPU time to run, compared to approximately 200 minutes for the equivalent OpenFAST simulation.

From this study, we have learned that the magnitude of P_β, which is of the order of 0.01 MW, is virtually negligible relative to the rated power of the turbine at 25 MW. The consequence of this is that the contribution of ΔAE_β, which is computed from the mean power $\bar{P}_\beta$ consumed, to $LCOE$ is not very significant. However, the peak power $P^\star_\beta$ is

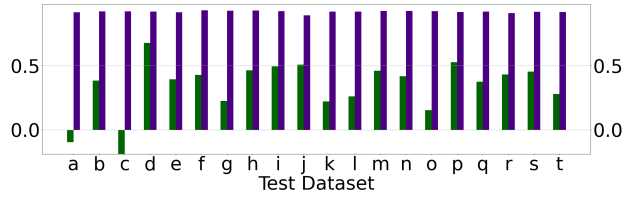

Fig. 5: R^2 for each Test Dataset for the $\beta(k)$, $\lambda = 10^{-8}, \nu = 10^{-4}$ (green, left axis) and $P_\beta(k)$, $\lambda = 10^{-4}, \nu = 10^{-8}$ (purple, right axis) Models

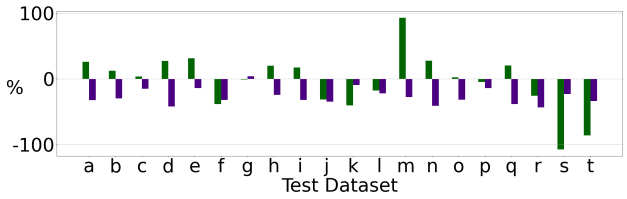

Fig. 6: Relative Error, ϵ, between True & Modeled Values of the mean, $\bar{P}_\beta$ (green), and peak, $P^\star_\beta$ (purple) power for all Test Datasets, $\lambda = 10^{-4}, \nu = 10^{-8}$

nevertheless an important factor in the overall cost of the system due to its contribution to the required rating of the actuation motors and thus to the $CAPEX$ in Eqn. (1).

D. R^2 Evaluation

Fig. 5 shows R^2 for the β and P_β models, for the true time-series values of each test dataset compared to those generated by the respective models. Clearly there are some outlying datasets for the β model, namely dataset a and c, that perform poorly relative to others, but generally the score hovers just above 0.4 in the case of the β model (green bars) and 0.9 in the case of the P_β model (purple bars).

Fig. 6 compares the true and modeled mean, $\bar{P}_\beta$ (green), and peak, $P^\star_\beta$ (purple), blade-pitch motor power, as measured over each test dataset for the $\lambda = 10^{-4}, \nu = 10^{-8}$ case. While the predictions of $\bar{P}_\beta$ err up to $\pm 100\%$ in some cases, the prediction errors for $P^\star_\beta$ are within $[-40, 4]\%$ over all datasets, with a mean error of approximately 25%. Given the overall very small values of mean power, the estimation errors in mean power are not of concern.

V. CONCLUSIONS AND FUTURE WORK

The SINDyC models generated in this work produce satisfactory predictions when compared with the data from OpenFAST simulations. The modeled values of the peak blade-pitch motor power $P^\star_\beta$ generally match the simulated values within 25%. The general trends of the modeled β and P_β values seem to correspond particularly well to those in the time-series simulations. Once the computational effort of running OpenFAST simulations to generate training data has been completed, the generated models can be used in iterative optimization algorithms with negligible computational overhead. The median values found by the ensembling method are indicative of the dominant terms in each case and the

normalized variance reflects the reliability of these values across different datasets and candidate function libraries.

Results could likely be improved by employing a larger training dataset, by including a more formal forward or backward feature selection process to find the best subset of candidate library functions, or by optimizing the tuning parameters, λ and ν, using a cross-validation approach. Additionally, more complex, albeit possibly more accurate models could be developed using auto-regressive inputs (i.e. from previous time steps).

As for future work, ultimately we would like to generate a model of the blade-pitch motor power consumption which is purely a function of the turbine specifications, the control objectives which dictate the blade-pitch actuation of that turbine, and the characteristics of the wind disturbance expected, such that we could efficiently compute the approximate power consumed and peak power required by the blade pitch actuation system over a given period of time and therefore estimate the impact of this actuation on the LCOE of the total wind farm without needing to run expensive simulations. Next steps would be to extend the input space with the set of turbine parameters (e.g. the rotor diameter, the tower height, the number of blades, the rated generator power, the rated wind speed, the rated generator speed, etc.), characteristics of the wind field (e.g. different mean wind speeds, directions, turbulence intensities, wind shear values, etc.) and the control objectives that determine how the blade-pitch actuation system behaves. Future work also includes generating a similar model for the costs of replacing the bearings of the blade-pitch actuation system as well as for IPC and yaw angle actuation and the associated power consumption.

REFERENCES

[1] M. Shields, P. Beiter, J. Nunemaker, A. Cooperman, and P. Duffy, "Impacts of turbine and plant upsizing on the levelized cost of energy for offshore wind," *Applied Energy*, vol. 298, p. 117 189, 2021.

[2] L. Y. Pao and K. E. Johnson, "A tutorial on the dynamics and control of wind turbines and wind farms," in *Proc. American Control Conf.*, 2009, pp. 2076–2089.

[3] N. J. Abbas, D. S. Zalkind, L. Pao, and A. Wright, "A reference open-source controller for fixed and floating offshore wind turbines," *Wind Energy Science*, vol. 7, no. 1, pp. 53–73, 2022.

[4] A. S. E. Mendoza, D. T. Griffith, C. Qin, E. Loth, and N. Johnson, "Rapid approach for structural design of the tower and monopile for a series of 25 mw offshore turbines," in *J. Physics: Conf. Series*, IOP Publishing, 2022.

[5] M. Shan, *Load Reducing Control for Wind Turbines: Load Estimation and Higher Level Controller Tuning based on Disturbance Spectra and Linear Models.* BoD–Books on Demand, 2022.

[6] A. Stetco, F. Dinmohammadi, X. Zhao, V. Robu, D. Flynn, M. Barnes, J. Keane, and G. Nenadic, "Machine learning methods for wind turbine condition monitoring: A review," *Renewable Energy*, vol. 133, pp. 620–635, 2019.

[7] A. Clifton, L. Kilcher, J. Lundquist, and P. Fleming, "Using machine learning to predict wind turbine power output," *Env. Research Letters*, vol. 8, no. 2, p. 024 009, 2013.

[8] S. Pei and Y. Li, "Wind turbine power curve modeling with a hybrid machine learning technique," *Applied Sciences*, vol. 9, no. 22, p. 4930, 2019.

[9] M. B. Kane, "Machine learning control for floating offshore wind turbine individual blade pitch control," in *Proc. American Control Conf.*, 2020, pp. 237–241.

[10] E. Kaiser, J. N. Kutz, and S. L. Brunton, "Sparse identification of nonlinear dynamics for model predictive control in the low-data limit," *Proc. Royal Society A*, vol. 474, no. 2219, p. 20 180 335, 2018.

[11] P. Zheng, T. Askham, S. L. Brunton, J. N. Kutz, and A. Y. Aravkin, "A unified framework for sparse relaxed regularized regression: SR3," *IEEE Access*, vol. 7, pp. 1404–1423, 2018.

[12] R. Tibshirani, "Regression shrinkage and selection via the LASSO," *J. Royal Statistical Society: Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[13] *Working with OpenFAST documentation*, `https://ebranlard-openfast.readthedocs.io/en/doc/source/working.html`, [Online; accessed 20-August-2022], 2022.

[14] B. M. de Silva, K. Champion, M. Quade, J.-C. Loiseau, J. N. Kutz, and S. L. Brunton, "pySINDy: A python package for the sparse identification of nonlinear dynamics from data," *arXiv preprint arXiv:2004.08424*, 2020.

[15] A. Henry, *achenry/bldpitch_modeling*. [Online]. Available: `https://github.com/achenry/bldpitch_modeling`.