Using High-fidelity Time-Domain Simulation Data to Construct Multi-fidelity State Derivative Function Surrogate Models for use in Control and Optimization

Athul Krishna Sundarrajan[1] and Daniel Herber[2]

[1] Graduate Student
Department of Systems Engineering
Colorado State University

[2] Assistant Professor
Department of Systems Engineering
Colorado State University

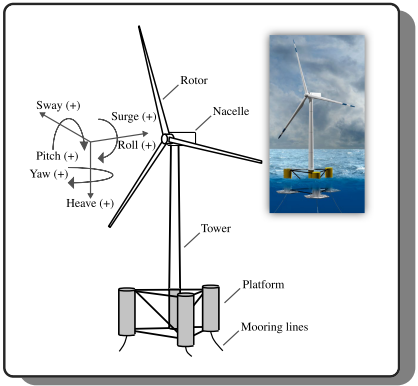
November 1, 2023, IMECE 2023; New Orleans

1

Introduction

→ Introduction

- The goal of this study is to develop surrogate models of dynamic systems that can be used in design optimization studies
- Detailed models of these systems can be computationally expensive
- Numerical programming approaches used to solve the design problem can require several hundred function evaluations
- Using these models directly in design optimization studies can be inefficient



→ Introduction (Continued)

State-Space Representation

$$\frac{d\boldsymbol{\xi}}{dt} = \dot{\boldsymbol{\xi}} = \boldsymbol{f}(\boldsymbol{\xi}(t), \boldsymbol{u}(t), \boldsymbol{p}) \tag{1a}$$

$$\boldsymbol{y}(t) = \boldsymbol{g}(\boldsymbol{\xi}(t), \boldsymbol{u}(t), \boldsymbol{p}) \tag{1b}$$

- For dynamic systems, along with input-output relations, the evolution of the states over the given time horizon is an additional consideration
- Open-loop optimal control studies can be used to identify the optimal inputs $\boldsymbol{u}_{\text{opt}}$ that represent the best possible performance
- Evaluating the evolution of the state trajectories over the given time horizon is computationally expensive
- Creating a surrogate model of this function can help reduce computational time

→ Introduction (Continued)

DFSM

$$\dot{\boldsymbol{\xi}} = \boldsymbol{f}(\boldsymbol{u}, \boldsymbol{\xi}) \approx \hat{\boldsymbol{f}}(\boldsymbol{u}, \boldsymbol{\xi}) \tag{2a}$$

$$\boldsymbol{y} = \boldsymbol{g}(\boldsymbol{u}, \boldsymbol{\xi}) \approx \hat{\boldsymbol{g}}(\boldsymbol{u}, \boldsymbol{\xi}) \tag{2b}$$

- Constructing a surrogate model of this system has been studied under the term **derivative function surrogate model (DFSM)**[1]
- The DFSM provides the state derivative ($\dot{\boldsymbol{\xi}}$) values for given inputs ($\boldsymbol{u}, \boldsymbol{\xi}$)
- The steps involved in the construction are:
  - Create a sampling scheme for the inputs $\boldsymbol{I} = (\boldsymbol{u}, \boldsymbol{\xi})$
  - Evaluate the derivative function $\boldsymbol{O} = \boldsymbol{f}(\boldsymbol{I})$
  - Train a surrogate model that maps $\boldsymbol{I} \rightarrow \boldsymbol{O}$

[1] Deshmukh and Allison 2017; Lefebvre, Belie, and Crevecoeur 2018; Zhang, Wu, and Lu 2022

Caveats

1. Previous studies have assumed direct access to the derivative function
   - But this is not the case with FOWT simulation tool used in this study called WEIS
2. Prior information about linear model structure has not been used in the construction of the model
   - Some state derivatives ($\dot{\boldsymbol{\xi}}$) can be linear relations of inputs $(\boldsymbol{u}, \boldsymbol{\xi})$. For example:

$$\dot{\xi}_3 = \xi_2 - \xi_1 \quad \text{or} \quad \dot{\xi}_1 = \xi_3$$

3. Limited efforts have been made to validate the model once constructed
   - Model refinement schemes have been used to identify the optimal results
4. Key outputs $(\boldsymbol{y})$ are not captured by the model
   - When designing FOWT, quantities such as tower base force/moment and generated power are important

2

DFSM Construction

→ Overview

DFSM

$$\boldsymbol{f}(\cdot) \approx \hat{\boldsymbol{f}}_{low}(\cdot) + \boldsymbol{e}(\cdot) \tag{3}$$

- The goal is to construct a multi-fidelity DFSM model with the structure mentioned above
- $\hat{\boldsymbol{f}}_{\text{low}}(\cdot)$ is a low-fidelity linear-fit model
- $\boldsymbol{e}(\cdot)$ is a higher-fidelity component that attempts to approximate the remaining error
- The derivative function in Eq. (1a) cannot be evaluated directly, but a black box code can be simulated for a given input $\boldsymbol{u} \in \mathbb{R}^{n_u}$ to get the corresponding outputs $\boldsymbol{y} \in \mathbb{R}^{n_y}$.
- The states $\boldsymbol{\xi} \in \mathbb{R}^{n_\xi}$ are available from the outputs of the simulation $\boldsymbol{y}$, and the model does not have any other internal states, such that $\boldsymbol{\xi} \subset \boldsymbol{y}$.

→ Steps Involved

1. Run the necessary simulations to obtain the baseline data for state and output trajectories
2. Construct at least a C^1 continuous polynomial approximation of the state trajectories $\hat{\boldsymbol{\xi}}(t)$ and then evaluate polynomial approximation derivative $\dot{\hat{\boldsymbol{\xi}}}(t)$
3. Using the input-output data, construct a least-squares linear-fit approximation creating $\hat{\boldsymbol{f}}_{\text{low}}$
4. Using the input-output data, evaluate the remaining error between the actual state derivatives and the linear-fit model
5. Train a nonlinear surrogate model on this error using a selected approach determining $\boldsymbol{e}$
6. Validate the resulting multi-fidelity model

→ Step 1

Step (1)

Run the necessary simulations to obtain the baseline data for state and output trajectories

- For a given system we generate total of n_{sim} simulations for different control inputs $\boldsymbol{u}$ to get the corresponding outputs $\boldsymbol{y}(t)$
- From $\boldsymbol{y}$, the state trajectories $\boldsymbol{\xi}$ can be extracted and organized as:

$$\boldsymbol{T} = \begin{bmatrix} \boldsymbol{t}^{(1)} & \boldsymbol{t}^{(2)} & \dots & \boldsymbol{t}^{(n_{\text{sim}})} \end{bmatrix} \tag{4a}$$

$$\boldsymbol{I} = \begin{bmatrix} \boldsymbol{U} \\ \boldsymbol{X} \end{bmatrix} = \begin{bmatrix} \boldsymbol{u}^{(1)} & \boldsymbol{u}^{(2)} & \dots & \boldsymbol{u}^{(n_{\text{sim}})} \\ \boldsymbol{\xi}^{(1)} & \boldsymbol{\xi}^{(2)} & \dots & \boldsymbol{\xi}^{(n_{\text{sim}})} \end{bmatrix} \tag{4b}$$

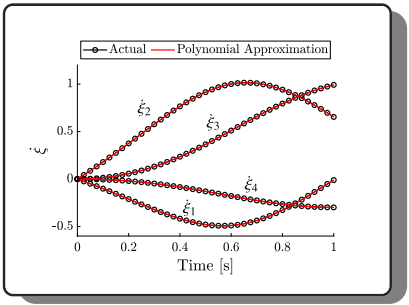
→ Step 2

Step (2)

Construct at least a C^1 continuous polynomial approximation of the state trajectories $\hat{\boldsymbol{\xi}}(t)$ and then evaluate polynomial approximation derivative $\dot{\hat{\boldsymbol{\xi}}}(t)$

- Construct a cubic-spline interpolation scheme for $\boldsymbol{\xi}(t)$ on t
- Cubic-spline interpolation scheme can provide continuous first and second derivatives

$$\dot{\boldsymbol{X}} = \left[\dot{\boldsymbol{\xi}}^{(1)} \quad \dot{\boldsymbol{\xi}}^{(2)} \quad \dots \quad \dot{\boldsymbol{\xi}}^{(n_{\text{sim}})}\right] \quad (5)$$



→ Step 3

Step (3)

Using the input-output data, construct a least-squares linear-fit approximation creating $\hat{\boldsymbol{f}}_{low}$

- The low-fidelity portion is found by constructing a least-squares approximation between the inputs $\boldsymbol{I}$ and the state derivatives $\dot{\boldsymbol{X}}$:

$$\hat{\boldsymbol{f}}_{\text{low}}(\boldsymbol{I}) = \hat{\boldsymbol{f}}_{\text{L}}(\boldsymbol{I}) = \boldsymbol{L}\boldsymbol{I} \tag{6a}$$

$$\boldsymbol{L} = (\boldsymbol{I}\boldsymbol{I}^T)^{-1}\boldsymbol{I}^T\dot{\boldsymbol{X}} \tag{6b}$$

- If the system can be characterized by additional parameters $\boldsymbol{w}$, then a LPV system can be constructed as:

$$\hat{\boldsymbol{f}}_{\text{L}} = \boldsymbol{L}(\boldsymbol{w})\boldsymbol{I} = \begin{bmatrix}\boldsymbol{B}_{\text{L}}(\boldsymbol{w}) & \boldsymbol{A}_{\text{L}}(\boldsymbol{w})\end{bmatrix}\boldsymbol{I} \tag{7}$$

→ Step 4

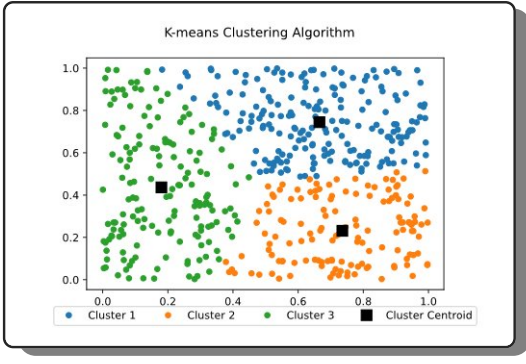
Step (4)

Using the input-output data, evaluate the remaining error between the actual state derivatives and the linear-fit model

- Before constructing the corrective function $\boldsymbol{e}(\cdot)$, it is necessary to subsample from the evaluated error:

$$\boldsymbol{E} = \dot{\boldsymbol{X}} - \boldsymbol{L}\boldsymbol{I} \tag{8}$$

- It is computationally expensive to construct a model using all the data
- We use the k-means method to extract the subsamples



→ Step 5

Step (5)

Train a nonlinear surrogate model on this error using a selected approach determining $\boldsymbol{e}$

- Radial basis functions (RBFs) are used to construct the nonlinear error corrective function $\boldsymbol{e}$ in this study:

$$F(\boldsymbol{x}) = \sum_{i=1}^{N} w_i \cdot \phi(\|\boldsymbol{x} - \boldsymbol{x}_i\|_2) \tag{9a}$$

$$\phi(\boldsymbol{x}) = \exp(-\boldsymbol{x}^2) \tag{9b}$$

→ Step 6

- The sequence of steps can be repeated to get a surrogate model for the outputs $\boldsymbol{y}$

Step (6)

$$\dot{\boldsymbol{\xi}} \approx \hat{\boldsymbol{f}} = \boldsymbol{A}\boldsymbol{\xi} + \boldsymbol{B}\boldsymbol{u} + \boldsymbol{e}_f(\boldsymbol{\xi}, \boldsymbol{u}) \tag{10a}$$

$$\boldsymbol{y} \approx \hat{\boldsymbol{g}} = \boldsymbol{C}\boldsymbol{\xi} + \boldsymbol{D}\boldsymbol{u} + \boldsymbol{e}_g(\boldsymbol{\xi}, \boldsymbol{u}) \tag{10b}$$

3

FOWT Case Study

→ FOWT Model

- We use the IEA-15 MW FOWT model with a semi-submersible platform
- The main states are the platform pitch (Θ_p), generator speed (ω_g), and their first time derivatives ($\dot{\Theta}_p, \dot{\omega}_g$)

$$\boldsymbol{\xi} = [\Theta_p, \omega_g, \dot{\Theta}_p, \dot{\omega}_g]^T \tag{11}$$

$$\dot{\boldsymbol{\xi}} = [\dot{\Theta}_p, \dot{\omega}_g, \ddot{\Theta}_p, \ddot{\omega}_g]^T \tag{12}$$

- The controls are the the generator torque (τ_g) and the blade pitch (β)

$$\boldsymbol{u} = [\tau_g, \beta]^T \tag{13}$$

- The tower base fore-aft shear force (T_F) and side-to-side moment (T_M) are the outputs considered

$$\boldsymbol{y} = [T_F, T_M]^T \tag{14}$$

- System simulations are obtained for ten different trajectories from DLC 1.1
- 80% are used to train the DFSM model, and the rest are used for testing

→ Problem Formulation

- An optimal control problem is formulated to maximize the power produced
- Power generation vs. load reduction is a key trade-off in wind turbine design

Problem Formulation

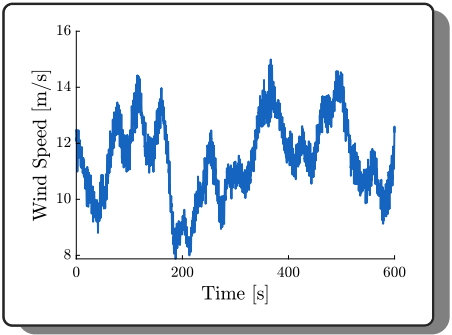
$$\underset{\boldsymbol{u},\boldsymbol{\xi}}{\textit{min:}} \int_{t_0}^{t_f} \left[(-\tau_g \omega_g) + \boldsymbol{u}^T \boldsymbol{W} \boldsymbol{u} \right] \mathrm{d}t \tag{15a}$$

$$\textit{sub to:}\ \dot{\boldsymbol{\xi}} = \hat{\boldsymbol{f}}(\boldsymbol{u}, \boldsymbol{\xi}) \tag{15b}$$

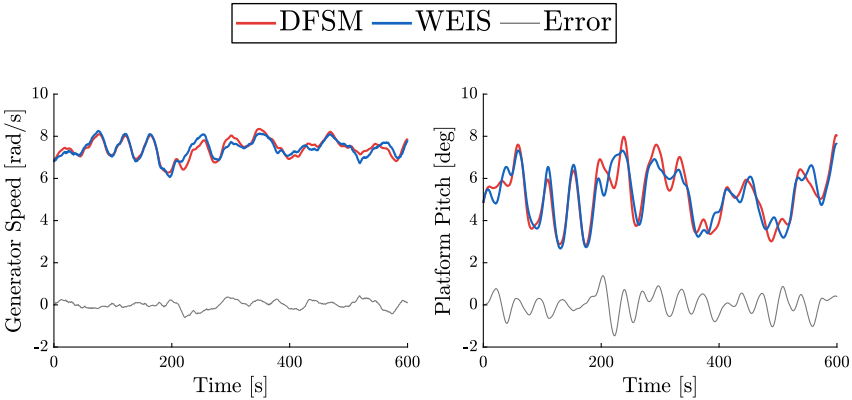
$$\boldsymbol{y} = \hat{\boldsymbol{g}}(\boldsymbol{u}, \boldsymbol{\xi}) \tag{15c}$$

$$\boldsymbol{\xi}_{\min} \leq \boldsymbol{\xi} \leq [\Theta_{p,\max}, 7.2] \tag{15d}$$

$$\Theta_{p,\max} = [5, 7] \quad \textit{[deg]}$$



→ Validation Results

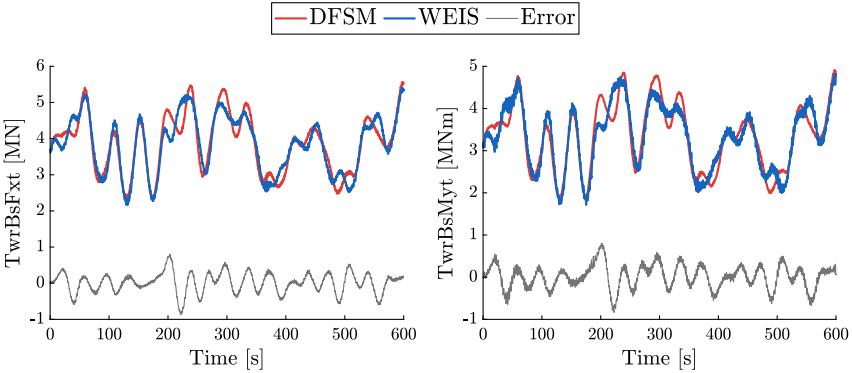


(a) Generator Speed.

(b) Platform Pitch.

WEIS simulation time: 20 minutes, **DFSM simulation time**: 4 minutes

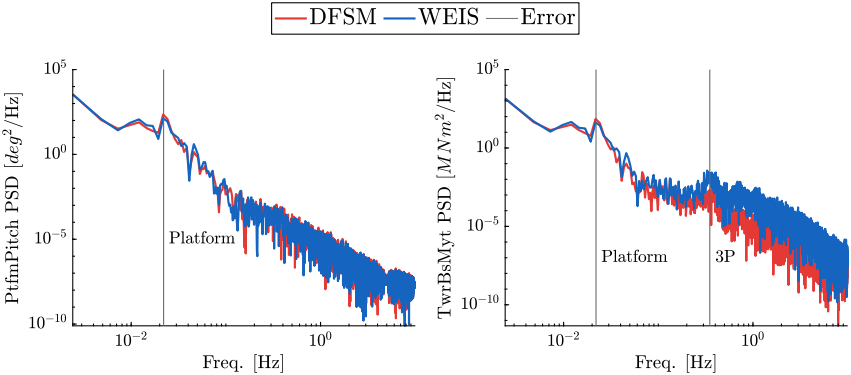
→ Validation Results (Continued)



(a) Tower-base share force.

(b) Tower-base shear moment.

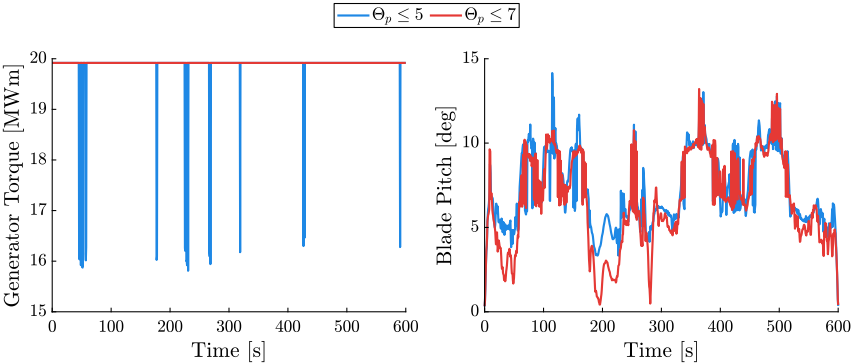
→ Validation Results (Continued)



(a) PSD of platform pitch.

(b) PSD of tower-base shear moment.

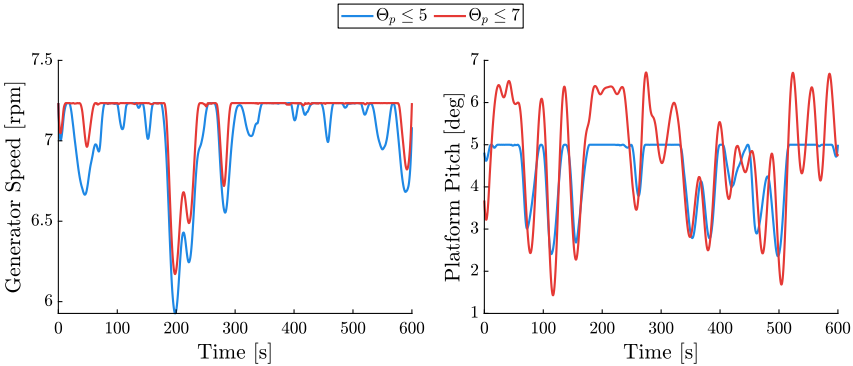
→ Optimal Controls



(a) Optimal τ_g.

(b) Optimal β.

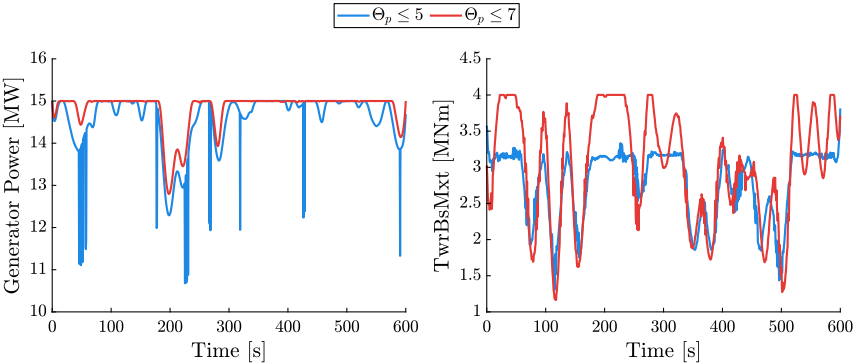
→ Optimal States



(a) Optimal ω_g.

(b) Optimal Θ_p.

→ Outputs



(a) Optimal P.

(b) Optimal T_M.

4

Conclusion

→ FOWT Model

- In this article, we explored the use of a multi-fidelity DFSM approach that can be used to approximate the dynamic model of nonlinear systems.
- We proposed an approach to extract the state derivative information from system simulations
- With the information, the multi-fidelity DFSM consists of a least-squares linear-fit low-fidelity model and an additive nonlinear error corrective function
- The use of the DFSM model to approximate a FOWT is demonstrated
- Application of the DFSM to approximate the response of axial-flow marine tubes has been carried out
- The approach presented here must be extended to include plant variables as inputs to the DFSM
- Scalable nonlinear surrogate modeling approaches can also be explored

→ References

- A. P. Deshmukh and J. T. Allison (2017). "Design of Dynamic Systems Using Surrogate Models of Derivative Functions". *J. Mech. Design* 139.10
- T. Lefebvre, F. D. Belie, and G. Crevecoeur (2018). "A trajectory-based sampling strategy for sequentially refined metamodel management of metamodel-based dynamic optimization in mechatronics". *Optim. Control Appl. Methods* 39.5
- Q. Zhang, Y. Wu, and L. Lu (2022). "A Novel Surrogate Model-Based Solving Framework for the Black-Box Dynamic Co-Design and Optimization Problem in the Dynamic System". *Mathematics* 10.18

Questions?

Using High-fidelity Time-Domain Simulation Data to Construct Multi-fidelity State Derivative Function Surrogate Models for use in Control and Optimization

IMECE, New Orleans

November 1, 2023

Athul Krishna Sundarrajan

Daniel R. Herber

Colorado State University

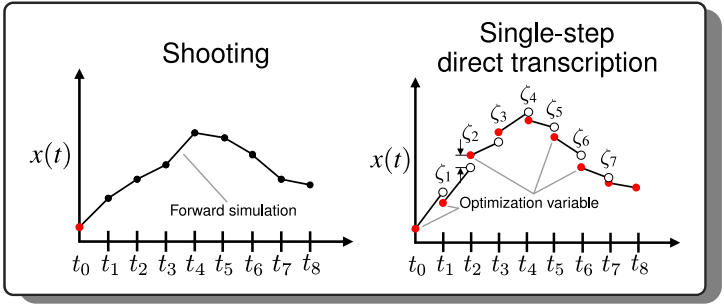
Athul.Sundarrajan@colostate.edu

https://github.com/danielrherber/dt-qp-project

WEIS/examples/17_DFSM

Appendix

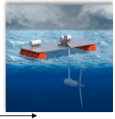
→ Open-Loop Optimal Control

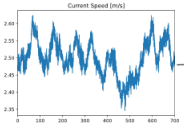


Two popular direct numerical methods used to solve open-loop optimal control problems are the **direct shooting** and **direct transcription** methods that determine the optimal trajectories $\boldsymbol{\xi}_{\text{opt}}(t)$ and $\boldsymbol{u}_{\text{opt}}(t)$

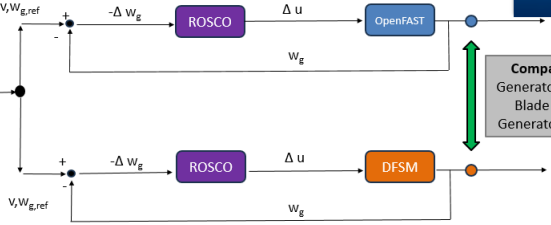
→ Additional Results

Use Case 1: Controls Tuning





Input: Current speed



Comparison: Generator Speed, Blade Pitch, Generator Power

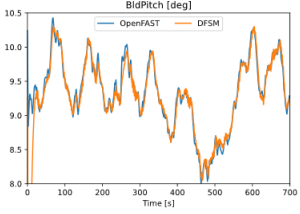
Adopted from Abbas et. al 2022

Use Case 1: Use the DFSM model instead of OpenFAST to obtain the control signals and behavior for a given DLC

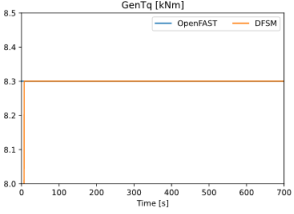
NREL | 8

→ Additional Results (continued)

Controls



Blade Pitch



Generator Torque

About **70x speed-up** for one simulation

1. Only the generator speed DOF is enabled, outputs are not included in the study
2. OpenFAST simulation time : **3.1 hours**
3. DFSM simulation time: **2.7 minutes**, DFSM construction time: **0.97 minutes**